Perancangan dan Implementasi *Image Watermarking* Dengan *Spread Spectrum* Berbasis Android *Platform*

Artikel Ilmiah

Peneliti:
Dennis Oktavianus Sugiharto (672011008)
Magdalena A. Ineke Pakereng, M.Kom.

Program Studi Teknik Informatika
Fakultas Teknologi Informasi
Universitas Kristen Satya Wacana
Salatiga
Mei 2015

Perancangan dan Implementasi *Image Watermarking* Dengan *Spread Spectrum* Berbasis Android *Platform*

Artikel Ilmiah

Diajukan kepada
Fakultas Teknologi Informasi
Untuk Memperoleh Gelar Sajarna Komputer

Peneliti:
Dennis Oktavianus Sugiharto (672011008)
Magdalena A. Ineke Pakereng, M.Kom.

Program Studi Teknik Informasi
Fakultas Teknologi Informasi
Universitas Kristen Satya Wacana
Salatiga

Mei 2015

Perancangan dan Implementasi *Image Watermarking* Dengan *Spread Spectrum* Berbasis Android *Platform*

Oleh

Dennis Oktavianus Sugiharto
NIM : 672011008

Artikel Ilmiah

Diajukan Kepada Program Studi Teknik Informatika, Fakultas Teknologi Informasi guna memenuhi sebagian dari persyaratan untuk mencapai gelar Sarjana Komputer

Disetujui oleh,

Magdalena A. Ineke Pakereng, M.Kom.
Pembimbing

Diketahui oleh,

Dr. Dharmaputra T. Palekahelu, M.Pd.
Dekan

Suprihadi, S.Si., M.Kom.
Ketua Program Studi

Fakultas Teknologi Informasi
Universitas Kristen Satya Wacana
Salatiga
2015

Lembar Pengesahan

Judul Tugas Akhir	:	Perancangan Dan Implementasi *Image Watermarking* Dengan *Spread Spectrum* Berbasis Android *Platform*
Nama Mahasiswa	:	Dennis Oktavianus Sugiharto
NIM	:	672011008
Program Studi	:	Teknik Informatika
Fakultas	:	Teknologi Informasi

Menyetujui,

Magdalena A. Ineke Pakereng, M.Kom.
Pembimbing

Mengesahkan,

Dr. Dharmaputra T. Palekahelu, M.Pd.
Dekan

Suprihadi, S.Si., M.Kom.
Ketua Program Studi

Dinyatakan Lulus Ujian Tanggal 9 Juni 2015

Penguji:

1. T. Arie Setiawan Prasida, ST., M.Cs.
2. Christine Dewi, S.Kom., M.Cs.

Pernyataan

Artikel Ilmiah berikut ini:

Judul : Perancangan dan Implementasi *Image Watermarking* Dengan *Spread Spectrum* Berbasis Android *Platform*

Pembimbing : Magdalena A. Ineke Pakereng, M.Kom.

Adalah benar hasil karya saya :

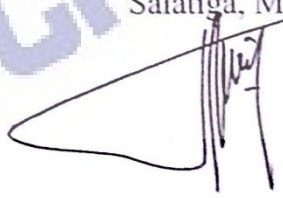
Nama : Dennis Oktavianus Sugiharto

NIM : 672011008

Saya menyatakan tidak mengambil sebagai atau seluruh dari hasil karya orang lain kecuali sebagaimana yang tertulis pada daftar pustaka.

Pernyataan ini dibuat dengan sebenar-benarnya sesuai dengan ketentuan yang berlaku dalam penulisan artikel ilmiah.

Salatiga, Mei 2015



Dennis Oktavianus Sugiharto

FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN SATYA WACANA
Jalan Diponegoro 52 – 60
Phone. (0298) 321212 (Hunting)
Fax. (0298) 321433
E-mail: fti@uksw.edu
Salatiga 50711 – INDONESIA

LEMBAR PERSETUJUAN PUBLISH JURNAL

Dengan mempertimbangkan isi dari jurnal mahasiswa :

Nama Mahasiswa : Dennis Oktavianus S.
NIM : 672011008

Maka jurnal ini dinyatakan :

~~LAYAK TERBIT~~ / TIDAK LAYAK TERBIT

Menyetujui,

Maydalena M.S. Jacobs P.
(.....)
Pembimbing 1

(.....)
Pembimbing 2

Mengetahui,

I. Arie Setyawan P.
(.....)
Penguji 1

Christine Dewi
(.....)
Penguji 2

PERPUSTAKAAN UNIVERSITAS
UNIVERSITAS KRISTEN SATYA WACANA
Jl. Diponegoro 52 – 60 Salatiga 50711
Jawa Tengah, Indonesia
Telp. 0298 – 321212, Fax. 0298 321433
Email: library@adm.uksw.edu ; http://library.uksw.edu

PERNYATAAN PERSETUJUAN AKSES

Saya yang bertanda tangan di bawah ini:

Nama : DENNIS OKTAVIANUS SUGIHARTO

NIM : 672011008 Email : dennisoktavianus@yahoo.co.id

Fakultas : TEKNOLOGI INFORMASI Program Studi : TEKNIK INFORMASI

Judul tugas akhir : PERANCANGAN DAN IMPLEMENTASI IMAGE WATERMARKING DENGAN SPREAD SPECTRUM BERBASIS ANDROID PLATFORM

Dengan ini saya menyerahkan hak *non-eksklusif** kepada Perpustakaan Universitas – Universitas Kristen Satya Wacana untuk menyimpan, mengatur akses serta melakukan pengelolaan terhadap karya saya ini dengan mengacu pada ketentuan akses tugas akhir elektronik sebagai berikut (beri tanda pada kotak yang sesuai):

☑ a. Saya mengijinkan karya tersebut diunggah ke dalam aplikasi Repositori Perpustakaan Universitas, dan/atau portal GARUDA

☐ b. Saya tidak mengijinkan karya tersebut diunggah ke dalam aplikasi Repositori Perpustakaan Universitas, dan/atau portal GARUDA**

* *Hak yang tidak terbatas hanya bagi satu pihak saja. Pengajar, peneliti, dan mahasiswa yang menyerahkan hak non-ekslusif kepada Repositori Perpustakaan Universitas saat mengumpulkan hasil karya mereka masih memiliki hak copyright atas karya tersebut.*

** *Hanya akan menampilkan halaman judul dan abstrak. Pilihan ini harus dilampiri dengan penjelasan/ alasan tertulis dari pembimbing TA dan diketahui oleh pimpinan fakultas (dekan/kaprodi).*

Demikian pernyataan ini saya buat dengan sebenarnya.

Salatiga, 30 Juni 2015

DENNIS OKTAVIANUS. S

Tanda tangan & nama terang mahasiswa

Mengetahui,

MAGDALENA A. INEKE PAKERONG, M. Kom.

Tanda tangan & nama terang pembimbing I

Tanda tangan & nama terang pembimbing II

PERPUSTAKAAN UNIVERSITAS
UNIVERSITAS KRISTEN SATYA WACANA
Jl. Diponegoro 52 – 60 Salatiga 50711
Jawa Tengah, Indonesia
Telp. 0298 – 321212, Fax. 0298 321433
Email: library@adm.uksw.edu ; http://library.uksw.edu

PERNYATAAN TIDAK PLAGIAT

Saya yang bertanda tangan di bawah ini:

Nama : DENNIS OKTAVIANUS SUGIHARTO

NIM : 672011008 Email : dennisoktavianus@yahoo.co.id

Fakultas : Teknologi Informasi Program Studi : Teknik INFORMATIKA

Judul tugas akhir : PERANCANGAN DAN IMPLEMENTASI IMAGE WATERMARKING DENGAN SPREAD SPECTRUM BERBASIS ANDROID PLATFORM

Pembimbing : 1. MAGDALENA A. INEKE PAKERENG., M.KOM.

2. _____

Dengan ini menyatakan bahwa:

1. Hasil karya yang saya serahkan ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar kesarjanaan baik di Universitas Kristen Satya Wacana maupun di institusi pendidikan lainnya.
2. Hasil karya saya ini bukan saduran/terjemahan melainkan merupakan gagasan, rumusan, dan hasil pelaksanaan penelitian/implementasi saya sendiri, tanpa bantuan pihak lain, kecuali arahan pembimbing akademik dan narasumber penelitian.
3. Hasil karya saya ini merupakan hasil revisi terakhir setelah diujikan yang telah diketahui dan disetujui oleh pembimbing.
4. Dalam karya saya ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali yang digunakan sebagai acuan dalam naskah dengan menyebutkan nama pengarang dan dicantumkan dalam daftar pustaka.

Pernyataan ini saya buat dengan sesungguhnya. Apabila di kemudian hari terbukti ada penyimpangan dan ketidakbenaran dalam pernyataan ini maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya saya ini, serta sanksi lain yang sesuai dengan ketentuan yang berlaku di Universitas Kristen Satya Wacana.

Salatiga, 30 JUNI 2016

DENNIS OKTAVIANUS.S

Tanda tangan & nama terang mahasiswa

F-LIB-080

Perancangan dan Implementasi *Image Watermarking* Dengan *Spread Spectrum* Berbasis Android *Platform*

Dennis Oktavianus Sugiharto [1], **Magdalena A. Ineke Pakereng** [2]
Fakultas Teknologi Informasi
Universitas Kristen Satya Wacana
Jl. Diponegoro 52-60, Salatiga 50711, Indonesia
E-mail: dennisoktavianus@yahoo.co.id[1], ineke.pakereng@staff.uksw.edu[2]

Abstract

Ease of deployment of digital images via the internet has a positive side and Negative especially for owners of the original digital image. The positive side of the ease of rapid deployment is the owner of the image spread of digital image files on various media. While the downside is that if there is no copyright that serves as protector of the image it will be very easily recognized kepemelikannya by other parties. Watermarking is one solution to protect the copyright and know the results of the image. With Image Watermarking, the copyright resulting image will be protected through the insertion of additional information such as owner information and the authenticity of the image. Least Significant Bit (LSB) is one algorithm that is simple and easy to understand. Digital image watermarking insertion into the LSB run on mobile media. This application is made to the Java programming language with the Android SDK and Library using Eclipse tools. The results show that the cover image before and after insertion of visually revealed no significant differences.

***Keywords**: Watermarking, Spread Spectrum, LSB Modified,* Aplikasi *Mobile,* Android

Abstrak

Kemudahan penyebaran citra digital melalui internet memiliki sisi positif dan negatif terutama bagi pemilik asli citra digital tersebut. Sisi positif dari kemudahan penyebaran adalah dengan cepatnya pemilik citra tersebut menyebarkan file citra digital pada berbagai media. Sedangkan sisi negatifnya adalah jika tidak ada hak cipta yang berfungsi sebagai pelindung citra maka akan sangat mudah diakui kepemelikannya oleh pihak lain. *Watermarking* merupakan salah satu solusi untuk melindungi hak cipta dan mengetahui hasil dari *image*. Dengan *Image Watermarking*, hak cipta *image* yang dihasilkan akan terlindungi melalui penyisipan informasi tambahan seperti informasi pemilik dan keaslian pada *image*. *Least Significant Bit* (LSB) merupakan salah satu algoritma yang sederhana dan mudah dipahami. Penyisipan watermarking ke citra digital dengan LSB dijalankan pada media *mobile*. Aplikasi ini dibuat dengan bahasa program Java dengan *Library* Android SDK dan dengan menggunakan tools *Eclipse*. Hasil pengujian menunjukkan bahwa *cover image* sebelum dan sesudah disisipi secara visual tidak menampakkan perbedaan yang signifikan.

Kata Kunci : *Watermarking, Spread Spectrum,* LSB Termodifikasi, Aplikasi *Mobile,* Android

[1] Mahasiswa Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana Salatiga
[2] Staf Pengajar Fakultas Teknologi Informasi Universitas Kristen Satya Wacana Salatiga

1. Pendahuluan

Salah satu karya yang intelektual yang dilindungi adalah barang dalam bentuk digital, seperti *software* dan produk multimedia seperti teks, music (dalam format MP3 atau WAV), gambar atau citra (*image*) dan video digital (VCD). Selama ini penggandaan atas produk digital tersebut dilakukan secara bebas dan leluasa. Pemegang hak cipta atas produk digital tersebut tentu dirugikan karena pemilik citra tidak mendapat royalti dari usaha penggandaan tersebut. Sebenarnya masalah penyalahgunaan hak cipta pada bidang multimedia tidak hanya mengenai penggandaan dan pendistribusiannya saja, tetapi juga mengenai label kepemilikan. Kebanyakan produk digital tersebut tidak mencantumkan siapa pemegang hak ciptanya. Walaupun bukti kepemilikan itu ada, biasanya informasi kepemilikan disertakan pada sampul pembungkus yang menerangkan bahwa produk multimedia tersebut adalah milik pembuatnya. Masalahnya, distribusi produk multimedia saat ini tidak hanya secara *offline*, tetapi juga dapat dilakukan lewat internet dan media lainnya yang saat ini sudah sangat mudah dimiliki seperti *smart phone* [1]. Salah satu cara untuk melindungi hak cipta multimedia adalah dengan menyisipkan informasi kedalam data multimedia tersebut dengan teknik *watermarking*. Informasi yang disisipkan kedalam data multimedia tersebut disebut *watermark* dan *watermark* dapat dianggap pesan teks dari pemilik sah atas produk multimedia tersebut. Dengan demikian, *watermark* yang disisipkan menjadi hak cipta dari pemiliknya. Pemberian pesan berupa teks dengan teknik *watermarking* ini dilakukan sedemikian sehingga nformasi yang disisipkan tidak merusak data digital yang dilindungi. Sehingga orang yang membuka produk multimedia yang sudah disisipkan *watermark* tidak menyadari kalau didalam data multimedia tersebut terkandung label kepemilikan pembuatnya.

Penerapan *watermarking* citra digital dapat dilakukan pada tipe gambar hitam putih (*grayscale*) ataupun citra berwarna (RGB) dengan format yang bermacam-macam, seperti : JPEG, BMP, TIFF atau PNG, sehingga dalam penerapan watermarking citra digital tidak mengacu pada tipe gambar dan format satu saja [2].

Berdasarkan latar belakang permasalahan terkait keamanan hak cipta suatu citra pada berbagai bidang tersebut dirancang sebuah aplikasi untuk membuktikan hak cipta citra dengan mengambil judul penelitian perancangan dan implementasi *image watermarking* dengan spread spectrum berbasis android *platform*.

2. Tinjauan Pustaka

Penelitian yang berjudul "*Perancangan Steganografi Dengan Media Gambar Pada Aplikasi Berbasis Android*", dibahas mengenai penggunaan metode LSB termodifikasi yang diimplementasikan untuk menyisipkan pesan pada *file* citra berbasis android. Hasil dari penelitian tersebut menunjukkan bahwa dengan metode LSB termodifikasi ukuran *file* citra tidak mengalami perubahan dalam segi ukuran setelah dilakukan proses *encoding*. Selain itu metode tersebut cocok untuk semua file citra [3].

Penelitian yang berjudul "*Perancangan Aplikasi Watermarking Pada Media Fotografi Sebagai Perlindungan Hak Cipta Menggunakan Metode Spread Spectrum*", membahas tentang teknik penyembunyian data atau informasi yang bersifat rahasia ke data lain. Informasi disandikan dengan menggunakan *Spread Spectrum*. Pada penelitian tersebut disimpulkan bahwa *Spread Spectrum* memiliki performa yang sedikit lebih baik dari algoritma yang lain [4].

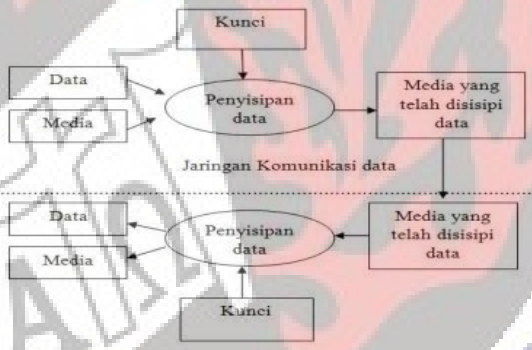
Berdasarkan penelitian yang sudah pernah dilakukan terkait *Watermarking*, maka akan dilakukan penelitian yang membahas tentang Perancangan dan Implementasi *Image Watermarking* dengan *Spread Spectrum* Bebasis Android *Platform*. Aplikasi yang dibangun menggunakan bahasa pemograman Java dengan Library Android SDK dan dengan menggunakan tools *Eclipse* yang berfungsi untuk melabelkan data gambar dengan label yang diinginkan. Aplikasi yang dibangun berfungsi untuk menyisipkan berkas *watermark* berupa teks pada *image* yang bertujuan untuk memproteksi *image* tersebut. Teknik yang digunakan dalam implementasi adalah *spread spectrum*, dimana berkas *watermark* disisipkan secara berurut ke dalam domain spasial sehingga keberadaan *watermark* tidak dapat dipersepsi oleh panca indera dan memiliki kekokohan yang baik. Perbedaan dengan perancangan sebelumnya adalah aplikasi *image watermarking* yang implementasikan pada data *mobile* dengan menggunakan metode *spread spectrum* yang dimana kapasitas penyisipan lebih banyak dan dalam teknik penyisipan berkas *watermark* teks pada *image* dilakukan secara berurut, dan aplikasi *image watermarking* yang dibuat untuk menanamkan sebuah berkas informasi *watermark* ke dalam *image* yang berformat *.PNG.

Portable Network Graphic (PNG) format dirancang agar menjadi lebih baik dengan format yang terdahulu yaitu GIF dan sudah dilegalkan. PNG di rancang untuk algoritma *losslessley* untuk menyimpan sebuah bitmap image.PNG mempunyai persamaan fitur dengan GIF salah satunya adalah (*multiple images*), meningkatkan sesuatu contohnya (*interlacing*, kompresi) dan penambahan fitur-fitur yang terbaru (gamma *storage*, full alpha channel, *true color support, error detection*). Medukung untuk Web *browser* dimana dapat dilakukan *plug-ins* pada web *browser*. PNG mampu mencapai 16 bit (*grayscale*) atau 48 bit untuk *true color* per pixel, dan mencapai 16 bits dari alpha data. PNG mendukung dua buah metode dari transparansi, satu buah color penutup seperti pada GIF89a's dan alpa channel. PNG's dengan Full alpha channel mampu mencapi 64K level dari transparansi untuk masing-masing pixel (216 = 65.536) memungkinkan PNG dapat membuat gambar lebih bercahaya dan membuat bayang-bayang *background* dari pewarnaan yang berbeda [5].

Watermarking memanfaatkan kekurangan-kekurangan sistem indera manusia, seperti mata dan telinga. Berdasarkan kekurangan inilah, *watermarking* dapat diterapkan pada berbagai media *digital*. Jadi, *watermarking* merupakan suatu cara untuk penyembunyian atau penamaan data/informasi tertentu (baik hanya berupa catatan umum maupun rahasia) ke dalam suatu data *digital* lainnya, tetapi tidak diketahui kehadirannya oleh indera manusia (indera pengelihatan dan

pendengaran) dan mampu menghadapi proses pengolahan sinyal *digital* sampai pada tahap tertentu. *Digital Image Watermarking* dapat diklasifikasikan berdasarkan domain menjadi dua jenis yaitu: Domain spasial (piksel) *watermark* ditanamkan pada piksel tertentu pada suatu *image* contohnya *Least Significant Bit* (LSB) dan algoritma *pseudorandom*. Domain frekuensi diperoleh dengan melakukan *transformasi image*, contoh *transformasi image* adalah *Discrete Cosine Transform* (DCT), *Discrete Wavelet Transform* (DWT), *Discrete Fourier Transform* (DFT). Stuktur *watermarking* pada data *digital* seperti *text*, citra, video, *audio*, dilakukan langsung pada jenis data *digital* tersebut (misalnya untuk citra dan video pada domain spasial, dan *audio* pada domain waktu) atau terlebih dahulu dilakukan transformasi ke dalam domain yang lain. Berbagai transformasi yang dikenal dalam pemrosesan sinyal *digital* seperti FFT (*Fast Foirier Transform*), DCT (*Discrete Cosine Transform*), DWT (*Discrete Wavelet Transform*), dan sebagainya [5].

Penerapan *watermarking* pada berbagai domain dengan berbagai *transform* turut mempegaruhi berbagai parameter penting dalam *watermarking*. Terdapat tiga sub-bagian *watermarking* yang membentuknya yaitu: 1) penghasil label *watermark*; 2) proses penyembunyian label; 3) menghasilkan kembali label *watermark* dari data yang ter-*watermarking* [6].



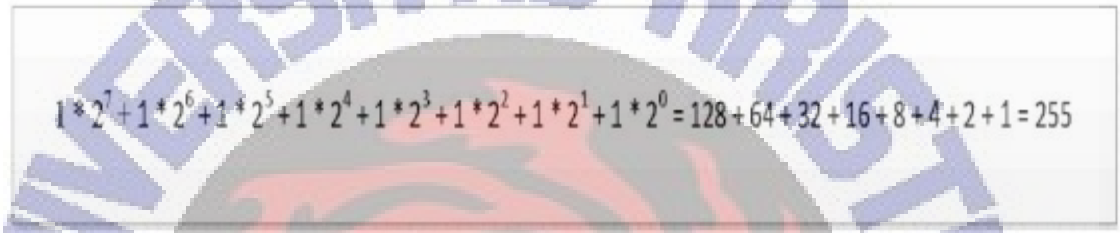
Gambar 1 Bagan Sistem *Watermark* [6].

Bagan Sistem *Watermark* pada Gambar 1 menjelaskan bahwa label *watermark* adalah sesuatu data/informasi yang akan dimasukkan ke dalam data *image* yang ingin di-*watermark*. Ada dua jenis label yang akan digunakan : 1) Teks biasa : Label *watermark* dari teks biasanya menggunakan nilai-nilai ASCII dari masing-masing karakter dalam teks yang kemudian dipecah atas satu *bit* saja akan memberikan hasil yang berbeda dengan teks sebenarnya; 2) Logo atau *image* maupun suara : Berbeda dengan teks, kesalahan pada beberapa *bit* masih dapat memberikan persepsi yang sama dengan aslinya oleh pendengaran maupun penglihatan, tetapi kerugianya adalah jumlah data yang cukup besar. *Key* pada Gambar 1 digunakan untuk mencegah penghapusan secara langsung *watermark* oleh pihak yang tidak bertanggung jawab. Sedangkan untuk ketahanan proses pengolahan lainya, tergantung pada metode *watermarking* yang digunakan. Tetapi dari berbagai penelitian yang dilakukan belum ada suatu metode *watermarking* yang ideal yang dapat tahan terhadap semua proses pengolahan data rahasia tersebut, dan dari masing-masing penelitian lebih memfocuskan pada hal-hal tertentu yang dianggap penting [6].

Metode *spread spectrum* pada penelitian ini diilhami dari skema komunikasi *spread spectrum*, yang mentransmisikan sebuah sinyal pita sempit ke dalam sebuah kanal pita lebar dengan menambah redudansi (pengeluaran) *bit-bit* data sehingga diharapkan dapat meningkatkan *robustness* [7].

Least Significant Bit adalah bagian dari barisan data biner (basis dua) yang mempunyai nilai paling kecil. Letaknya adalah paling kanan dari barisan *bit*. Sedangkan *most significant bit* adalah sebaliknya, yaitu angka yang paling besar dan letaknya disebelah paling kiri [9].

Contohnya adalah bilangan biner dari 255 adalah 11111111 (kadang-kadang diberi huruf b pada akhir bilangan tersebut sehingga menjadi 1111 1111b). bilangan tersebut dapat berarti pada Gambar 2.

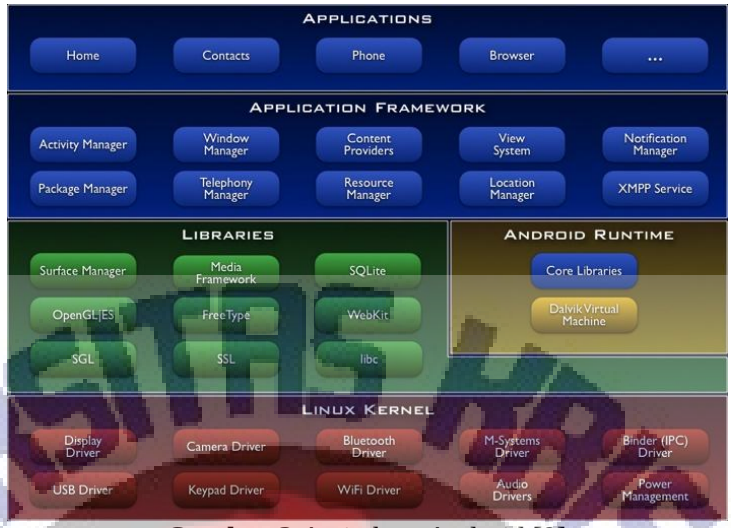


Gambar 2 Bilangan Biner [9]

Barisan angka 1 pada Gambar 2, dijelaskan sebagai berikut. Angka 1 paling kanan bernilai 1, dan merupakan nilai paling kecil, yang disebut dengan *least significant bit* (*bit* yang paling kecil), sedangkan bagian paling kiri bernilai 1128 dan disebut dengan *most significant bit* (*bit* yang paling besar).

Aplikasi *Mobile* adalah program yang digunakan untuk melakukan sesuatu pada sistem komputer. *Mobile* dapat diartikan sebagai perpindahan yang mudah dilakukan dari suatu tempat ke tempat lain, misalnya telepon gengga, yang dapat digunakan dengan berpindah-pindah tempat dengan mudah dari suatu tempat ke tempat lain tanpa pemutusan atau terputusnya komunikasi. Sedangkan akases informasi dari aplikasi *mobile* biasanya hanya berupa teks sederhana. Apabila berupa gambar, maka berupa gambar dengan ukuran yang tidak terlalu besar [8].

Android adalah sebuah sistem operasi *mobile* yang *open source*. Android sendiri mencangkup sistem operasi *middleware*, dan aplikasi mobile yang berbasis Linux Kernel yang dikembangkan oleh Google dan *Open Handset Alliance*. Tujuan dari android *opem source project* adalah untuk membangun produk *real-world* yang dapat meningkatkan pengalamam para pengguna perangkat *mobile*. Adapun arsitektur android dimana android memiliki banyak komponen dalam arsitektur pembangunannya. Berikut ini merupakan gambaran arsitektur android yang terbagi menjadi beberapa layer [8].



Gambar 3 Arsitektur Android [8]

3. Metode Dan Perancangan Sistem

Penelitian yang dilakukan, diselesaikan melalui tahapan penelitian yang terbagi dalam empat tahapan, yaitu: (1) Identifikasi Masalah dan Studi Literatur, (2) Perancangan Sistem, (3) Implementasi Sistem, dan (4) Pengujian sistem dan analisis hasil pengujian.

Gambar 4 Tahapan Penelitian [10]

Tahapan penelitian pada Gambar 4, dapat dijelaskan sebagai berikut. *Tahap pertama*: identifikasi masalah, yaitu mengidentifikasi masalah-masalah yang akan dibahas serta mendapatkan data literatur yang terkait dengan proses *embedding* dan *extracting* terhadap data teks pada *cover image*, menggunakan Metode *Spread Spectrum* pada gambar yang berwarna cerah; *Tahap kedua*: perancangan sistem yang meliputi perancangan proses *embedding* dan *extracting* pada sistem *watermarking* yang akan dibangun; *Tahap ketiga*: implementasi sistem, yaitu mengimplementasikan tahapan penelitian pertama dan kedua ke dalam sebuah program, dengan membuat aplikasi/program sesuai kebutuhan sistem berdasarkan perancangan sistem yang telah dilakukan. Misalnya aplikasi/program yang dapat menyembunyikan pesan rahasia yang disisipkan pada

data *image* dengan menggunakan metode *Spread Spectrum*; dan *Tahap keempat*: pengujian sistem dan analisis hasil pengujian, yaitu dilakukan pengujian terhadap *watermarking* yang dibangun dengan tujuan untuk mengetahui pengaruh *encoding* dan *decoding* pada proses suatu sistem yang akan dijalankan. Proses LSB**,** dijelaskan pada Contoh 1.

Contoh 1 :

Jika digunakan *image* 24 bit color sebagai *cover*, sebuah bit dari masing-masing komponen *Red, Green* dan *Blue* dapat digunakan sehingga 3 bit dapat disimpan pada setiap pixel. Sebuah image 800 x 600 pixel dapat digunakan untuk menyimpan 1.440.000 bit (180.000 *byte*) data rahasia.

```
00100111 11101001 11001000
00100111 11001000 11101001

11001000 00100111 11101001
```

Jika diinginkan untuk menyembunyikan karakter A (10000001b) dihasilkan :

```
00100111 11101000 11001000
00100110 11001000 11101000

11001000 00100111 11101001
```

Perubahan pada LSB ini akan terlalu kecil untuk terdeteksi oleh mata manusia sehingga pesan dapat disembunyikan secara efektif. Pemilihan image harus dilakukan secara teliti, karena perubahan pada LSB dapat menyebabkan terjadinya perubahan warna yang ditampilkan pada citra. Akan lebih baik jika image yang digunakan berupa image grayscale karena perubahan warnanya akan lebih sulit dideteksi oleh mata manusia.

Proses *Seed* Awal, dijelaskan sebagai berikut. Hal petama dilakukan dalam proses *embedding* adalah dengan menentukan *seed* awal terlebih dahulu, dengan cara mengubah *password* (sesuai keinginan *user*) ke dalam bentuk string biner.

Contoh 2 :

Password : "Okta"

O	nilai ascii	79	string biner	01001111
k	nilai ascii	107	string biner	01101011
t	nilai ascii	116	string biner	01110010
a	nilai ascii	97	string biner	01100001

Proses string biner karakter pertama yaitu "O" akan dilakukan fungsi *Exclusive*-OR dengan string biner karakter berikutnya yaitu "k" untuk menghasilkan nilai string biner yang baru.

"O" *Exclusive*-OR "t" = 00100100

Selanjutnya hasil dari "O" *Exclusive*-OR "k" akan di- *Exclusive*-OR lagi dengan string biner karakter berikutnya yaitu "t".

"Ok" *Exclusive*-OR "t" = 01010110

Maka didaptkan hasil dari "Ok" *Exclusive*-OR "t" akan di- *Exclusive*-OR dengan string biner karakter berikutnya "a".

"Okt" *Exclusive*-OR "a" = 00110111

Maka didapatkan hasil dari *Exclusive*-OR Okta = 00110111, yaitu jika diubah menjadi nilai desimal adalah 55 yang merupakan *seed* awal (Xo).

Proses *Spreading*, dijelaskan sebagai berikut. Setelah didapatkan *seed* awal maka langkah selanjutnya yaitu memasukan pesan yang akan disisipkan. Pesan yang akan disisipkan adalah "fti".

Contoh 3 :

f	nilai ascii	102	string biner	01100110
t	nilai ascii	105	string biner	01110100
i	nilai ascii	116	string biner	01101001

digabungkan menjadi : 01100110.01110100.01101001

Proses pertama yang dilakukan terhadap pesan rahasia dalam metode *Spread Spectrum* adalah dengan melakukan proses *spreading* pesan biner sebanyak *input* (sesuai keinginan *user*), misalkan pada contoh ini dilakukan proses *spreading* pesan biner sebanyak 4 kali sehingga akan menghasilkan segmen baru, yaitu :

00001111.11110000.00001111.11110000.00001111.11111111.
00001111.00000000.00001111.11110000.11110000.00001111

Terlihat bahwa setiap *bit* dalam segmen pesan akan mengalami penggandaan bit sebanyak 4 kali dan ukuran segmen pesan menjadi panjang 4 kali dari ukuran segmen semula.

Proses Modulasi, dijelaskan sebagai berikut. Langkah selanjutnya adalah proses modulasi terhadap *bit-bit* yang telah mengalami proses *spreading* tersebut dengan mengacaknya menggunakan *pseudonoise signal* yang dibangkitkan dengan algoritma LCG (*Linear Congruential Generator*). Pada sistem yang dibangun, rumus ini menggunakan tiga konstanta, yaitu a=12, c=19 dan m=255. Fomulasi deretan bilangan bulat *pseudorandom* adalah sebagai berikut :

$Xn+1 = (aXn + c)$ modulus m

Dalam hal ini :

Xn : bilangan bulat ke-n
a : bilangan pengali
c : bilangan penambah
m : modulus
Xo : bila awal berupa bilangan bulat tidak negatif

Perhitungan pembangkitan bilangan acak LCG adalah seperti berikut :

Contoh 4 :

Seed awal (Xo) = 55
a = 12
c = 19
m = 255

Maka dapat dihitung bahwa :

$X_1 = (12 * 55 + 19)$ *mod* $255 = 169$
$X_2 = (12 * 169 + 19)$ *mod* $255 = 7$
$X_3 = (12 * 7 + 19)$ *mod* $255 = 103$
$X_4 = (12 * 103 + 19)$ *mod* $255 = 235$
$X_5 = (12 * 235 + 19)$ *mod* $255 = 34$
$X_6 = (12 * 34 + 19)$ *mod* $255 = 172$
$X_7 = (12 * 172 + 19)$ *mod* $255 = 43$

X_8 = (12 * 43 + 19) *mod* 255 = 25
X_9 = (12 * 25 + 19) *mod* 255 = 64
X_{10} = (12 * 64 + 19) *mod* 255 = 22
X_{11} = (12 * 22 + 19) *mod* 255 = 28
X_{12} = (12 * 28 + 19) *mod* 255 = 100

Fungsi pembangkitan ini dilakukan sesuai jumlah piksel pada *image*. *Noise* yang dihasilkan ini tidak betul-betul bilangan acak, karena ada nilai awal pembangkitnya. Oleh sebab itu, disebut *pseudonoise signal*. Kemudian nilai desimal tersebut akan diubah ke dalam bentuk biner yang hasilnya adalah sebagai berikut :

10101001.00000111.01100101.11110000.00100010.10101100.
00101011.00011001.01000000.00010110.00011100.01100100

Maka untuk mendapatkan hasil modulasi, segmen pesan akan dimodulasi dengan *pseudonoise signal* menggunakan fungsi *Exclusive*-OR.

Diketahui :

Segmen pesan :

00001111.11110000.00001111.11110000.00001111.11111111.
00001111.00000000.00001111.11110000.11110000.00001111

Pseudonoise signal :

10101001.00000111.01100101.111.1000.00100010.10101100.
00101011.00011001.01000000.00010110.00011100.01100100

Maka hasil proses modulasi antara *segmen pesan* dengan *pseudonoise signal* menggunakan fungsi *Exclusive*-OR adalah :

10100110.11110111.01101010.00011000.00101101.01010011.
00100100.00011001.01001111.11100110.11101100.01101011

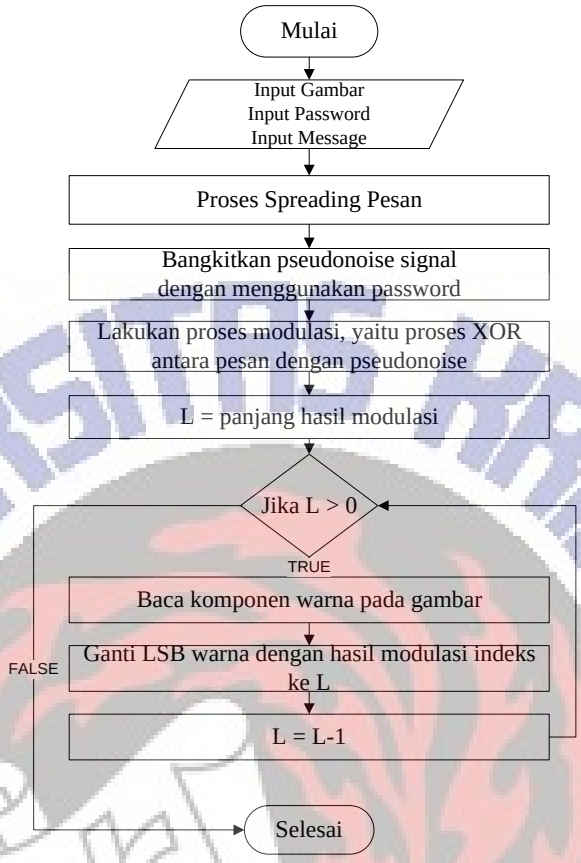
Hasil dari proses modulasi inilah yang akan disisipkan ke *bit-bit* LSB. Setelah disisipi hasil modulasi pada proses sebelumnya, maka data *raster* tersebut menjadi :

Red	*Green*	*Blue*
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**

1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**
1 1 1 1 1 1 1 **0**	1 1 1 1 1 1 1 **1**	1 1 1 1 1 1 1 **1**

Hasil ini dapat diartikan bahwa rata-rata setiap 1 piksel pada *image* akan disisipi 3 *bit* data hasil modulasi.

Proses *Embedding*, dijelaskan sebagai berikut. Proses *Embedding* hanya data dilakukan jika lokasi yang dipilih oleh *user* memiliki jumlah yang sama aau lebih besar dari panjang *bit* pesan. Jika *user* memilih lokasi yang tidak mencukupi, maka sistem akan memberikan pesan kesalahan. Hal ini dikarenakan tiap 1 (satu) *bit* pesan memerlukan 1 (satu) *byte* lokasi penyisipan. Untuk menyisipkan 1 (satu) karakter (1 karakter ASCII = 1 *byte*, 1 *byte* = 8 *bit*), maka diperlukan 8 (delapan) lokasi (warna). Proses *Embedding* dalam sistem *watermarking*, terdiri dari proses *Spreading* (merentangkan) pesan sebesar 1 sampai 100 kali, hal ini akan membuat pesan menjadi 1 sampai 100 kali lebih panjang. Kemudian membangkitkan *pseudonoise* yang memiliki panjang sama dengan pesan yang telah direntangkan, dengan menggunakan *password* yang dimasukan. Berikutnya dilakukan proses modulasi yaitu proses *Exclusive*-OR antara pesan hasil rentang dengan *pseudonoise*. Proses modulasi ini berfungsi untuk menyandikan pesan. Selanjutnya adalah menyisipkan *bit-bit* hasil modulasi ke dalam komponen warna pada *image*. Proses *embedding* dalam bentuk *flowchart*, dapat dilihat pada Gambar 5.



Gambar 5 Proses *Embedding*

Proses *Extracting*, dijelaskan sebagai berikut. Proses *extracting* data merupakan proses untuk membaca pesan yang disisipkan di dalam media penampung (*cover*). Proses *extracting* dalam sistem yang dibangun dijelaskan sebagai berikut. Pilih *file* yang berisi pesan rahasia yang telah disisipkan kemudian masukan *extract key*. Hasil dari variabel penampung akan ditampilkan oleh program. Proses *Extracting* dalam sistem *watermarking*, merupakan kebalikan dari proses *embedding*. Proses dimulai dengan membaca *bit* dari komponen warna, mengumpulkan *bit* tersebut ke dalam satu variabel hasil *extracting*. Dengan menggunakan *password* yang dimasukan, dibangkitkan variabel *pseudonoise*. Selanjutnya adalah proses demodulasi untuk memperoleh *bit* pesan. *Bit* pesan yang dihasilkan, masih dalam bentuk *spreading*, sehingga perlu disusutkan. Proses *extracting* dalam bentuk *flowchart* dapat dilihat pada Gambar 6.

Gambar 6 Proses *Extracting*

Hal pertama yang dilakukan dalam proses *extracting* adalah dengan mengambil *bit-bit* LSB pada *image* untuk mendapatkan *bit-bit* hasil modulasi. Selanjutnya dilakukan proses penyaringan agar mendapat *bit-bit* hasil modulasi. Setelah semua *bit-bit* hasil modulasi diperoleh, kemudian dilakukan proses demodulasi dengan *pseudonoise signal* yang sama pada proses modulasi agar memperoleh *bit-bit* yang berkolerasi, dijelaskan sebagai berikut :

Contoh 6 (diambil dari Contoh 5) :

Hasil *extracting* :

```
10100110.11110111.01101010.00011000.00101101.01010011.00100100.00011001.01001111.11100110.11101100.01101011
```

Pseudonoise signal :

```
10101001.00000111.01100101.111.1000.00100010.10101100.00101011.00011001.01000000.00010110.00011100.01100100
```

Maka hasil proses demodulasi antara hasil *extracting* dengan *pseudonoise signal* menggunakan fungsi *Exclusive*-OR adalah :

```
00001111.11110000.00001111.11110000.00001111.11111111.00001111.00000000.00001111.11110000.11110000.00001111
```

Hasil ini belum merupakan isi pesan yang sesungguhnya.

Proses *De-Spreading*, dijelaskan sebagai berikut. Dibutuhkan sebuah proses lagi agar isi pesan yang sebenarnya dapat ditangkap yaitu dengan perhitungan terhadap faktor pengali proses *spreading*. Proses berikut dinamakan

proses *de-spreading* yang berguna untuk menyusutkan hasil demodulasi menjadi isi pesan yang sebenarnya.

Contoh 7 (diambil dari Contoh 6) :

```
00001111.11110000.00001111.11110000.00001111.11111111.00001111.00000000.00001111.11110000.11110000.00001111
```

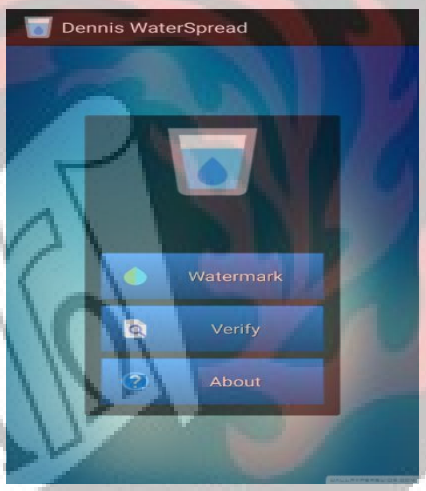
Maka terlihat suatu pola pengulangan dari *bit-bit* tersebut yaitu 4 kali *bit* "1", 4 kali *bit* "0", 8 kali *bit* "1", dan 8 kali *bit* "0". Dengan demikian faktor pengalinya dapat ditentukan yaitu 4 dan proses penyusutan (*de-spreading*) segmen tersebut menjadi :

```
01100110.01110100.01101001
```

String biner di atas merupakan hasil dari *de-spreading* yang jika dikonveri ke dalam karakter ASCII adalah mengandung arti "fti".

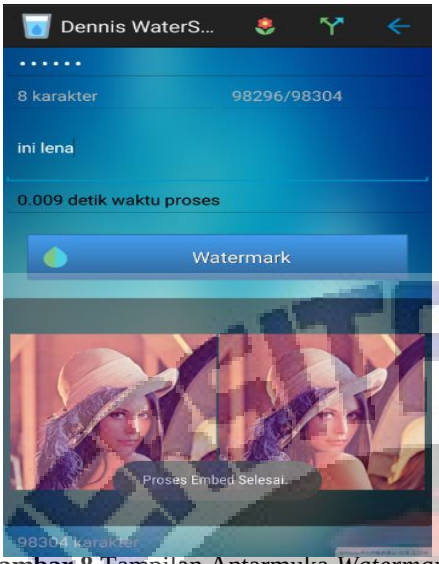
4. Hasil dan pembahasan

Hasil implementasi sistem *watermarking* yang dibangun, ditunjukan pada Gambar 7.

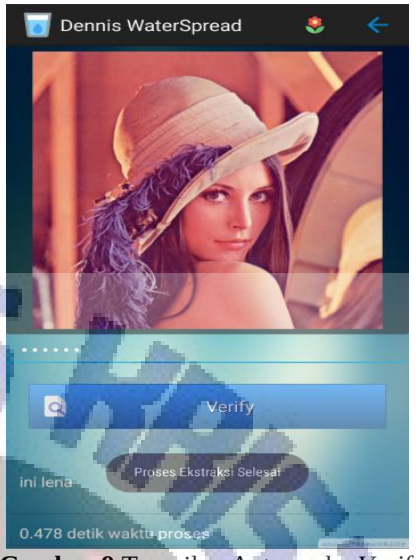


Gambar 7 Tampilan Aplikasi *Watermarking*

Gambar 7 menunjukkan tampilan awal dalam program *watermarking*, dalam program tersebut memiliki 3 menu pilihan yang dapat digunakan, untuk menjalankan proses *watermarking*, *user* dapat memilih menu *watermark* dan setelah *user* memilih maka akan muncul tampilan seperti Gambar 8 dan 9.



Gambar 8 Tampilan Antarmuka *Watermark*



Gambar 9 Tampilan Antarmuka *Verify*

Gambar 8 menampilkan *form* yang digunakan untuk melakukan *embedding* gambar. Pada *form* ini, disediakan fasilitas untuk memilih *cover image* dan mengirim gambar hasil *embedding. File* gambar yang digunakan untuk *cover image* adalah gambar bertipe PNG. Gambar yang telah di *embedding* akan ditampilkan bersebelahan dengan *cover image*. Gambar 9 menampilkan *form* yang digunakan untuk melakukan ekstraksi gambar. Pada *form* ini, disediakan fasilitas untuk memilih *cover image* yang sudah di-*embedding*. Untuk melakukan proses *verify*, *user* harus memasukkan *password* yang didapatkan pada saat proses *watermark*, maka pesan yang telah disisipkan pada *file* gambar akan ditampilkan.

Kode Program 1 Perintah untuk Menyisipkan *Bit* Informasi ke Dalam Citra Digital

```
1.  int color = raster.getPixel(x, y);
2.  int R = Color.red(color);
3.  int G = Color.green(color);
4.  int B = Color.blue(color);
5.
6.  if (indexChar < chars.length) {
7.          R = LSBStego.replaceLSB((byte) R, chars[indexChar]);
8.          indexChar++;
9.  }
10.
11. if (indexChar < chars.length) {
12.         G = LSBStego.replaceLSB((byte) G, chars[indexChar]);
13.         indexChar++;
14. }
15.
16. if (indexChar < chars.length) {
17.         B = LSBStego.replaceLSB((byte) B, chars[indexChar]);
18.         indexChar++;
19. }
```

Kode Program 1 merupakan perintah untuk menyisipkan informasi ke dalam citra digital. Pada piksel citra digital, diperoleh tiga komponen warna, *red*, *green*, dan *blue* (perintah pada baris 1-4). Kemudian pada tiap komponen warna disisipkan *bit* pesan (perintah baris 6-9).

Kode Program2 Perintah untuk Ektraksi *Bit* Informasi dari Dalam Citra Digital

```
1.  byte[] media = BitmapCrypto.extractByte(b);
2.  int lengthBit = 8 * 4;
```

```
3.  int i = 0;
4.
5.  char[] bufferLength = new char[lengthBit];
6.  for (; i < lengthBit; i++) {
7.          bufferLength[i] = retrieveLSB(media[i]);
8.  }
9.  byte[] length = fromBinary(new String(bufferLength));
10. int panjangPesan = 8 * byteArrayToInt(length);
11.
12. char[] buffer = new char[panjangPesan];
13. for (int j = 0; j < panjangPesan; j++) {
14.         buffer[j] = retrieveLSB(media[i++]);
15. }
16. return fromBinary(new String(buffer));
```

Kode Program 2 merupakan perintah untuk membaca informasi yang disisipkan pada citra digital. Proses awal adalah membaca elemen warna pada citra digital (perintah pada baris 1). Kemudian membaca panjang pesan yang disisipkan (perintah pada baris 5-9). Berdasarkan angka panjang pesan yang diperoleh, dibaca *bit-bit* pesan *watermark* (perintah pada baris 12-16).

Pengujian terhadap sistem *watermarking* yang dibangun, dilakukan dengan tujuan untuk melihat apakah sistem telah memnuhi konsep *watermarking* teknik *Spread Spectrum*, dan algoritma penyandian LCG. Pengujian yang dilakukan dijelaskan sebagai berikut.

Pengujian dilakukan pada *smartphone* dengan spesifikasi sebagai berikut :

Sistem Operasi : Android OS 4.1 (jelly bean)
CPU : Dual *Core* 1.2 GHz, Cortex A9
GPU : Power VR SGX531U
RAM : 1 GB
Internal : 4 GB

Pengujian pengaruh ukuran teks terhadap waktu proses pada data berulang (Pengujian 1), dilakukan dengan tujuan untuk mengetahui pengaruh ukuran teks terhadap waktu proses, pengujian 1 dilakukan dengan menggunakan gambar yang sama yaitu *file* gambar lena.png dengan ukuran *file* 462 KB, dimensi yang sama yaitu 512 x 512, teks yang sama tetapi panjangya berbeda yaitu teks yang disisipkan adalah "ini lena" yang dilakukan sehingga membentuk *string* dengan panjang karakter 128 karakter, 256 karakter, 512 karakter dan 1024 karakter, dimana 1 kararkter berukuran 1 *byte*, proses *spreading* pesan biner yang sama yaitu sebanyak 4 kali. Hasil pengujian 1 dapat dilihat pada Tabel 1.

Tabel 1 Hasil Rata-Rata Pengujian Pengaruh Ukuran Teks Terhadap Waktu Proses Pada Data Berulang

No	Gambar	Dimensi	Teks (pesan)	*Embeding* (detik)	*Extracting* (detik)
1.	Lena.png	512 x 512	128 *bytes*	0,06	0,04
2.	Lena.png	512 x 512	256 *bytes*	0,14	0,11
3.	Lena.png	512 x 512	512 *bytes*	0,29	0.27
4.	Lena.png	512 x 512	1024 *bytes*	0,60	0.58

Kesimpulan yang dapat diambil dari *pengujian 1* adalah ukuran teks yang disisipkan mempengaruhi waktu penyisipan. Semakin panjang teks, semakin besar waktu yang dibutuhkan oleh program.

Pengujian Pengaruh Dimensi Gambar Terhadap Waktu Proses (*Pengujian 2*), dilakukan dengan tujuan untuk mengetahui pengaruh dimensi

gambar terhadap waktu proses. *Pengujian 2* dilakukan dengan menggunakan gambar yang sama dan dimensi yang berbeda (otomatis *size* berbeda), dan teks yang sama dengan panjang 1024 karakter (1024 *byte*) diambil dari buku *Alice in Wonderland* [11]. Semua gambar yang memiliki ukuran panjang lebih dari 768 piksel akan diubah ukuran dimensinya secara otomatis menjadi 768 piksel (lebar sama). Berdasarkan hal tersebut maka dimensi maksimal gambar yang digunakan adalah 768 x 768 piksel, proses *spreading* pesan biner yang sama yaitu sebanyak 4 kali. Hasil *pengujian 2* dapat dilihat pada Tabel 2.

Tabel 2 Hasil Rata-Rata Pengujian Pengaruh Dimensi Gambar Terhadap Waktu Proses.

No	Gambar	Ukuran *File*	Dimensi	Teks	*Embedding* (detik)	*Extracting* (waktu)
1.	Lenna (1).png	204 KB	336 x 336	1024 *bytes*	0,23	0,20
2.	Lenna (2).png	346 KB	480 x 480	1024 *bytes*	0,24	0,22
3.	Lenna (3).png	484 KB	600 x 600	1024 *bytes*	0,25	0,23
4.	Lenna (4).png	695KB	768 x 768	1024 *bytes*	0,27	0,25

Kesimpulan dari hasil *pengujian 2* adalah dimensi gambar mempengaruhi waktu proses. Semakin kecil ukuran dimensi gambar, semakin cepat waktu proses.

Pengujian Pengaruh Susunan Warna Gambar Terhadap Waktu Proses (Pengujian 3), dilakukan dengan tujuan untuk mengetahui pengaruh susunan warna gambar terhadap waktu proses. *Pengujian 3* dilakukan dengan menggunakan empat gambar yang berbeda (otomatis *size* berbeda), dengan dimensi yang sama yaitu 800 x 600 piksel, dan teks yang sama dengan panjang 1024 karakter (1024 *byte*) diambil dari buku *Alice in Wonderland* [11], proses *spreading* pesan biner yang sama yaitu sebanyak 4 kali. Hasil *pengujian 3* dapat dilihat pada Tabel 3.

Tabel 3 Hasil Rata-Rata Pengujian Pengaruh Susunan Warna Gambar Terhadap Waktu Proses

Gambar	Ukuran *File*	Dimensi	Teks	*Embedding* (detik)	*Extracting* (detik)
sky.png	360 KB	800 x 600	1024 *bytes*	0,15	0,13
chrysant.png	471 KB	800 x 600	1024 *bytes*	0,19	0,17
tulips.png	575 KB	800 x 600	1024 *bytes*	0,22	0,20
bendera.png	4.29 KB	800 x 600	1024 *bytes*	0,23	0.21

Hasil dari *pengujian 3* adalah berbedanya *file* gambar sekalipun memiliki dimensi yang sama, memerlukan waktu proses yang berbeda. Hal ini dipengaruhi oleh ukuran *file* gambar. Semakin kecil ukuran gambar, semakin cepat waktu yang diperlukan untuk melakukan proses penyisipan.

Pengujian Integritas Pesan (Pengujian 4), dilakukan dengan cara menghitung nilai *hash/checksum* dari pesan sebelum disisipkan dengan pesan setelah proses ekstraksi. Pesan dinyatakan utuh jika memiliki nilai yang sama.

Perhitungan *hash* dilakukan dengan menggunakan algoritma MD5 [12]. Hasil *pengujian 4* dapat dilihat pada Tabel 4.

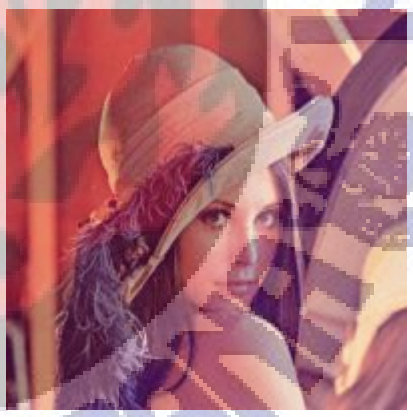
Tabel 4 Hasil Pengujian Integritas Pesan

Panjang Pesan	*Hash* Awal	*Hash* Akhir	Kesimpulan
32 *byte*	e4ffd2cbeffdd293 c1a0fadd9b57820c	e4ffd2cbeffdd293 c1a0fadd9b57820c	File utuh
128 *byte*	ba43e33d2d2c561f cc22e0792a363255	ba43e33d2d2c561f cc22e0792a363255	File utuh
256 *byte*	ff0e05140db8071f 39ae641cce9385d7	ff0e05140db8071f 39ae641cce9385d7	File utuh
512 *byte*	fdbdf9bb183f806d 30135a02804cc2f0	fdbdf9bb183f806d 30135a02804cc2f0	File utuh
1024 *byte*	6a5cbca9a0026b13 c3c8385d5ea1d0ed	6a5cbca9a0026b13 c3c8385d5ea1d0ed	File utuh

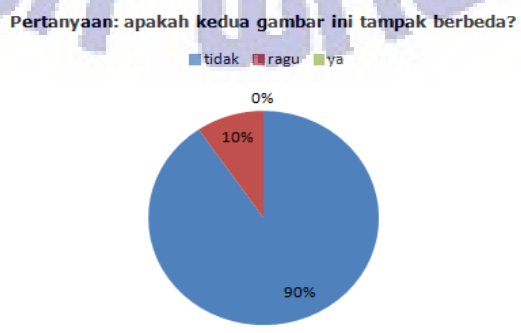
Berdasarkan hasil *pengujian 4* pada Tabel 4, dapat disimpulkan bahwa aplikasi *watermarking* yang dibangun, berhasil menyisipkan pesan dan mengekstraksi pesan tanpa menyebabkan perubahan/kerusakan pada pesan.

Pengujian Perbandingan Visual Gambar Sebelum dan Sesudah Proses Watermarking (Pengujian 5), dilakukan dengan cara menunjukkan *file* gambar (*cover*) sebelum dan sesudah proses *watermarking* kepada 30 responden. Kepada responden ditanyakan apakah kedua *file* tampak berbeda. *File* yang diujikan dapat dilihat pada Gambar 10 dan Gambar 11. Hasil kuesioner dapat dilihat pada Gambar 12.

Gambar 10 Gambar Sebelum *Watermarking*



Gambar 11 Gambar Sesudah *Watermarking*



Gambar 12 Grafik Pengujian Perbandingan Visual

Gambar 12 menunjukkan grafik *pengujian 5*, disimpulkan bahwa gambar tidak mengalami perubahan signifikan secara visual setelah proses *watermarking*.

Pengujian Pengaruh Perbedaan Panjang Password Dengan Konstanta Spread Terhadap Waktu Proses (Pengujian 6), dilakukan dengan tujuan untuk mengetahui pengaruh perbedaan panjang *password* dengan konstanta *spread* terhadap waktu proses. *Pengujian 6* dilakukan dengan menggunakan gambar dan tipe gambar yang sama, dengan dimensi yang sama yaitu 768 x 768 piksel, dan teks yang sama dengan panjang 1024 karakter (1024 *byte*) diambil dari buku *Alice in Wonderland* [11], proses *spreading* pesan biner yang berbeda-beda. Hasil *pengujian 6* dapat dilihat pada Tabel 5.

Tabel 5 Hasil Rata-Rata Pengujian Pengaruh Perbedaan Panjang *Password* Dengan Konstanta *Spread* Terhadap Waktu Proses

Gambar	Ukuran *File*	Panjang *Password*	Konstanta *Spread*	*Embedding* (detik)	*Extracting* (detik)
Lenna(1).png	695 KB	19 karakter	10 kali	0,06	0,04
Lenna(2).png	695 KB	7 karakter	15 kali	0,07	0,05
Lenna(3).png	695 KB	9 karakter	20 kali	0,09	0,07
Lenna(4).png	695 KB	4 karakter	25 kali	0,1	0,09

Kesimpulan dari hasil *pengujian 6* adalah, berbedanya panjang *password* tidak berpengaruh terhadap proses waktu, tetapi berbedanya konstanta *spread* memerlukan waktu proses yang berbeda, semakin besar konstanta *spread*, semakin lama waktu yang diperlukan untuk melakukan proses penyisipan.

Pengujian pengaruh image pada aplikasi mobile (pengujian 7), dilakukan dengan tujuam untuk mengetahui hasil pesan yang telah di kirim dan di teriama melalui media social pada aplikasi mobile. *Pengujian 7* dilakukan dengan menggunakan gambar dan tipe gambar yang sama, dengan dimensi yang sama yaitu 200 x 200 piksel, dena teks yang sama dengan panjang 1024 karakter (1024 *byte*). Hasil *pengujian 7* dapat di lihat pada Tabel 6.

Tabel 6 Hasil Pengujian Pengaruh Image Pada Aplikasi Mobile

Gambar	Ukuran *File*	Media	Kirim	Terima	Keterangan
Lena.png	695 KB	BBM	Berhasil	Berhasil	HD
Lena.png	695 KB	Line	Berhasil	Berhasil	HD
Lena.png	695 KB	WA	Berhasil	Berhasil	HD
Lena.png	695 KB	Email	Berhasil	Berhasil	HD

Kesimpulan dari hasil *pengujian 7* adalah, berbedanya media social tidak berpengaruh terhadap proses pengiriman pada gambar dan tidak mempengaruhi gambar yang telah dikirim maupun diterima.

5. Simpulan

Berdasarkan penelitian yang sudah dilakukan dapat disimpulkan: 1) sistem *watermarking* yang dibangun menggunakan metode *spread spectrum*, dapat melakukan penyisipan (*embedding*)/pengambilan (*extracting*) data pada *byte* komponen warna, pada *filecover* berformat PNG; 2) ukuran teks yang disisipkan mempengaruhi waktu proses, isi teks tidak menunjukkan pengaruh yang signifikan terhadap waktu proses; 3) perbedaan *file* gambar mempengaruhi kecepatan proses, hal ini disebabkan tiap gambar memiliki susunan piksel warna yang berbeda sehingga memiliki ukuran *file* yang berbeda juga, ukuran *file* inilah yang mempengaruhi kecepatan proses penyisipan; 4) aplikasi *watermarking* yang dibangun, berhasil menyisipkan pesan dan mengekstraksi pesan tanpa menyebabkan perubahan/kerusakan pada pesan; 5) gambar tidak mengalami perubahan signifikan secara visual setelah proses *watermarking*; 6) berbedanya tipe gambar sekalipun memiliki dimensi yang sama, memerlukan waktu proses yang berbeda, hal ini dipengaruhi oleh ukuran *file* gambar; 7) berbedanya panjang *password* tidak berpengaruh terhadap proses waktu, tetapi berbedanya konstanta *spread* memerlukan waktu proses yang berbeda, semakin besar konstanta *spread*, semakin lama waktu yang diperlukan untuk melakukan proses penyisipan. Saran untuk pengembangan aplikasi ke depan adalah: 1) data yang disisipkan dapat dikembangkan tidak hanya data teks, namun juga data gambar atau *audio*; 2) *file* gambar *watermark* tahan terhadap serangan manipulasi data (*cropping*, *compression*, *rotating*); 3) *file watermark* dapat menggunakan *file audio* maupun video, ataupun *file* gambar dengan ukuran yang lebih besar; 4) *file* yang di *watermark* tidak hanya data image saja, namun juga data mobile lainnya.

6. Daftar Pustaka

[1] Septianingsih, Rina, 2009, Implementasi Watermarking Pada Citra Digital Menggunakan Metode LSB, Jurnal, Jakarta: Program Studi Teknik Informatika Universitas Gunadarma.

[2] Ischam, Ali, 2009, *Watermarking* Citra Digital Menggunakan Transformasi *Hybrid* DWT Dan DCT, Jurnal, Semarang: Program Studi Teknik Informatika Universitas Diponegoro Semarang.

[3] Tullah, Rachmat, 2014, Perancangan Steganografi Dengan Media Gambar Pada Aplikasi Berbasis Android, Jurnal, Jakarta,: Program Studi Teknik Informatika STMIK Bina Sarana Global.

[4] Setiawan, I Made Robi Budi, 2012, Perancangan Aplikasi *Watermarking* Pada Media Fotografi Sebagai Perlindungan Hak Cipta Menggunakan Metode *Spread Spectrum*, Skripsi, Salatiga: Program Studi Teknik Informatika Universitas Kristen Satya Wacana.

[5] Sridevi, Damodaram, & Narasimham, 2009, *Efficient Method of Audio Steganography by Modified LSB Algorithm and Strong Encryption Key with Enhanced Security*, Jurnal, Hyderabad : Department of Computer Science and Engineering-JNTUH.

[6] Lesley, Mitchell, and Talal G. Shamoon, 2004, *Robustness And Security Of Digital Watermark,* STAR Lab InterTrust Technologies Corporation USA.
[7] Putranto, Adam, 2009, Steganografi Melalui Media Gambar Dengan Metode *Spread Spectrum*, Skripsi, Salatiga: Program Studi Teknik Informatika Universitas Kristen Satya Wacana.
[8] Flikkema, Paul G., 1997, *Spread Spectrum Techniques For Wireless Communications,* IEEE Signal Processing Magazine.
[9] Tullah, Agusli, Iqbal dan Halim, 2014, Perancangan Steganografi Dengan Media Gambar Pada Aplikasi Berbasis Android, Jurnal, Jakarta: STMIK Bina Sarana.
[10] Utomo, Tri Prasetyo, 2012, Steganografi Gambar Dengan Metode *Least Significant Bit* Untuk Proteksi Komunikasi Pada Media *Online*, Jurnal, Bandung: Jurusan Teknik Informatika UIN Sunan Gunung Djati.
[11] Hasibuan, Zainal, A., 2007, *Metodologi Penelitian Pada Bidang Ilmu Komputer Dan Teknologi Informasi : Konsep, Teknik, dan Aplikasi*, Jakarta: Program Studi Ilmu Komputer Universitas Indonesia.
[12] Carroll, Lewis, 2008, *Alice's Adventures in Wonderland*.
[13] Rivest, R., 1992, *The MD5 Message-Digest Algorithm*.