RANCANG BANGUN APLIKASI *SINGLE SIGN-ON SERVER* MENGGUNAKAN AUTENTIKASI GAMBAR

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Pada
Jurusan Teknik Informatika

Oleh :

GUNTORO
10651004298

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU
PEKANBARU
2011

RANCANG BANGUN APLIKASI *SINGLE SIGN ON SERVER* MENGGUNAKAN AUTENTIKASI GAMBAR

GUNTORO
10651004298

Tanggal Sidang : 19 Mei 2011
Periode Wisuda : Juli 2011

Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Pada tugas akhir ini dikembangkan sebuah aplikasi sistem *single sign-on* dengan menerapkan autentikasi *login* menggunakan gambar. Sistem *single sign-on* merupakan sebuah teknologi yang mengizinkan pengguna jaringan agar dapat mengakses sumber daya dalam jaringan hanya dengan menggunakan satu akun pengguna saja. Dengan menggunakan *single sign-on*, seorang pengguna hanya cukup melakukan proses autentikasi sekali saja untuk mendapatkan izin akses terhadap semua layanan yang terdapat di dalam jaringan. Autentikasi *login* berbasis teks pada sistem *single sign-on* yang sudah ada saat ini, mempunyai kelemahan, salah satunya adalah pencurian *password* dengan aplikasi *keylogger*.

Aplikasi Sistem *single sign-on* yang dikembangkan dengan menerapkan autentikasi menggunakan gambar. Gambar yang digunakan telah diberikan sebuah keamanan yaitu menggunakan teknik steganografi dengan metode *Least Significant Bit.*

Sistem *single sign-on* yang dibangun menggunakan pemrograman PHP. Berdasarkan hasil pengujian sistem *single sign-on* yang dilakukan dapat dilihat bahwa sistem dapat berjalan dengan baik dan autentikasi gambar dengan menerapkan metode *least significant bit* dapat diimplementasikan pada sistem *login single sign-on.*

Kata Kunci : Autentikasi, *Least Significant Bit* , *Single Sign-On*, Steganografi

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel **Halaman**

BAB I
PENDAHULUAN

1.1 Latar Belakang

Otentikasi merupakan suatu proses untuk menentukan apakah seseorang berhak mengakses suatu aplikasi *web* atau tidak. Cara yang paling sederhana adalah dengan menggunakan proses *login*, seseorang memasukkan *username* dan *password* (*credential*) kemudian diotentikasi apakah *credential* tersebut *valid* atau tidak, jika *valid* maka seseorang tersebut boleh mengakses, jika tidak maka dia tidak boleh mengakses. Sebagian besar aplikasi *web* saat ini menggunakan cara tersebut, dengan berbagai tambahan keamanan.

Menjadi suatu masalah ketika seorang pengguna memiliki banyak aplikasi *web* yang membutuhkan otentikasi. Dia harus menghafal banyak *credential,* walaupun banyak orang membuat *credential* yang sama untuk berbagai aplikasi *web.* Terdapat masalah lagi jika pengguna membuat satu *credential* untuk berbagai aplikasi *web*, karena pengguna harus memasukkan *credential* berulang kali. Misalkan penguna akan menggunakan layanan *email* maka dia harus memasukkan *credential*, jika pengguna akan menggunakan layanan *forum* maka dia harus memasukkan *credential* dan begitu seterusnya. Oleh karena itu dibutuhkan suatu sistem yang dapat mengintegrasikan seluruh layanan aplikasi dan mengelola proses autentikasi masing-masing sistem layanan, menjadi proses autentikasi. Proses autentikasi pada sistem yang terintegrasi ini memerlukan sebuah sistem tambahan yang menjadi penghubung antara sistem *integrator* dengan sistem layanan aplikasi. Sistem inilah yang dapat menangani seluruh autentikasi setiap aplikasi sistem, sistem ini dikenal dengan Sistem *Single Sign-On*

Sistem *Single Sign-On* merupakan sebuah teknologi yang mengizinkan pengguna jaringan agar dapat mengakses sumber daya dalam jaringan hanya dengan menggunakan satu akun pengguna saja. Dengan menggunakan *Single Sign-On*, seorang pengguna hanya cukup melakukan proses autentikasi sekali saja untuk mendapatkan izin akses terhadap semua layanan yang terdapat di dalam

jaringan. Autentikasi *login* berbasis teks pada sistem *single sign-on* yang sudah ada saat ini, mempunyai kelemahan, salah satunya adalah pencurian *password* dengan aplikasi *keylogger*. Oleh karena itu untuk meminimalisir kelemahan tersebut, diterapkan autentikasi berbasis teks dan gambar, yang mana gambar tersebut sudah diberikan keamanan menggunakan steganografi berbasis *Least Significant Bit*.

Pada penelitian yang berjudul ”*Open Source in Web-based Applications: A Case Study on Single Sign-On”* oleh Agostino dkk (2009), penelitian tersebut mengevaluasi sistem *single sign-on* berbasis *open source* menggunakan aplikasi CAS (*Central Authentication Service*) yang dikembangkan oleh *Yale University*, *SourceID* dan JOSSO (*Java Open Single Sign-On*) serta melakukan perbandingan terhadap ketiga aplikasi *single sign-on* tersebut. Pada penelitian yang berjudul *A Taxonomy of Single Sign-On Systems* oleh Pashalidis dkk (2003), membahas tentang pendekatan sistem *single sign-on* masa depan dalam konteks yang lebih terstruktur, skema sistem *single sign-on* serta beberapa perbedaan penting dalam hal keamanan sistem *single sign-on*. Pada Tugas Akhir yang berjudul ”Implementasi Sistem *Single Sign-On* Berbasis *Java*” oleh Nursyamsi (2009), penelitian tersebut membahas tentang implementasi sistem *single sign-on* berbasis *java* dengan menggunakan aplikasi *Java Open Single Sign-On* (JOSSO).

Sistem *single sign-on* dengan menerapkan autentikasi menggunakan gambar memiliki beberapa manfaat, salah satunya adalah dengan sistem *single sign-on*, seseorang pengguna *web portal* cukup melakukan sekali *login* untuk beberapa situs *web* dan autentikasi menggunakan gambar dapat memberikan jaminan *Non-repudiation* atau keaslian terhadap gambar yang digunakan. Dengan sistem ini dapat memberikan kemudahan, kenyamanan serta keamanan.

Dari latar belakang diatas maka tujuan pada Tugas Akhir ini adalah bagaimana membangun serta menguji sistem *single sign-on* dengan menerapkan autentikasi menggunakan gambar.

1.2 Rumusan Masalah

Berdasarkan identifikasi masalah diatas, maka rumusan masalah yang akan disajikan dalam penulisan penelitian ini yaitu bagaimana menganalisa, merancang, membangun serta menguji sistem *Single Sign-On* dengan menerapkan sistem autentikasi menggunakan gambar.

1.3 Batasan Masalah

Dalam pembuatan Tugas Akhir ini mempunyai beberapa batasan masalah diantaranya adalah :

1. Penelitian sistem *single sign-on* dititikberatkan pada aplikasi berbasis *web*.
2. Kategori Sistem *Single Sign-On* yang dibangun berbasis Otorisasi (*Authorization*)
3. *File* gambar yang akan digunakan untuk autentikasi telah di sisipi pesan dengan metode steganografi *Least Significant Bit* (LSB).
4. Tidak membahas ukuran *pixel* yang digunakan untuk penyisipan kedalam gambar.
5. Metode pengamanan menggunakan *file* gambar hanya digunakan pada aspek *Nonrepudiation*.

1.4 Tujuan

Tujuan yang ingin dicapai pada Tugas Akhir ini adalah Membangun serta menguji sistem *Single Sign-On* dengan menerapkan sistem autentikasi menggunakan gambar.

1.5 Sistematika Penulisan

Sistematika penulisan tugas akhir ini dibagi menjadi 6 (enam) bab. Setiap bab terdiri dari sub bab dan penjelasan yang tersusun sehingga mudah untuk dipahami. Berikut penjelasan tentang masing-masing bab:

BAB I Pendahuluan

Merupakan deskripsi umum dari tugas akhir ini, yang meliputi: latar belakang masalah, rumusan masalah, batasan masalah, tujuan penyusunan tugas akhir serta sistematika penulisan tugas akhir.

BAB II Landasan Teori

Pada bab ini menjelaskan tentang dasar-dasar yang digunakan untuk penelitian tugas akhir ini antara lain konsep *Single Sign On*, mekanisme kerja protokol TCP/IP, metode identifikasi dan otentikasi, steganografi, serta *web services*.

BAB III Metodologi Penelitian

Dalam bab ini menjelaskan mengenai cara yang dilakukan dalam menyelesaikan persoalan yang menjadi objek penelitian.

BAB IV Analisa dan Perancangan

Berisi pembahasan mengenai deskripsi kebutuhan sistem dan perancangan komponen.

BAB V Implementasi dan Pengujian

Berisi pembahasan mengenai lingkungan pengembangan sebuah aplikasi dan hasil pengujian aplikasi.

BAB VI Penutup

Dalam bab ini akan dijelaskan beberapa kesimpulan yang didapatkan dari Tugas Akhir serta saran untuk penelitian selanjutnya.

BAB II
LANDASAN TEORI

2.1 *Single Sign-On* (SSO)

Teknologi *Single Sign-On* (sering disingkat menjadi SSO) adalah teknologi yang mengizinkan pengguna jaringan agar dapat mengakses sumber daya dalam jaringan hanya dengan menggunakan satu akun pengguna saja. Teknologi ini sangat diminati, khususnya dalam jaringan yang sangat besar dan bersifat heterogen (di saat sistem operasi serta aplikasi yang digunakan oleh komputer adalah berasal dari banyak *vendor*, dan pengguna dimintai untuk mengisi informasi dirinya ke dalam setiap *platform* yang berbeda tersebut yang hendak diakses oleh pengguna). Dengan menggunakan SSO, seorang pengguna hanya cukup melakukan proses autentikasi sekali saja untuk mendapatkan izin akses terhadap semua layanan yang terdapat di dalam jaringan.(Jani Hursti, 1997)
Keuntungan sebuah sistem menggunakan SSO adalah sebagai berikut :

1. Mengurangi tingkat kejenuhan *user* dalam penggunaan *password*.
2. Mengurangi waktu yang digunakan untuk memasukkan *password* kembali untuk sebuah identitas yang sama.
3. Dapat mendukung otentikasi konvensional seperti *Windows Credential.*
4. Mengurangi biaya IT seiring dengan berkurangnya *user* yang meminta bantuan mengenai hal otentikasi yang dalam hal ini adalah permasalahan di *username* atau *password.*
5. Keamanan di semua *level* akses baik masuk maupun keluar sistem.

Beberapa kategori sistem *single sign-on* yaitu :

1. Autentikasi (*Authentication*)

Sistem *single sign-on* berbasis autentikasi yang mana SSO *server* hanya memberikan *service* apakah *user* A telah ter-autentikasi atau belum, SSO *server* tidak melakukan proses otorisasi atas *user* yang sedang aktif tersebut. Proses otorisasi sendiri dilakukan pada setiap aplikasi. Beberapa contoh yang

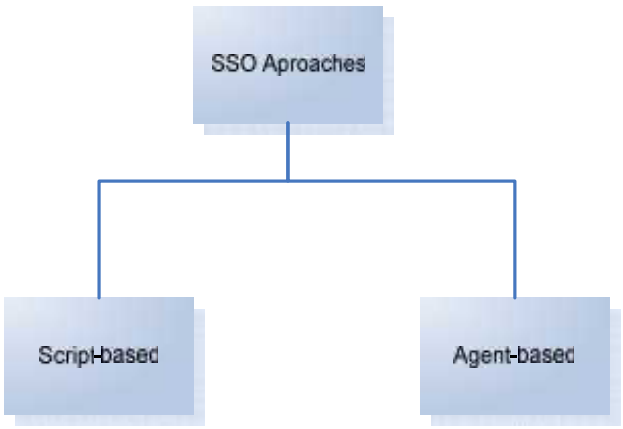
menggunakan proses autentikasi adalah *OpenID* dan *fbconnect. OpenID* merupakan salah satu yang menggambarkan SSO autentikasi ini. Ketika seseorang akan *comment* pada suatu *blog*, kita bisa menggunakan fasilitas *OpenID*. Dimana *OpenID* hanya memberitahu *wordpress* bahwa *user* telah terautentikasi dan *wordpress* sendiri bisa mendapatkan data *user*, seperti nama maupun *e-mail*. Sama halnya dengan *fbconnect*, ketika *user* telah ter-autentikasi pada *facebook*, maka *facebook* tidak melakukan otorisasi pada aplikasi *client*. Jadi hanya sebatas autentikasi.

2. Otoriasi (*Authorization*)

Tugas SSO *server* untuk SSO-otorisasi memiliki tugas sedikit lebih berat, karena setelah memastikan *user* telah ter-autentikasi, SSO *server* masih harus menghandle otorisasi *user* tersebut.

Sistem *single sign-on* yang dibangun pada tugas akhir ini adalah berbasis otorisasi (*Authorization*).

2.1.1 Arsitektur Sistem SSO



Gambar 2.1 Pendekatan sistem SSO

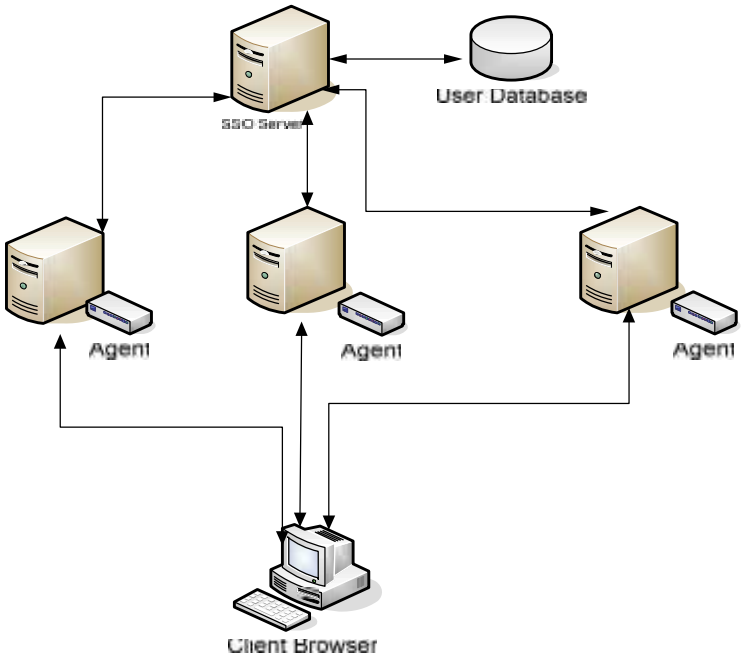
(Sumber : Springer-Verlag Berlin Heidelberg, 2003)

Solusi sistem SSO didasarkan pada salah satu dari dua tingkat pendekatan-pendekatan *script* dan pendekatan *agent*. Pendekatan agent lebih digunakan dalam Tugas Akhir ini karena dianggap lebih cocok untuk aplikasi berbasis *web* atau

sevice provider (SP). Gambar 2.1 menunjukkan pembagian dari pendekatan sistem SSO.

Agent merupakan sebuah program kecil yang berjalan pada tiap-tiap *web server*. *Agent* ini membantu mengkoordinir aliran kerja dari SSO dalam hal otentikasi pengguna dan penanganan sesi. Solusi dari arsitektur sistem SSO ditunjukkan oleh Gambar 2.2. Arsitektur SSO memiliki dua bagian utama; *agent* yang berada di *web server* atau SP dan sebuah *server* SSO yang berdedikasi yang mana akan dijelaskan berikut ini :

1. ***Agent*** : Sebuah *agent* menterjemahkan setiap permintaan HTTP yang masuk ke *web server*. Hanya ada satu *agent* di tiap-tiap *web server*, yang mana *host* bagi aplikasi/SP. *Agent* tersebut akan berinteraksi dengan *browser* klien pada sisi pengguna, dan dengan *server* SSO pada sisi SP.
2. **SSO *server*** : *server* SSO menggunakan *cookies temporer* (sementara) untuk menyediakan fungsi manajemen sesi. Sebuah *cookies* terdiri dari informasi seperti *user-id, session-id*, *session creation time, session expiration time* dan lain-lain.



Gambar 2.2 Arsitektur Sistem SSO

(Sumber : Springer-Verlag Berlin Heidelberg, 2003)

2.1.2 Persyaratan Sistem *Single Sign-On*

Beberapa persyaratan dalam membangun sebuah sistem *Single Sign-On* diantaranya (Ardagna, dkk 2009) yaitu :

1. ***Authentication***

   Fitur utama dari sistem SSO adalah memberikan suatu mekanisme otentikasi. Biasanya menggunakan otentikasi *username* dan *password.*

2. ***Strong Authentication***

   Untuk keamanan yang lebih tinggi, otentikasi menggunakan *username* dan *password* sederhana tidaklah cukup. Solusi utama yaitu menggunakan mekanisme otentikasi berdasarkan biometrik seperti (sidik jari, *retina scan* dan lain sebagainya).

3. ***Authorization***

   Setelah sistem otentikasi sudah dapat digunakan. Maka sistem dapat memberikan suatu otorisasi.

4. ***Provisioning***

   Ketentuan merupakan kondisi yang dibutuhkan sebelum keputusan diambil (Bettini, Jajodia, Sean Wang, & Wijesekera, 2002). Jadi tanggung jawab *user* memastikan bahwa permintaan dikirim dengan memuaskan.

5. ***Federation***

   Federation berkaitan erat dengan tingkat kepercayaan.

6. **C.I.M (*Centralized Identity Management*)**

7. ***Client Status Info***

   Sistem arsitektur SSO berarti pertukaran informasi pengguna antara *server* SSO dan layanan untuk memenuhi otentikasi serta otorisasi.

8. ***Single Point of Control***

   Tujuan dari implementasi sistem SSO yaitu menyediakan pengontrolan jalur akses yang unik bagi para pengguna.

9. ***Standard Compliance***

   Protocol – protocol yang digunakan dalam sistem SSO misalnya teknologi X.509 (*Public-Key* Infrastruktur), untuk *security* menggunakan SAML serta

protocol untuk bertukar informasi pada lingkungan yang berbeda seperti SOAP.

10. Cross-Language Availability

Teknologi yang digunakan untuk mengembangkan sebuah aplikasi, misalnya penerapan protokol berbasis XML.

11. Password Proliferation Prevention

Tingkat keamanan *password*.

2.1.3 Produk-Produk SSO

Produk-produk SSO yang berbasis *open sour*ce yang umum digunakan saat ini adalah sebagai berikut :

1. CAS (*Central Authentication Service*)

CAS adalah sebuah *framework* untuk *Single Sign-On* yang dibuat dengan menggunakan bahasa Java. Konsep kerja CAS sebernarnya menggunakan *Ticket Granting* dimana ketika pengguna melakukan *login*, maka pengguna akan diberikan *ticket* (dalam hal ini tersimpan dalam *cookies)* yang nantinya akan digunakan untuk melakukan itentikasi pada setiap aplikasi *web.* Jadi di setiap aplikasi *web* akan dilakukan pengecekan dari *ticket* yang telah diberikan oleh CAS tersebut.

2. OpenSSO (*Open Single Sign-On*)

OpenSSO adalah sebuah infrastruktur yang mendukung layanan berbasis identitas, dan implementasi solusi dari *Single Sign-On* (SSO) transparan serbagai komponen keamanan dalam infrastruktur jaringan. OpenSSO berbasisi pada solusi *Identity Management* yang dikembangkan oleh Sun.

3. JOSSO (*Java Open Single Sign-On*)

JOSSO adalah sebuah infrastruktur SSO berbasis J2EE dan *opensource* yang bertujuan untuk menyediakan solusi bagi sentralisasi, *netral platform*, otorisasi dan otentikasi pengguna.

Tabel 2.1 merupakan perbandingan keunggulan dan kekurangan beberapa produk SSO tersebut :

Tabel 2.1 Perbandingan keunggulan beberapa produk SSO

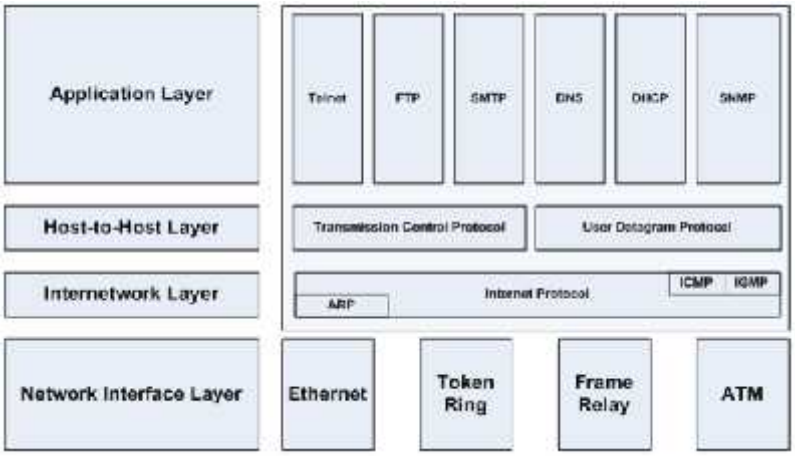
CAS	OpenSSO	JOSSO
1. Terdapat banyak *client libraries* untuk membantu mengintegrasikan CAS dengan aplikasi *server* 2. Tampilan pada halaman *login* dapat diganti 3. Dapat dikonfigurasi dengan mudah untuk mengaktifkan HTTP 4. Dilengkapi dengan *pligguble authenticators* untuk melakukan validasi terhadap LDAP, dll 5. Ukuran file 12 MB	1. *Agent* dapat di tempatkan ke berbagai aplikasi *server* seperti : Apache. Sun Java *System Web Server*, Microsoft IIS, Domino 2. Konfigurasi dapat dilakukan dengan menulis otentikasi modul 3. Keamanan layanan *web* menggunakan SAML (*Security Assertion Markup Language*) 4. Ukuran *file* 295 MB	1. Menyediakan sebuah infrastruktur *plugin* untuk berintegrasi dengan aplikasi *server* yang lain 2. Menggunakan layanan *web* untuk menyatakan identitas pengguna melalui protokol SOAP (*Simple Object Access Protokol*) 3. Dapat berintegrasi dengan aplikasi non Java (PHP,.Net, dll) 4. Ukuran *file* 45 MB

Tabel 2.2 Perbandingan kekurangan beberapa produk SSO

CAS	OpenSSO	JOSSO
Implementasi dasar hanya meliputi HTTPS	Tidak mendukung otentikasi *proxy* tetapi bisa dibuat sendiri dengan menggunakan otentikasi API	Tidak mendukung untuk beberapa aplikasi *server* tertentu

2.2 Mekanisme Kerja Protokol TCP/IP

Transfer Control Protocol/Internet Protocol (TCP/IP) pada dasarnya terdiri dari beberapa protokol yang berbeda, masing-masing dirancang untuk memenuhi tugas-tugas khusus dalam jaringan yang menggunakan TCP/IP. Berkat perinsip ini, tugas masing-masing protokol menjadi jelas dan sederhana. Protokol yang satu tidak perlu mengetahui cara kerja protokol yang lain, sepanjang masih bisa saling mengirim dan menerima data.



Gambar 2.3 Model Protokol TCP/IP

Oleh karena itu, TCP/IP menjadi protokol komunikasi data yang fleksibel. Protokol TCP/IP dapat diterapkan dengan mudah di setiap jenis komputer dan *interface* jaringan, karena sebagian besar isi kumpulan protokol ini tidak spesifik terhadap satu komputer atau jaringan tertentu. Agar TCP/IP dapat berjalan diatas *interface* jaringan tertentu, hanya perlu dilakukan perubahan pada protokol yang berhubungan dengan *interface* saja.

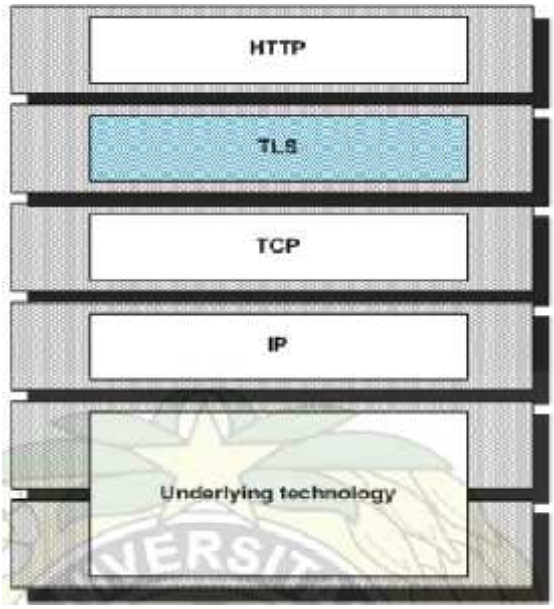
Model referensi TCP/IP pada gambar 2.3 menunjukkan empat lapisan dalam model TCP/IP yaitu : *Network Interface Layer, Internetwork Layer*, *Host-Host Layer/Transport La*yer, dan *Aplication Layer*. Untuk sistem *Single Sign On* (SSO) ini yang berhubungan dengan protokol otentikasi hanya terdapat pada lapisan/*layer Application, Transport* dan *Internet*. Pada setiap lapisan tersebut akan dilihat dari sisi keamanan/*security* ini dikarenakan sistem otentikasi erat dengan faktor keamanan.

2.2.1 *Security* Lapisan Aplikasi (*Application Layer Security)*

Implementasi keamanan pada lapisan aplikasi adalah yang paling mudah dan sederhana, jika komunikasi *internet* melibatkan dua pihak, seperti kasus komunikasi *e-mail* dan *telnet.* Pengirim dan penerima mempunyai kesepakatan untuk menggunakan protokol yang sama dan jenis layanan keamanan yang diinginkan. Pada lapisan ini terdapat 2 protokol yang digunakan, yaitu PGP (*Pretty Good Privacy*) dan SSH (*secure shell*).

2.2.2 *Security* Lapisan *Transport* (*Transport Layer Security*)

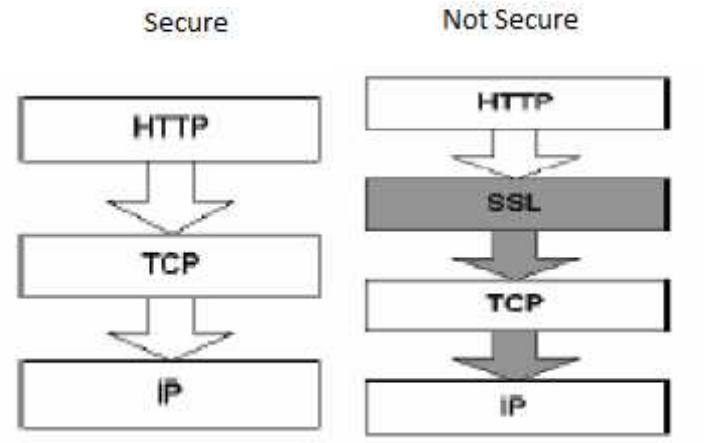
Transport Layer Security (TLS) berada antara lapisan aplikasi dan lapisan *transport*. Pada Gambar 2.4 diperlihatkan, bahwa TLS berada diantara lapisan protokol HTTP (aplikasi) dan protokol TCP (*transport*). Dengan demikian dapat dinyatakan bahwa lapisan aplikasi dalam hal ini HTTP menggunakan TLS dan TLS menggunakan layanan dari lapisan *transport* dalam hal ini TCP untuk membawa informasi.



Gambar 2.4 Posisi TLS pada *layer-layer* protokol IP

TLS dirancang untuk menyediakan keamanan pada lapisan *transport*. TLS diperoleh dari suatu protokol keamanan yang disebut dengan SSL (*Secure Socket*

Layer) yang dirancang oleh *Netscape* guna menjamin keamanan WWW. TLS adalah bentuk lain dari SSL, yang dirancang oleh IETF (*Internet Engineering Task Force*) untuk transaksi di *internet* seperti ditunjukkan pada gambar 2.5.



Gambar 2.5 SSL lapisan terpisah dalam susunan protokol *Internet*

2.2.3 *Security* Lapisan *Internetwork Layer* (*IPSec*)

IP *Security* (*IPSec*) merupakan sekumpulan protokol yang dirancang oleh IETF (*Internet Engineering Task Force*) untuk menyediakan layanan keamanan bagi paket data yang dibawa di *internet*. IPSec tidak didefinisikan untuk menggunakan berbagai metoda enkripsi atau otentikasi yang spesifik. Tetapi *IPSec* menyediakan sebuah *framework* dan sebuah mekanisme. *IPSec* mendefinisikan dua protokol yang digunakan pada IP (lapisan *network*) seperti : *Protokol Authentication Header* (AH) dan protokol *Encapsulating Security Payload* (ESP).

2.3 Aspek-Aspek Keamanan Komputer

Keamanan komputer meliputi beberapa aspek di antaranya (Dony Ariyus, 2006) :

1. *Authentication* : agar penerima informasi dapat memastikan keaslian pesan tersebut datang dari orang yang dimintai informasi.

2. *Integrity* : keaslian pesan yang dikirim melalui jaringan dan dapat dipastikan bahwa informasi yang dikirim tidak dimodifikasi oleh orang yang tidak berhak dalam perjalanan informsasi.
3. *Nonrepudiation* : merupakan hal yang bersangkutan dengan si pengirim. Si pengirim tidak dapat mengelak bahwa dialah yang mengirim informasi tersebut.
4. *Authority* : Informasi yang berada pada sistem jaringan tidak dapat dimodifikasi oleh pihak yang tidak berhak atas akses tersebut.

2.4 Metode – Metode Identifikasi dan Otentikasi

Elemen *interface* yang pertama kali ditemui kebanyakan subjek ketika mengakses sistem informasi adalah identifikasi dan otentikasi. Tahap identifikasi memperkenankan subjek mengklaim sebagai entitas tertentu dengan menunjukkan bukti-bukti identitas. Bukti-bukti tersebut dapat berupa ID pengguna atau nomor PIN, atau yang lebih kompleks seperti atribut fisik. Setelah subjek mengklaim sautu identitas, sistem memvalidasi apakah pengguna tersebut terdaftar dalam *database* pengguna dan membuktikan bahwa subjek tersebut adalah benar-benar sebagai entitas yang diklaimnya.

Tahap otentikasi meminta objek menunjukkan informasi tambahan yang sesuai dengan informasi tentang subjek tersebut yang telah disimpan. Dua tahap ini sering disebut dengan otentikasi dua faktor, yang memberikan proteksi terhadap subjek yang tidak memiliki otoritas untuk mengakses sistem. Setelah subjek di otentikasi, sistem kontrol akses mengevaluasi hak dan izin subjek untuk mengabulkan atau menolak permintaan terhadap objek. Tahap ini disebut dengan tahap otoritas.

Tabel 2.3 Tipe-Tipe Otentikasi

Authentication Type	*Description*	*Examples*
Type 1	*What you know*	*Password, passphrase, PIN, lock combination*
Type 2	*What you have*	*Smart card, token device*

Authentication Type	*Description*	*Examples*
Type 3	*What you are*	*Biometrics,-fingerprint, palm print, retina/iris pattern, voice pattern.*

Ada tiga kategori/tipe umum dari informasi otentikasi. Praktek pengamanan yang baik biasanya membuat tahap identifikasi dan otentikasinya memerlukan input setidaknya dari dua tipe berbeda. Tiga tipe umum data otentikasi dijelaskan pada Tabel 2.3.

Tipe otentikasi yang paling umum dan paling mudah untuk diimplementasikan adalah otentikasi tipe 1. Yang dilakukan adalah meminta subjek membuat *password, passphrase*, atau nomor PIN. Alternatif lain adalah menyediakannya untuk pengguna. Kesulitan dalam otentikasi tipe 1 adalah perlunya mendorong subjek untuk membuat *frase* yang sangat sulit diterka oleh orang lain, namun tidak terlalu rumit sehingga sulit untuk diingat. *Password* (*frase* atau PIN) yang sulit diingat akan mengurangi nilai dari *password* itu sendiri. Hal tersebut dapat terjadi bila *administrator* terlalu sering memerlukan penggantian *password* sehingga pengguna kesulitan untuk mengingat *password* terbaru. Jadi, yang disarankan adalah menjaga *password* secara rahasia dan aman. Aturan-aturan berikut ini adalah pentunjuk yang baik untuk membuat *password* yang aman :

1. *Password* setidaknya memiliki panjang 6 karakter.
2. *Password* setidaknya mengandung sebuah angka atau karakter tanda baca.
3. Tidak menggunakan kosakata atau kombinasi kosakata.
4. Tidak menggunakan data pribadi, seperti tanggal kelahiran, nama anggota keluarga atau binatang peliharaan, atau lagu atau hobi favorit.
5. Tidak sesekali menuliskan *password*.
6. Membuat *password* yang mudah diingat tapi suit diterka.

Data otentikasi tipe 2 lebih rumit untuk dilakukan karena subjek perlu membawa suatu alat atau sejenisnya. Alat tersebut umumnya perangkat elektronik

yang menghasilkan suatu nilai yang bersifat sensitif terhadap waktu atau suatu jawaban untuk di *input*. Meskipun otentikasi tipe 2 lebih rumit, tipe ini hampir selalu lebih aman dibandingkan dengan otentikasi tipe 1.

Otentikasi tipe 3, atau biometrik dalah yang paling canggih. Biometrik menggambarkan pendeteksian dan pengklasifikasian dari atribut fisik. Terdapat banyak teknik biometrik yang berbeda diantaranya :

1. Pembacaan sidik jari/telapak tangan.
2. Geometri tangan
3. Pembacaan retina/iris
4. Pengenalan suara
5. Dinamika tanda tangan

Karena kerumitannya, biometrik adalah tipe otentikasi yang paling mahal untuk diimplementasikan. Tipe ini juga lebih sulit untuk dipeliharan karena sifat ketidak-sempurnaan dari analisis *biometerik.* Dianjurkan untuk berhati-hati beberapa masalah-masalah utama dari *error-error biometrik*. Pertama, sistem mungkin menolak subjek yang memiliki otoritas. Ukuran kesalah semacam ini disebut *false rejection rate* (FRR). Di sisi lain, sistem *biometrik* mungkin menerima subjek yang salah. Ukuran kesalahan semacam ini disebut dengan *false acception rate* (FAR). Yang menjadi masalah adalah ketika sensitifitas sistem *biometrik* diatur untuk menurunkan FRR, maka FAR meningkat. Begitu juga berlaku sebaliknya. Posisi pengaturan yang terbaik adalah bila nilai FRR dan FAR seimbang, ini terjadi pada *crossover error rate* (CER).

2.5 Macam-Macam Format Gambar

Berikut ini adalah macam-macam format *file* gambar :

1. BMP (*Bitmap*)

   Format *file* ini merupakan format grafis yang fleksibel untuk *platform Windows* sehingga dapat dibaca oleh program grafis manapun. Format ini mampu menyimpan informasi dengan kualitas tingkat 1 *bit* samapi 24 *bit.* Kelemahan susah dalam pertukaran *file.*

2. JPEG (*Joint Photographic Experts Group*)

   Format *file* sering dimanfaatkan untuk menyimpan gambar yang akan digunakan untuk keperluan halaman *web, multimedia*, dan publikasi elektronik lainnya.

3. GIF (*Graphics Interchange Format*)

   Format *file* ini merupakan format standar untuk publikasi elektronik dan *internet.* Format *file* mampu menyimpan animasi dua dimensi yang akan dipublikasikan pada *internet*, desain halaman *web* dan publikasi elektronik.

4. PNG **(***Portable Network Graphics*)

   Format *file* ini digunakan untuk menampilkan objek dalam halaman *web.* Kelebihan dari format *file* ini dibandingkan dengan GIF adalah kemampuannya menyimpan *file* dalam *bit depth* hingga 24 bit serta mampu menghasilkan latar belakang (*background*) yang transparan dengan pinggiran yang halus.

2.6 Steganografi

Steganography (covered writing) didefinisikan sebagai ilmu dan seni untuk menyembunyikan pesan rahasia (*hiding message*) sedemikian sehingga keberadaan (eksistensi) pesan tidak terdeteksi oleh indera manusia.

Media yang digunakan umumnya merupakan suatu media yang berbeda dengan media pembawa informasi rahasia, dimana disinilah fungsi dari teknik *steganography* yaitu sebagai teknik penyamaran menggunakan media lain yang berbeda sehingga informasi rahasia dalam media awal tidak terlihat secara jelas.

2.6.1 Metode Steganografi pada Gambar

Sudah banyak metode yang digunakan untuk menyembunyikan pesan di dalam sebuah *image* tanpa mengubah tampilan *image,* sehingga pesan yang disembunyikan tidak akan terlihat. Berikut akan dibahas beberapa metode umum yang digunakan pada *image steganography.*

2.6.1.1 Penyisipan *Least Significant Bit*

Cara paling umum untuk menyembunyikan pesan adalah dengan memanfaatkan *Least-Significant Bit* (LSB). Walaupun banyak kekurangan pada metode ini, tetapi kemudahan implementasinya membuat metode ini tetap digunakan sampai sekarang. Metode ini membutuhkan syarat, yaitu jika dilakukan kompresi pada *stego*, harus digunakan *format lossless compression,* karena metode ini menggunakan *bit-bit* pada setiap pikses pada *image.* Jika digunakan *format lossy compression*, pesan rahasia yang disembunyikan dapat hilang. 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 piksel. Sebuah *image* 800 x 600 piksel dapat digunakan untuk menyembunyikan 1.440.000 bit (180.000 *bytes*) data rahasia.

Misalnya, di bawah ini terdapat 3 piksel dari *image* 24 *bit color* :

(00100111 11101001 11001000)

(00100111 11001000 11101001)

(11001000 00100111 11101001)

jika diinginkan untuk menyembunyikan karakter A (10000001)

dihasilkan :

(00100111 11101000 11001000)

(00100110 11001000 11101000)

(11001000 00100110 11101001)

dapat dilihat bahwa hanya 3 *bit* saja yang perlu diubah untuk menyembunyikan karakter A ini. Perubahan pada LSB ini akan terlalu kecil untuk terdeteksi oleh mata manusia sehingga pesan dapat disembunyikan secara efektif. Jika digunakan image 8 *bit color* sebagai *cover,* hanya 1 *bit* saja dari setiap piksel warna yang dapat dimodifikasi sehingga pemilihan *image* harus dilakukan dengan sangat hati-hati, karena perubahan LSB dapat menyebabkan terjadinya perubahan warna yang ditampilkan pada citra. Akan lebih baik jika *image* berupa *image grayscale* karena perubahan warnanya akan lebih sulit dideteksi oleh mata manusia. Proses ekstraksi pesan dapat dengan mudah dilakukan dengan mengekstrak LSB dari masing-masing piksel pada *stego* secara berurutan dan menuliskannya ke *output*

file yang akan berisi pesan tersebut. Kekurangan dari metode modifikasi LSB ini adalah bahwa metode ini membutuhkan tempat penyimpanan yang relatif besar. Kekurangan lain adalah bahwa *stego* yang dihasilkan tidak dapat dikompress dengan *format lossy compression*

2.6.1.2 *Masking* dan *Filtering*

Teknik *masking* dan *filtering* ini biasanya dibatasi pada *image* 24 *bit color* atau *image grayscale.* Metode ini mirip dengan *watermark,* dimana suatu image diberi tanda (*marking*) untuk menyembunyikan pesan rahasia. Hal ini dapat dilakukan, misalnya dengan memodifikasi *luminance* beberapa bagian dari *image*. Walaupun metode ini akan mengubah tampilan dari *image,* dimungkinkan untuk melakukannya dengan cara tertentu sehingga mata manusia tidak melihat perbedaannya. Karena metode ini menggunakan aspek *image* yang memang terlihat langsung, metode ini akan lebih "robust" terhadap kompresi (terutama *lossy compression*), *cropping,* dan beberapa *image processing* lain, bila dibandingkan dengan metode modifikasi LSB.

2.6.1.3 *Transformation*

Metode yang lebih kompleks untuk menyembunyikan pesan pada *image* ini dilakukan dengan memanfaatkan *Discrete Cosine Transformation* (DCT) dan *Wavelet Compression.* DCT digunakan, terutama pada kompresi JPEG, untuk metransformasikan blok 8x8 piksel yang berurutan dari *image* menjadi 64 koefisien DCT. Setiap koefisien DCT F(u,v) dari blok 8x8 piksel *image* f(x,y) dihitung sebagai berikut :

$$F(u,v) = \frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) * \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right] \quad \textbf{(2.1)}$$

dimana C(x) = 1 2 saat x sama dengan 0 dengan C(x)=1 saat x sama dengan 1. Setelah koefisien-koefisien diperoleh, dilakukan proses kuantisasi sebagai berikut:

$$F^2(u,v) = \left\lfloor \frac{F(u,v)}{Q(u,v)} \right\rfloor \quad \textbf{(2.2)}$$

dengan Q(u,v) adalah 64-elemen dari tabel kuantisasi. Walaupun *image* yang dikompresi dengan *lossy compression* akan menimbulkan kecurigaan karena perubahan LSB akan terlihat jelas, pada metode ini hal ini tidak akan terjadi karena metode ini terjadi di domain frekuensi di dalam image, bukan pada domain spasial, sehingga tidak akan ada perubahan yang terlihat pada *cover image. Wavelet Compression* adalah salah satu cara kompresi data yang cocok digunakan untuk kompresi *image, audio,* dan *video.* Tujuannya adalah untuk menyimpan data dalam "ruang" yang sekecil mungkin dalam sebuah *file*, karenanya hilangnya informasi tertentu memang sudah diharapkan akan terjadi, kompresi ini merupakan contoh *lossy compression.* Sama seperti DCT, *wavelet compression* juga berbasis pada domain frekuensi. Keuntungannya, *wavelet compression* lebih baik dalam merepresentasikan daerah transien, contohnya *image* bintang pada langit malam. Artinya, elemen dari data yang transien akan direpresentasikan dalam jumlah informasi yang lebih kecil daripada yang terjadi pada transformasi lain, seperti pada DCT. Kerugiannya, *wavelet compression* kurang baik digunakan pada data yang bersifat periodik dan *smooth.* Metode yang dilakukan pada *wavelet compression* akan dijelaskan sebagai berikut. Pertama-tama, dilakukan *wavelet transform* yang akan menghasilkan koefisien sesuai dengan jumlah piksel pada *image* sebagai berikut :

$$[W\psi f](a,b) = \frac{1}{\sqrt{|a|}} \sum_{-\infty}^{\infty} \psi \left(\frac{x-b}{a} f(x) dx \right) \quad \textbf{(2.3)}$$

Koefisien wavelet cjk diperoleh dengan :

$$c_{jk} = [W\psi f](2^{-j}, k2^{-j}) \quad \textbf{(2.4)}$$

dimana a = 2−j disebut *binary dilation* atau *dyadic dilation*, dan b = k2−j disebut *binary position* atau *dyadic position.* Setelah koefien *wavelet* diperoleh, koefisien ini dapat dikompresi dengan mudah karena informasi terkonsentrasi secara statistik pada beberapa koefisien tertentu saja. Prinsip ini disebut dengan *transform coding.* Setelah itu, koefisien-koefisien dikuantisasi, baru kemudian di-*encode* dengan *entropy encoding* atau *run length encoding*. Proses ekstraksi pesan dengan menggunakan metode transformasi ini dilakukan dengan melakukan

transformasi pada *stego* untuk memperoleh koefisien transformasi *image*. Pilih koefisien yang nilainya lebih kecil dari nilai *treshold.* Ekstrak *bit* data yang sesuai dengan koefisien ini dan tulis ke *output file* yang akan berisi pesan tersebut.

2.7 *Web Service*

2.7.1 Definisi *Web Service*

Web service adalah suatu sistem perangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antar sistem pada suatu jaringan. *Web service* digunakan sebagai suatu fasilitas yang disediakan oleh suatu *website* untuk menyediakan layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan-layanan (*service*) yang disediakan oleh suatu sistem yang menyediakan *web service*. *Web service* menyimpan data informasi dalam *format* XML, sehingga data ini dapat diakses oleh sistem lain walaupun berbeda *platform*, sistem operasi, maupun bahasa *compiler.*

Web service bertujuan untuk meningkatkan kolaborasi antar pemrogram dan perusahaan, yang memungkinkan sebuah fungsi di dalam *Web Service* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detil pemrograman yang terdapat di dalamnya.

Beberapa alasan mengapa digunakannya *web service* adalah sebagai berikut:

1. *Web service* dapat digunakan untuk mentransformasikan satu atau beberapa bisnis *logic* atau *class* dan objek yang terpisah dalam satu ruang lingkup yang menjadi satu, sehingga tingkat keamanan dapat ditangani dengan baik.
2. *Web service* memiliki kemudahan dalam proses *deployment*-nya, karena tidak memerlukan registrasi khusus ke dalam suatu sistem operasi. *Web service* cukup di-*upload* ke *web server* dan siap diakses oleh pihak-pihak yang telah diberikan otorisasi.
3. *Web service* berjalan di *port* 80 yang merupakan protokol standar HTTP, dengan demikian *web service* tidak memerlukan konfigurasi khusus di sisi *firewall*

2.7.2 Macam – Macam *Web Service*

2.7.2.1 XML-RPC

XMLRPC adalah akronim dari *eXtensible Markup Language – Remote Procedure Call*. Sebuah spesifikasi XML yang menjelaskan mengenai mekanisme pemanggilan prosedur jarak jauh dengan menggunakan XML. Bisa dikatakan, XMLRPC adalah salah satu bentuk *webservice* yang disederhanakan dari standar yang konvensional. Dua sistem yang benar-benar terpisah dan berbeda *platform* serta lingkungan bisa saling berkomunikasi lewat sarana *file* XML.

2.7.2.2 SOAP

SOAP (*Simple Object Access Protocol*) adalah standar untuk bertukar pesan-pesan berbasis XML melalui jaringan komputer atau sebuah jalan untuk program yang berjalan pada suatu sistem operasi (OS) untuk berkomunikasi dengan program pada OS yang sama maupun berbeda dengan menggunakan HTTP dan XML sebagai mekanisme untuk pertukaran data.

SOAP menspesifikan secara jelas bagaimana cara untuk meng-*encode header* HHTP dan *file* XML sehingga program pada suatu komputer dapat memanggil program pada pada komputer lain dan mengirimkan informasi, dan bagaimana program yang dipanggil memberikan tanggapan.

SOAP adalah protokol ringan yang ditujukan untuk pertukaran informasi struktur pada lingkup desentralisasi, dan terdistribusi. SOAP menggunakan teknologi XML utuk mendefinisikan rangka kerja pemesanan terekstrensi di mana menyediakan konstruksi pesan yang dapat dipertukarkan pada protokol berbeda. Rangka kerja dirancang bebas dari model pemrograman dan spesifikasi implementasi semantik

2.7.2.3 REST

REST (*Representational State Transfer*) merupakan sebuah gaya arsitekur untuk membangun suatu *web service*. Isitilah REST ini sendiri dikenalkan pada tahun 2000 dalam disertasi doctor Roy Fielding. Menurut Wikipedia, *web service*

merupakan suatu sistem atau aplikasi yang didesain untuk mendukung transaksi data antar komputer maupun sistem operasi melalui jaringan *internet*. Konsep terpenting dari REST adalah konsep untuk mengakses suatu *resources* (sumber informasi) serta metode yang digunakan untuk melakukan pertukaran *resources* dari *client ke server.*

Metode yang digunakan untuk melakukan pertukaran *resources* adalah menggunakan *http request method* atau umumnya dikenal dengan istilah "*verbs*". *Method* tersebut dapat disamakan dengan isitilah CRUD (*Create Retrieve Update Delete)* pada konsep *database*. Tabel 2.4 menggambarkan kolerasi antara *method* dengan konsep CRUD.

Tabel 2.4 Kolerasi *Method* dengan CRUD

No	*Method*	CRUD	Penjelasan
1	*Get*	*Retrieve*	Mendapatkan *Resource* yang diinginkan.
2	*Post*	*Create*	Menginputkan data/*resource* baru
3	*Put*	*Update*	Melakukan *update* terhadap *resource* yang dipilih
4	*Delete*	*Delete*	Menghapus data/*resource* yang dipilih

Pada tugas akhir ini metode steganografi menggunakan Penyisipan *Least Significant Bit* dan *web services* REST.

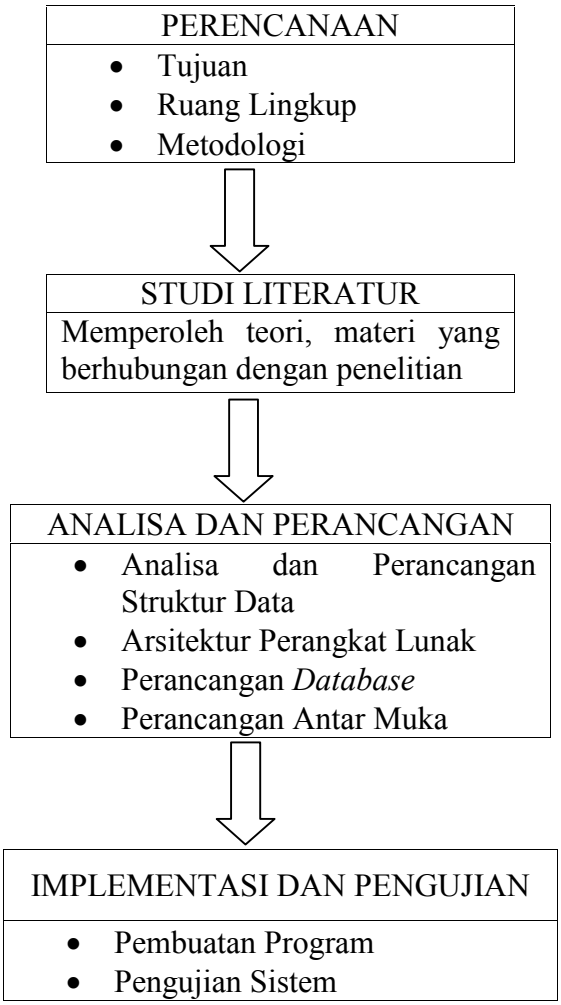
BAB III
METODOLOGI PENELITIAN

Pada bab ini akan diuraikan kerangka permasalahan dengan singkat dan langkah-langkah yang digunakan untuk membahas permasalahan yang diambil dalam penelitian ini. Pada bab ini juga akan dijelaskan alat dan metode yang digunakan untuk melakukan perencanaan dan mendapatkan spesifikasi kebutuhan perangkat lunak yang akan dibuat. Kerangka yang diusulkan diharapakan mampu memandu secara bertahap dari pengembangan penelitian dan memberikan sebuah solusi permasalahan untuk mencapai sasaran akhir dari penelitian ini.

3.1 Metodologi Penelitian

Metodologi penelitian merupakan dasar untuk memastikan bahwa semua langkah-langkah dan kegiatan penelitian lebih sistematis. Selain itu, metodologi penelitian mampu menentukan apakah penelitian akan berjalan dengan baik sehingga menghasilkan produk yang sesuai dengan tujuan penelitian ini.

Untuk mendapatkan informasi mengenai sistem *single sign-on* menggunakan autentikasi gambar, akan dilakukan beberapa pengamatan. Metodologi yang dibutuhkan untuk pengembangan penelitian mulai dari perencanaan, studi literatur, analisa dan perancangan, serta implementasi dan pengujian.



Gambar 3.1 Tahapan Metodologi Penelitian

3.1.1 Perencanaan

Tahap perancanaan merupakan tahap awal dari suatu penelitian. Pada tahap ini dilakukan serangkaian kegiatan yaitu mendiskusikan penelitian yang akan dibahas dengan para pakar yang berkompeten, menganalisa sasaran dari pengembangan penelitian dan digambarkan berdasarkan permasalahan yang ada.

Berikutnya mempelajari ruang lingkup penelitian untuk diberikan batasa-batasan pengembangan penelitian, kemudian dapat ditentukan metodologi dari penelitian tersebut.

3.1.2 Studi Literatur

Dimaksudkan untuk memperoleh teori-teori dan konsep-konsep yang mendasar mengenai materi yang berhubungan dengan sistem *single sign-on*, konsep Steganografi *Least Significant Bit*. Materi di peroleh dari buku-buku, artikel-artikel di *internet*, jurnal, serta makalah.

3.1.3 Analisa dan Perancangan

3.1.3.1 Analisa

Analisa dilakukan setelah data yang dikumpulkan telah lengkap, analisa ini menjabarkan beberapa data pendukung serta membahas dan menyelesaikan permasalahan-permasalahan yang akan diterapkan untuk membangun sistem. Adapun analisa yang dilakukan diantaranya yaitu :

1. Analisa sistem *Single Sign-On,* serta proses otorisasi sistem
2. Analisa proses metode *Least Significant Bit* yang akan digunakan untuk pengamanan gambar.
3. Analisa penggunaan *web service* REST

Pada saat menganalisa data, ada beberapa tahap yang harus dilakukan, yaitu mengidentifikasi kebutuhan sistem, fungsi sistem, memodelkan sistem dalam bentuk *flowchart* dan DFD.

3.1.3.2 Perancangan

Setelah tahap analisa selesai maka tahap selanjutnya mulai merancang sistem *Single Sign-On*. Pada tahap perancangan ini hal yang dilakukan adalah , arsitektur sistem *Single Sign-On*, membangun antarmuka sistem *Single Sign-On*, struktur *database.*

3.1.4 Implementasi dan Pengujian

Tahap terakhir yaitu pembuatan sistem berdasarkan analisa yang telah dilakukan atau mengimplementasikan. Selanjutnya dilakukan pengujian terhadap sistem yang telah dibangun agar dapat diketahui hasilnya.

3.1.4.1 Impementasi

Tahap implementasi merupakan tahap penulisan kode program berdasarkan analisa dan perancangan yang telah dilakukan. Bahasa pemrograman yang digunakan untuk membangun sistem adalah PHP dengan *database* MySQL.

3.1.4.2 Pengujian

Tahap pengujian yang dilakukan yaitu dengan menguji sistem *single sign-on* apakah sistem berjalan sesuai dengan yang diharapkan, serta menguji tingkat keamanan sistem dengan menggunakan aplikasi *Keylogger* dan *Wireshark*.

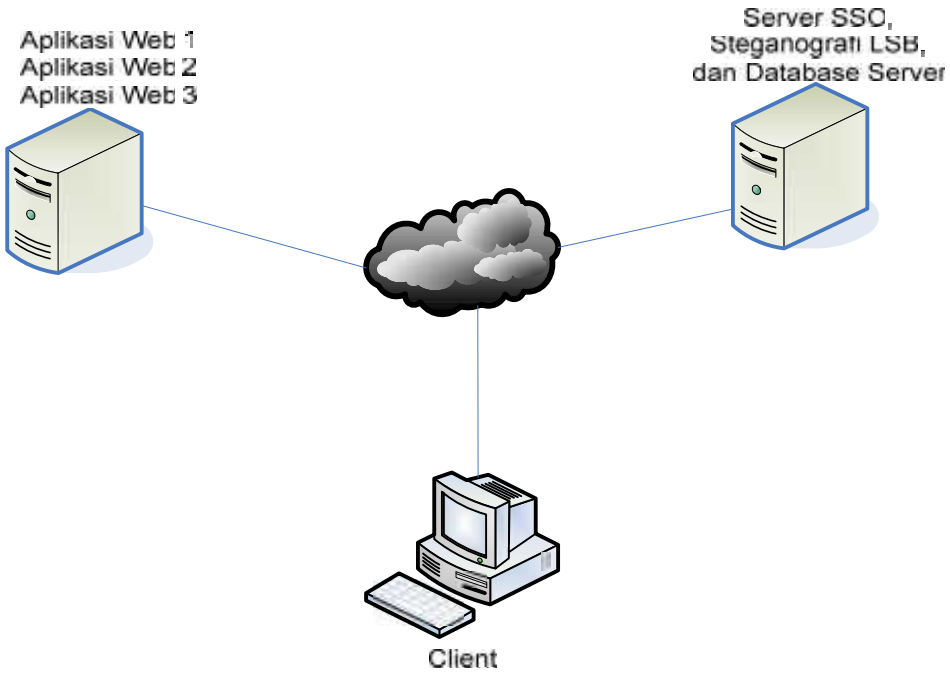
BAB IV
ANALISA DAN PERANCANGAN

Bab ini berisikan penjelasan tentang analisis dan perancangan. Subbab analisis berisi penjelasan tentang hasil analisis yang dilakukan dalam Tugas Akhir. Subbab perancangan berisi berisi penjelasan tentang hasil perancangan perangkat lunak yang dilakukan dalam Tugas Akhir.

4.1 Analisa Masalah

Masalah utama dari tugas akhir ini adalah bagaimana membangun sebuah sistem yang dapat melakukan *login* hanya sekali pada banyak aplikasi *web*. Pada subbab berikutnya akan dibahas deskripsi sistem *single sign-on* yang akan dibangun, analisis sistem *single sign-on* dan bagaimana proses *login* sistem *single sign-on* serta analisis penyisipan gambar dengan *least significant bit*.

4.1.1 Deskripsi Sistem *Single Sign On*



Gambar 4.1 Deskripsi Sistem *Single Sign On*

Pada Gambar 4.1 merupakan deskripsi sistem *single sign-on* yang akan dibangun. Topologi sistem *single sign-on* terdapat tiga bagian diantaranya adalah :

1. *Server* sistem *single sign-on*

   Server sistem SSO atau *Identity Provider* berfungsi menyediakan layanan autentikasi kepada pengguna/*client* yang membutuhkan autentikasi melalui aplikasi *web*. Pada *sistem single sign-on* tersebut, terdapat autentikasi tambahan yaitu autentikasi dengan gambar (*Least Signficant Bit*).

2. Aplikasi *Web* 1, Aplikasi *Web* 2, Aplikasi *Web* 3

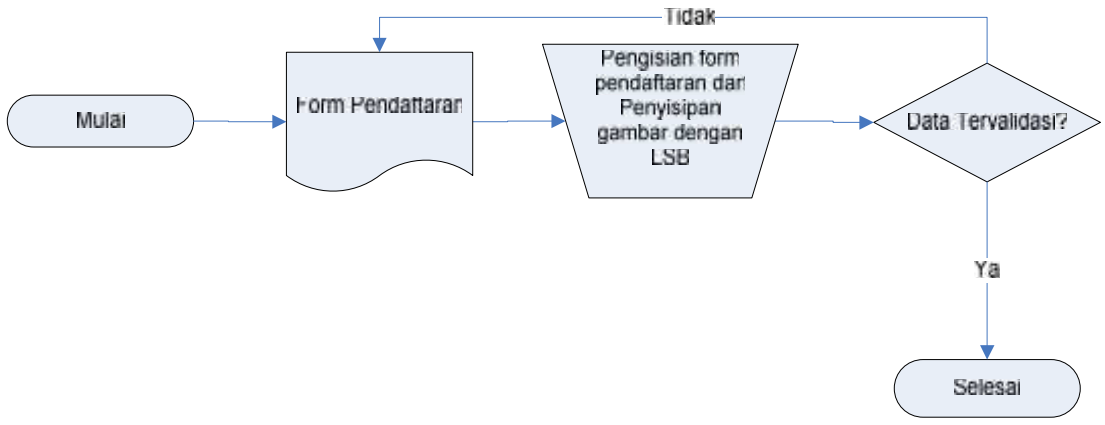
   Pada bagian ini merupakan *client* dari *server* SSO (aplikasi *web*) yang mana permintaan autentikasi diberikan kepada pengguna dan mengelola seluruh aliran autentikasi pengguna. *Client* SSO memvalidasi SSO sesi dan memperoleh informasi pengguna terkait dengan layanan *web server* SSO dengan menggunakan protokol REST

3. *Client* (Pengguna)

Seluruh *server* tersebut terkoneksi oleh jaringan, yang dapat dilihat pada gambar 4.1.

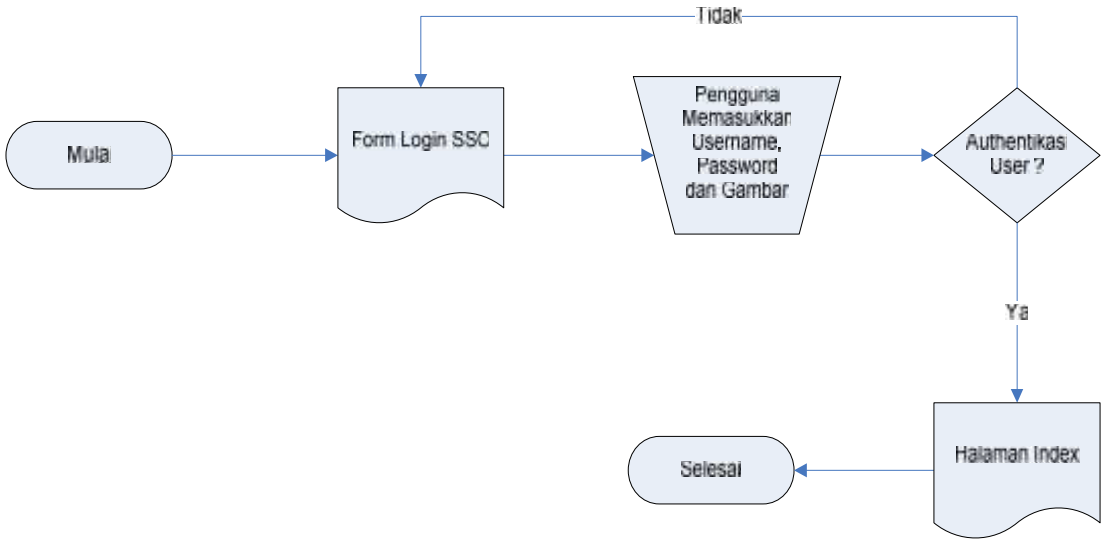
4.1.2 Analisa Sistem *Single Sign-On*

Model kerja sistem yang akan dibangun dapat digambarkan pada *flowchart* sebagai berikut



Gambar 4.2 Diagram Alir Pendaftaran *Single Sign-On*

Gambar 4.2 merupakan alir sistem pendaftaran pengguna baru, dimana seorang pengguna melakukan pendaftaran terlebih dahulu sebelum dapat mengakses sistem *single sign-on*, agar mendapatkan *account*. Calon *member* mengisi *form* pendaftaran serta melakukan penyisipan gambar, yang akan digunakan untuk autentikasi *login*.



Gambar 4.3 Diagram Alir *Login* Sistem *Single Sign-On*

Gambar 4.3 merupakan alur sistem *single sign-on* yang akan dibangun nantinya. Seorang *member* mengakses sebuah situs *web*, dan melakukan *login*, maka secara otomatis akan dibawa ke halaman Sistem *single sign-on Server*. *Member* memasukkan *username*, *password* dan *file* gambar. Jika data *member* terdaftar pada *server* maka *login* akan berhasil. Jika data *member* tidak terdaftar maka akan dikembalikan lagi ke halaman *login single sign-on*. Pada proses *login*, autentikasi dengan gambar melakukan perbandingan antara gambar yang ada di *server* dengan gambar yang dimasukkan *member* pada saat *login*.

Gambar yang digunakan sebagai autentikasi sebelumnya telah disisipi *text* yaitu menggunakan Metode *Least Significant Bit*. Untuk analisis penyisipan data pada gambar dengan menggunakan metode *Least Significant Bit* akan di jelaskan pada 4.1.4.

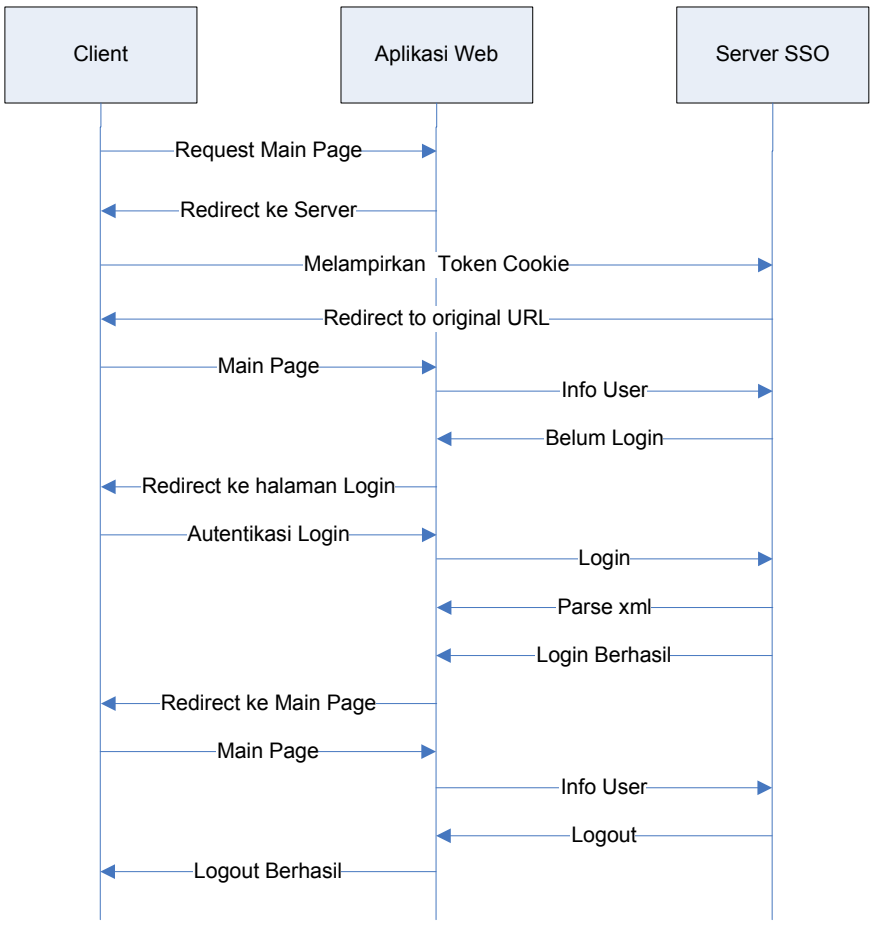
Sistem *Single sign-on* yang dibangun kali ini yaitu berbasiskan Otorisasi, karena sistem *single sign-on* ini diperuntukkan hanya pada satu lingkungan organisasi, yang mana prosesnya akan di jelaskan pada 4.1.3.

4.1.3 Analisa Proses Otorisasi Sistem *Single Sign-On*

Pada subbab ini akan dijelaskan bagaimana proses otorisasi pada sistem *single sign-on* yang akan dibangun. Pada sistem *single sign-on* ini terdapat tiga pihak yaitu :

1. *Client* adalah pengunjung situs *web*
2. Aplikasi Web adalah situs *web* yang dikunjungi
3. *Server* SSO

Berikut ini adalah gambaran proses *login* sistem *single sign-on* :



Gambar 4.4 Proses *Login* Sistem *Single Sign-On*

Dari gambar 4.4 dapat dijelaskan, ketika klien mengunjungi sebuah aplikasi *web*, disaat itu akan diperiksa apakah *token cookie* sudah ada pada *browser*. Berikut adalah contoh *pseudocode :*

```
Fungsi pemeriksaan session
function auto_attach = true
        if session token cookie = session token cookie
        if  auto_attach and session token
        redirect ke server untuk merequest URL
        end if
end
```

Algoritma 4.1 Fungsi Pemeriksaan *Session*

Jika belum ada, maka aplikasi *web* melakukan *redirect* ke *server* SSO dengan memberikan perintah untuk melampirkan sebuah *session* dan menetapkan identitas aplikasi *web* . URL *token* akan diacak dan *token* disimpan kedalam *cookie*. *Server* SSO akan menghasilkan *session key* berdasarkan aplikasi *web*, identitas aplikasi *web* dan *token* ini disimpan pada sesi klien/*browser*. *Session key* berisikan sebuah enkripsi MD5, sehingga seorang *hacker* tidak dapat mengetahui *session* tersebut. Berikut adalah contoh enkripsi *session* ke MD5.

```
function Get Session Id
        if session token null
          then session token di enkripsi ke md5
       end if
end
```

Algoritma 4.2 Fungsi Enkripsi *Session* ke MD5

Server akan me-*redirect* klien kembali ke URL aslinya. Setelah ini, klien dapat berkomunikasi dengan aplikasi *web*. Ketika klien mengakses aplikasi *web*, *server* akan merespon kepada aplikasi *web* bahwa pengunjung sedang tidak *login*, maka akan langsung di-*redirect* ke halaman *login*. Disini klien melakukan *login* dengan memasukkan *username, password* dan gambar, aplikasi *web* akan mengirimkan data tersebut ke *server* dengan melewati *session key*. Berikut adalah contoh *pseudocode* :

```
Fungsi redirect ke halaman login
if session tidak ada then logout
        else if request to server  then
          halaman index
```

```
        end if
end
```

Algoritma 4.3 Fungsi *Redirect* Halaman *Login*

```
Fungsi login (username, password)
If isset($username) and request($username)
Memanggil fungsi parseInfo xml
Login username=$username, password=$password
end if
end
```

Algoritma 4.4 Fungsi *Login*

```
Fungsi parse xml
function parse info xml
  xml object then
   userinfo = indentity
  key userinfo = string
end
```

Algoritma 4.5 Fungsi *Parse* XML

```
Fungsi logout
Function logout
Menghapus sessionID
Memanggil fungsi parseInfo xml
End
```

Algoritma 4.6 Fungsi *Logout*

Jika data tesebut benar maka *server* akan mengembalikan ke aplikasi *web* bahwa *login* berhasil, begitu sebaliknya ketika *client* (*member*) melakukan *logout* pada salah satu aplikasi *web*, secara otomatis aplikasi *web* yang lain akan *logout* dengan sendirinya tanpa perlu *logout* untuk kedua kalinya.

4.1.4 Analisa Penyisipan Data Pada Gambar dengan Metode *Least Significant Bit*

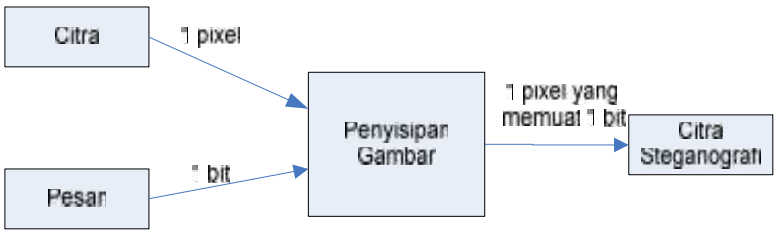
Tahap *embedding* atau penyisipan gambar, dimulai dengan mengubah citra digital yang tersusun atas *pixel* dengan menggunakan sinyal RGB (*Red, Green Blue*), menjadi *raster* data (bilangan *biner*) untuk menyisipkan pesan dalam informasi.

Kemudian bilangan *biner* dalam setiap *pixel* diambil bit rendah yaitu 1 *bit* terakhir dari sinyal *Blue*. Untuk pesan rahasia bertipe karakter akan diubah menjadi bilangan desimal yang kemudian akan diubah menjadi *bit* pesan (bilangan *biner*).

Penyisipan data atau pesan dilakukan dengan mengganti setiap *bit* rendah pada gambar dengan *bit* pesan. Jika *bit* rendah kurang dari *bit* pesan, maka *raster* ditambah 1. Jika *bit* rendah lebih besar dari *bit* pesan, maka raster data dikurang 1. Jika *bit* rendah sama dengan *bit* pesan, maka *raster* tidak dirubah.

Setelah itu dilakukan penyusunan kembali *pixel* yang sebelumnya telah disisipi *bit* pesan sesuai dengan *raster* data.

Pada gambar 4.5 merupakan metode penyisipan data ke dalam gambar :



Gambar 4.5 Metode Sisip

Untuk membentuk tahap *embedding* di perlukan sebuah fungsi sehingga data yang di peroleh dapat disisipkan kedalam *file* pembawa. Fungsi yang digunakan yaitu fungsi *write* (*file citra*) dan fungsi *read* (penyisipan pesan dalam gambar).

Berikut contoh *pseudocode Least Significant Bit* yang digunakan :

```
Algortima Least Significant Bit
Function write (data)
   Bits = ascii to biner
   Lenbit = strlen (bits)
     nx = imagesx (image object)
     ny = imagesy (image object)
  for (x=0, bit=0, x<nx, x++)
  for (y=0, y<ny, y++)
     pix = getcolor(image object, x, y)
  array (R,G,B)
end

function Read
```

```
    nx = imagesx(image object)
    ny = imagesy(image object)
     for x = 0, y<nx, x++)
     for y = 0, y<ny, y++)
   pix = getcolor(image object, x, y)
  data = (pix(R + 1) , (pix(G + 1), (pix(B + 1)
 return biner to ascii
end
```

Algoritma 4.7 Algoritma *Least Significant Bit*

Dari algoritma 4.7 maka *cover carrier* dapat disisipkan pesan dan menghasilkan gambar steganografi dengan tidak menurunkan tingkat kualitas gambar setelah disisipi, Pada analisa tersebut penggunaan gambar berformat PNG lebih kompatibel, karena perbandingan antara *file* asli dengan *file* setelah di lakukan steganografi, tidak memiliki banyak perubahan. Oleh karena itu format PNG tersebut digunakan sebagai autentikasi sistem *single sign-on* pada tugas akhir ini.

4.1.5 Analisa Penggunaan *Web Service* REST

REST (*Representational State Transfer*) *web service* adalah salah satu cara pendistribusian data yang populer saat ini antara *server* dan *client*, dengan menggunakan protokol HTTP. Alasan menggunakan *web service REST* adalah *REST* mudah dipelajari karena aplikasi *web* hanya perlu menggunakan koneksi HTTP ke URL tertentu dan tidak tergantung pada aturan *XML, web service REST* lebih digunakan pada *browser* dibandingkan dengan *web service* lainnya. Pada *REST*, metode HTTP yang digunakan diantaranya :

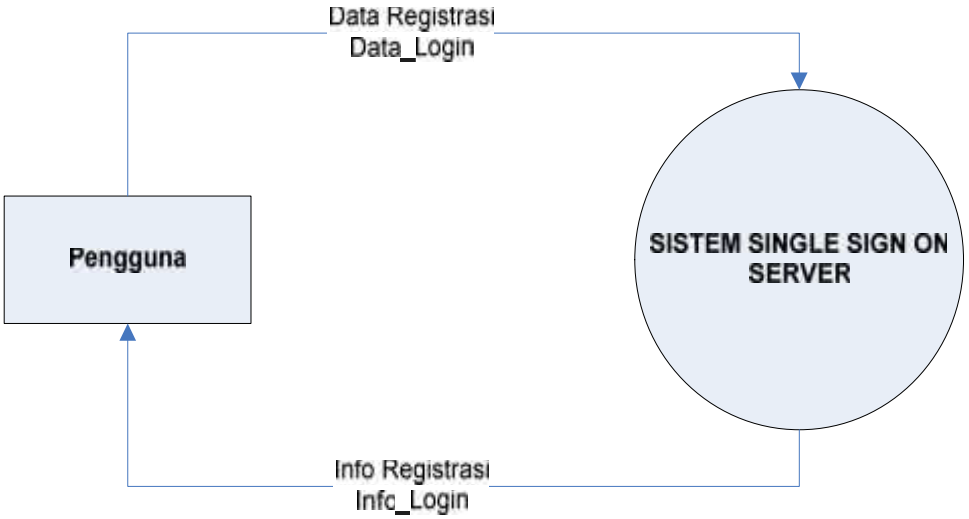
1. *POST* untuk membuat sumberdaya (*resource*) pada *server.*
2. *GET* untuk menerima sumberdaya.
3. *PUT* untuk merubah atau memperbaharui sumberdaya
4. *DELETE* untuk menghapus sumberdaya.

4.2 Deskripsi Fungsional

Aliran informasi yang ditransformasikan pada saat data bergerak dari *Input* menjadi *output* dapat dilihat di *Context Diagram* dan *Data Flow Diagram (DFD)*

4.2.1 *Context Diagram*

Diagram konteks *(Context Diagram)* digunakan untuk menggambarkan hubungan *input/output* antara sistem dengan dunia luarnya (kesatuan luar) suatu diagram kontek selalu mengandung satu proses, yang mewakili seluruh sistem.
Sistem *Single Sign-On* memiliki entitas yaitu pengguna.

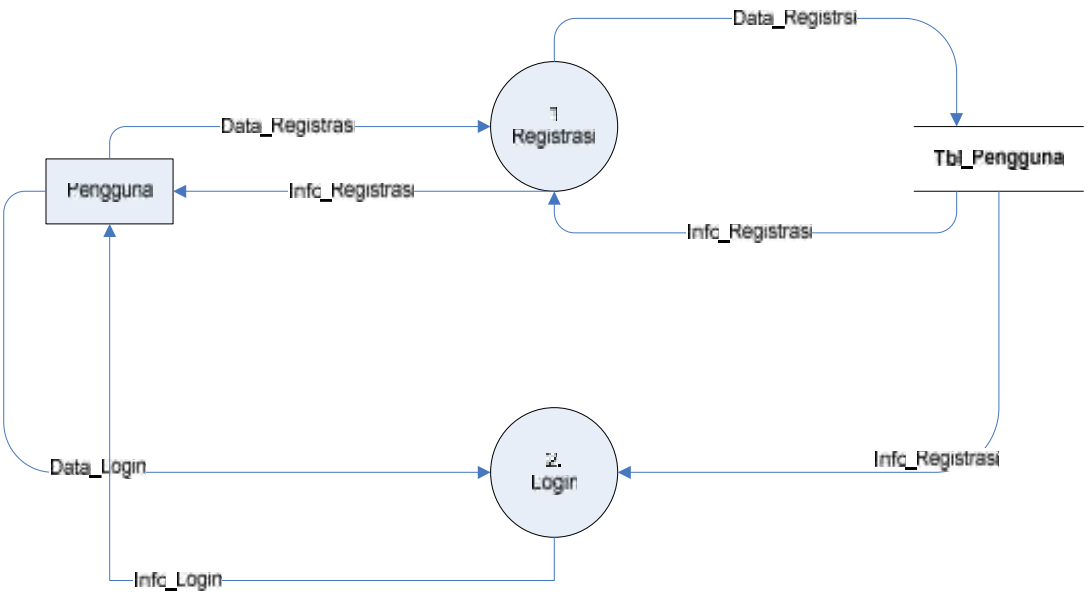


Gambar 4.6 *Context Diagram*

Entitas luar yang berinteraksi dengan sistem adalah Pengguna yang terdiri dari *Input* Data Registrasi dan *Input* Data *Login*

4.2.2 *Data Flow Diagram*

Data Flow Diagram (DFD) digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika.



Gambar 4.7 *Data Flow Diagram* (DFD) *Level* 1

Gambar 4.3 merupakan DFD level 1 dari Diagram Kontek diatas yang dipecah menjadi 2 (dua) buah proses beserta aliran datanya. Untuk keterangan masing-masing dapat dilihat kamus data pada tabel 4.1 berikut ini.

Tabel 4.1 Keterangan proses pada DFD level 1

No	Nama Proses	Masukan	Keluaran	Deskripsi
1	Registrasi	– *Input* data registrasi	– Status registrasi – Status *login*	Proses untuk melakukan registrasi *account single sign-on*
2	*Login*	– *Input username* – *Input Password* – *Input Autentikasi Gambar*	– Status *login*	Proses untuk melakukan *login* ke dalam sistem *single sign-on*

4.2.3 Perancangan Tabel

Berikut ini deskripsi perancangan tabel dalam *database.*

Tabel Pengguna

Nama : Pengguna
Deskripsi isi : Berisi data Pengguna
Primary key : id

Tabel 4.2 Pengguna

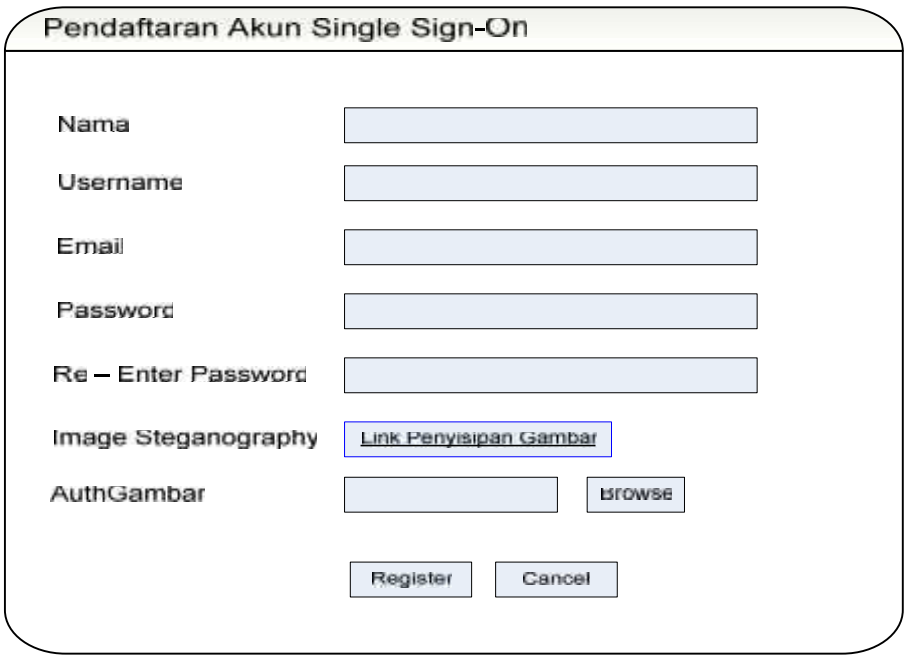
Nama *Field*	*Type* dan *Length*	Deskripsi	*Null*	*Default*
id	int (10)	ID	No	
nama	Varchar (50)	Nama	No	
username	Varchar (30)	*Username*	No	
password	Varchar (20)	*Password*	No	
email	Varchar (30)	*Email*	No	
image_auth	*Text*	Lokasi *File* Gambar	No	

4.2.4 Perancangan Antar Muka

Pada subbab ini akan ditampilkan rancangan antar muka sistem *single sign-on.* Sistem ini memiliki beberapa antar muka yaitu halaman pendaftaran pengguna, halaman *login* sistem *single sign-on*, halaman penyisipan gambar serta halaman utama atau halaman indeks.

4.2.4.1 Perancangan *Form* Registrasi Pengguna

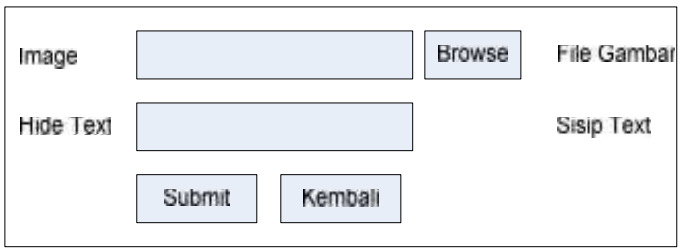
Berikut ini adalah perancangan *form* registrasi calon *member,* yang terdiri dari *Name, username, Your Email, password, Re-Enter Password*, serta *ImageAuthentication*. *Image Steganography* merupakan *link* ke *form* steganografi.



Gambar 4.8 Perancangan *Form* Registrasi

4.2.4.2 Perancangan *Form* Steganografi

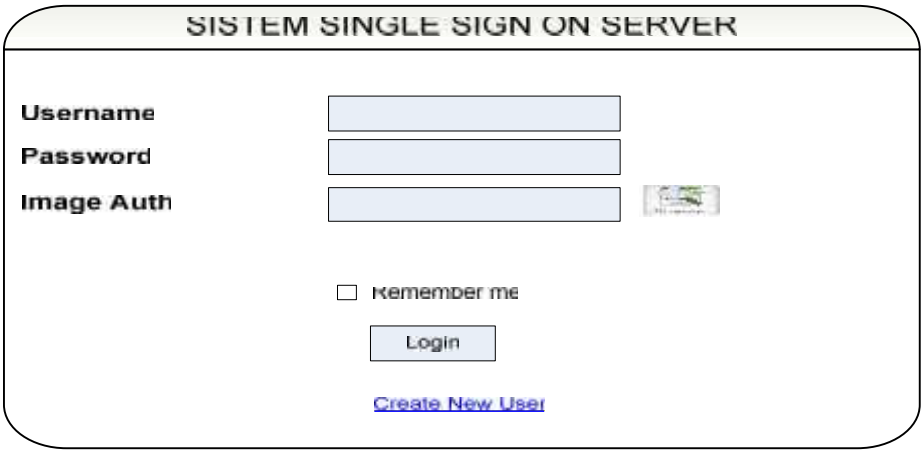
Berikut adalah *form* steganografi, yang terdiri dari *image* atau gambar yang akan di disisipkan sebuah data atau *file,* serta *hide file* yaitu data yang akan disisipkan.



Gambar 4.9 Perancangan *Form* Steganografi

4.2.4.3 Perancangan *Form Login Single Sign On*

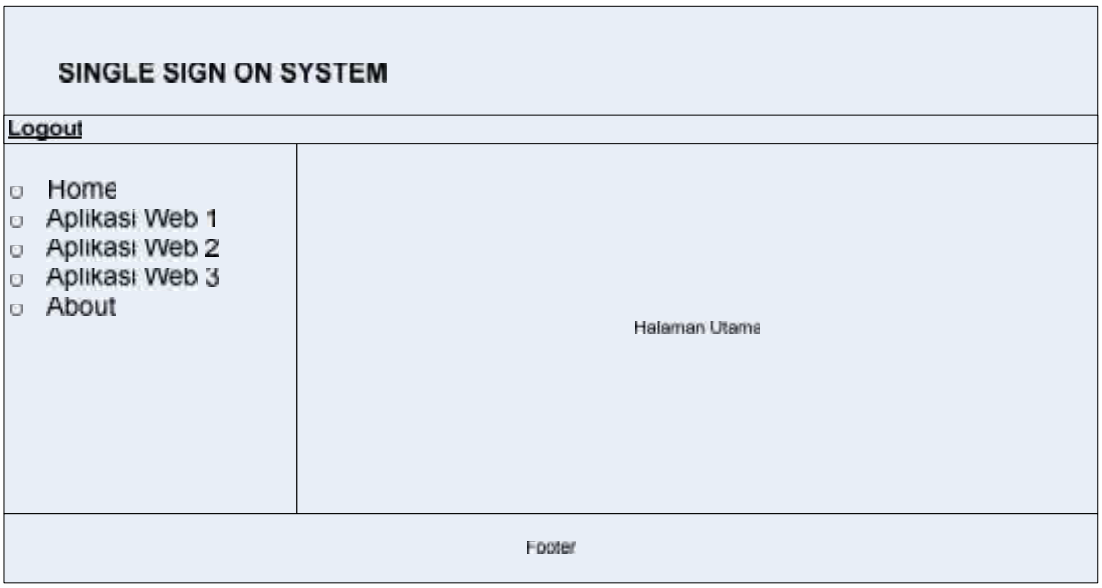
Berikut ini adalah perancangan *form login single sign-on* yang terdiri dari *username, password,* serta autentikasi gambar.



Gambar 4.10 Perancangan *Form Login Single Sign-On*

4.2.4.4 Perancangan Halaman Utama

Berikut adalah perancangan halaman utama atau *index,* jadi seorang *member* mengakses berbagai aplikasi *web* dari halaman tersebut. Pada halaman tersebut terdapat beberapa menu, diantaranya adalah *Home*, Aplikasi *Web* 1, Aplikasi *Web* 2, Aplikasi *Web* 3, dan *About.*



Gambar 4.11 Perancangan Halaman Utama

BAB V
IMPLEMENTASI DAN PENGUJIAN

5.1 Implementasi Sistem

Implementasi merupakan tahap dimana sistem siap dioperasikan pada keadaaan yang sebenarnya, sehingga akan diketahui apakah sistem yang dibuat benar-benar dapat menghasilkan tujuan yang ingin dicapai.

5.1.1 Lingkungan Implementasi

Pada prinsipnya setiap desain sistem yang telah dirancang memerlukan sarana pendukung yaitu berupa peralatan-peralatan yang sangat berperan dalam menunjang penerapan sistem yang didesain terhadap pengolahan data.

Implementasi sistem *single sign-on* dilakukan dalam lingkungan perangkat keras komputer yang memiliki spesifikasi sebagai berikut :

1. Prosesor Intel Core i3 2.40 GHz
2. Laptop DELL Inspiron 14'' resolusi 1280 x 768 *pixels*
3. Memori 2 GB
4. Harddisk 320 GB

Sedangkan lingkungan pengembangan sistem tersebut memiliki spesifikasi perangkat lunak sebagai berikut :

1. Sistem Operasi Microsoft Windows XP Profesional SP 2
2. Adobe Photoshop 7.0
3. XAMPP for Windows 1.6.4 dengan PHP 5.2.4 dan *Database* 5.0.45
4. *Web browser* Mozilla Firefox 3.6
5. VMware Workstation 5.5.1
6. Notepad ++ v5.0.2

5.1.2 Batasan Implementasi

Implementasi sistem *single sign-on* memiliki batasan sebagai berikut :

1. Implementasi dilakukan pada jaringan lokal, yaitu menggunakan aplikasi *virtual* VMWare.
2. Gambar yang digunakan sebagai autentikasi berformat PNG

5.1.3 Teknis Implementasi Sistem *Single Sign-On*

Teknik implementasi sistem dilakukan dengan beberapa tahapan sebagai berikut :

1. Implementasi pendaftaran akun *single sign-on*

   Pada tahap implementasi ini dilakukan pendaftaran akun sistem *single sign-on* pada *server single sign-on*
2. Implementasi penyisipan gambar dengan *Least Significant Bit*

   Pada tahap ini dilakukan penyisipan teks pada gambar, yang mana metode yang di gunakan menggunakan *Least Significant Bit.*
3. Implementasi *login single sign-on*

   Tahap ini di lakukan *login* pada *web portal*, ketika member menekan tombol *login* maka member akan di arahkan ke halaman *server single sign-on*. Disini *member* memasukkan *username, password* serta gambar. Ketika *member* mengunjungi pada *web portal* yang lain maka sistem akan memeriksa apakah *token cookie* sudah ada, jika *cookie*-nya sudah ada maka secara otomatis sudah masuk *login, member* tidak perlu memasukkan *username* dan *password* lagi.

Sistem *single sign-on* diimplementasikan dengan menggunakan aplikasi Notepad ++ dengan bahasa pemrograman PHP. *File* implementasi sistem *single sign-on* dapat dilihat pada tabel 5.1 dan 5.2 .

Tabel 5.1 Daftar *File Server Single Sign-On*

No	Nama *File*	Keterangan
1	Ssoserver.php	*File* ini digunakan untuk melakukan proses autentikasi dengan membuat *token cookie*, sehingga sistem bisa *single sign-on*
2	Sso.php	*File* ini merupakan *file* yang berada pada sisi *client*, dengan *file* ini, ketika seorang melakukan *login* maka akan *redirect* ke *server.*
3	Stego.php	*File* ini digunakan untuk melakukan penyisipan gambar dengan metode *least significant bit*.
4	Index.php	*File* ini merupakan halaman antar muka sistem *single sign-on*
5	Register.php	*File* ini digunakan untuk proses pendaftaran member
6	Login.php	*File* ini merupakan halaman antarmuka *login* sistem *single sign-on*

Tabel 5.2 Daftar *File* Pada *Client Web Portal*

No	Nama *File*	Keterangan
2	Sso.php	*File* ini merupakan *file* yang berada pada sisi *client*, dengan *file* ini, ketika seorang melakukan *login* maka akan *redirect* ke *server.*
4	Index.php	*File* ini merupakan halaman antar muka *web portal*
6	Login.php	*File* ini merupakan halaman antarmuka *login* sistem *web portal*

Implementasi secara rinci dapat dilihat pada lampiran B.

5.2 Pengujian Sistem

Pemrograman merupakan kegiatan penulisan program yang akan dieksekusi oleh komputer berdasarkan hasil dari analisa dan perancangan sistem. Sebelum program diimplementasikan, maka program tersebut harus bebas dari kesalahan. Pengujian program dilakukan untuk menemukan kesalahan-kesalahan yang mungkin terjadi.

5.2.1 Lingkungan Pengujian Sistem

Setelah tahap implementasi dilakukan maka tahap selanjutnya dengan pengujian dari implementasi yang telah dibuat. Tahap pengujian diperlukan agar dapat diketahui hasil dari program implementasi sistem. Program merupakan

kegiatan penulisan kode program yang akan dieksekusi oleh komputer berdasarkan hasil dari analisis dan perancangan sistem.

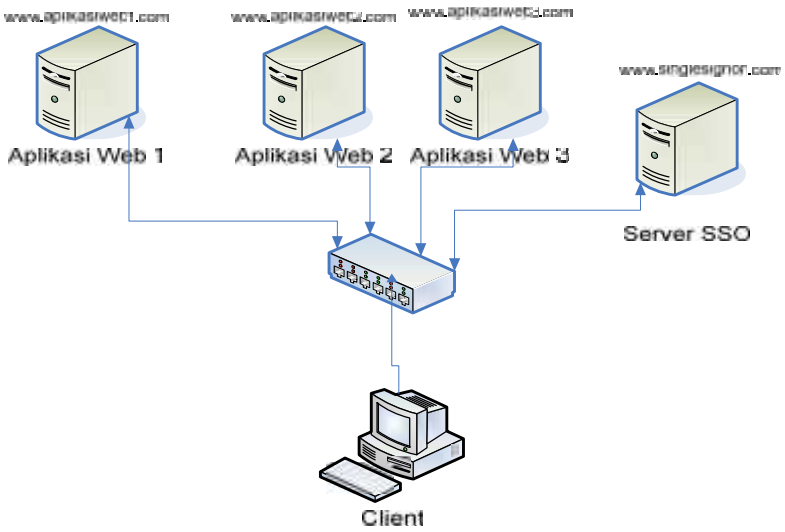
Lingkungan pengujian yang digunakan memiliki spesifikasi perangkat keras komputer sebagai berikut :

1. Prosesor Intel Core i3 2.40 GHz
2. Laptop DELL Inspiron 14'' resolusi 1280 x 768 pixels
3. Memori 2 GB
4. Harddisk 320 GB

Sedangkan lingkungan pengujian memiliki spesifikasi perangkat lunak sebagai berikut :

1. Sistem Operasi Microsoft Windows XP Profesional SP 2
2. XAMPP for Windows 1.6.4 dengan PHP 5.2.4 dan *Database* 5.0.45
3. *Web browser* Mozilla Firefox 3.6
4. VMware Workstation 5.5.1
5. Notepad ++ v5.0.2

Alamat DNS yang akan digunakan pada tugas akhir ini menggunakan alamat *virtual host*. Pada gambar 5.1 adalah arsitektur sistem *single sign-on* yang akan di ujikan :



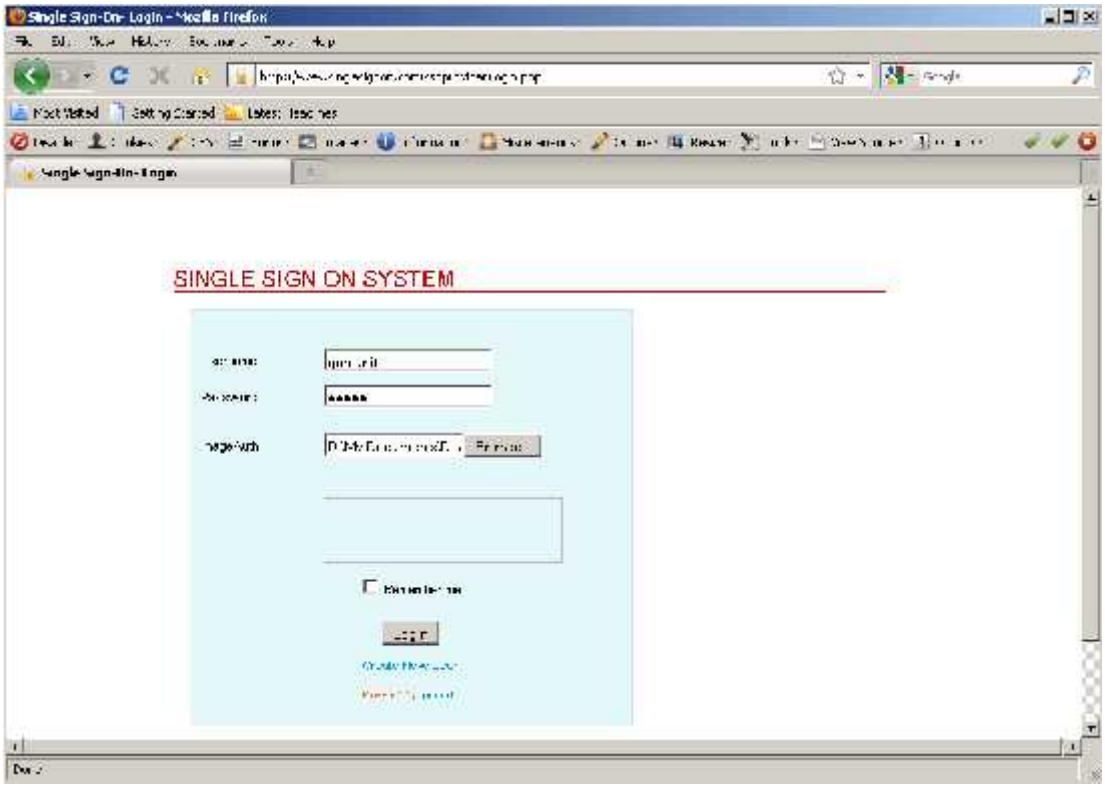
Gambar 5.1 Lingkungan Pengujian Sistem S*ingle Sign-On*

5.2.2 Pengujian Sistem *Single Sign-On*

Pada pengujian aplikasi *web* sistem *sigle sign-on*, bahwa seorang pengguna mengakses kesalah satu dari aplikasi *web*, *web* kemudian akan diarahkan ke *single sign-on server* dan pengguna memasukkan *credential*-nya. Ketika pengguna berhasil *login* maka secara otomatis seluruh aplikasi *web* yang lain sudah terbuka dengan sendirinya tanpa memasukkan *credential* lagi.

Proses pengujiannya adalah sebagai berikut :

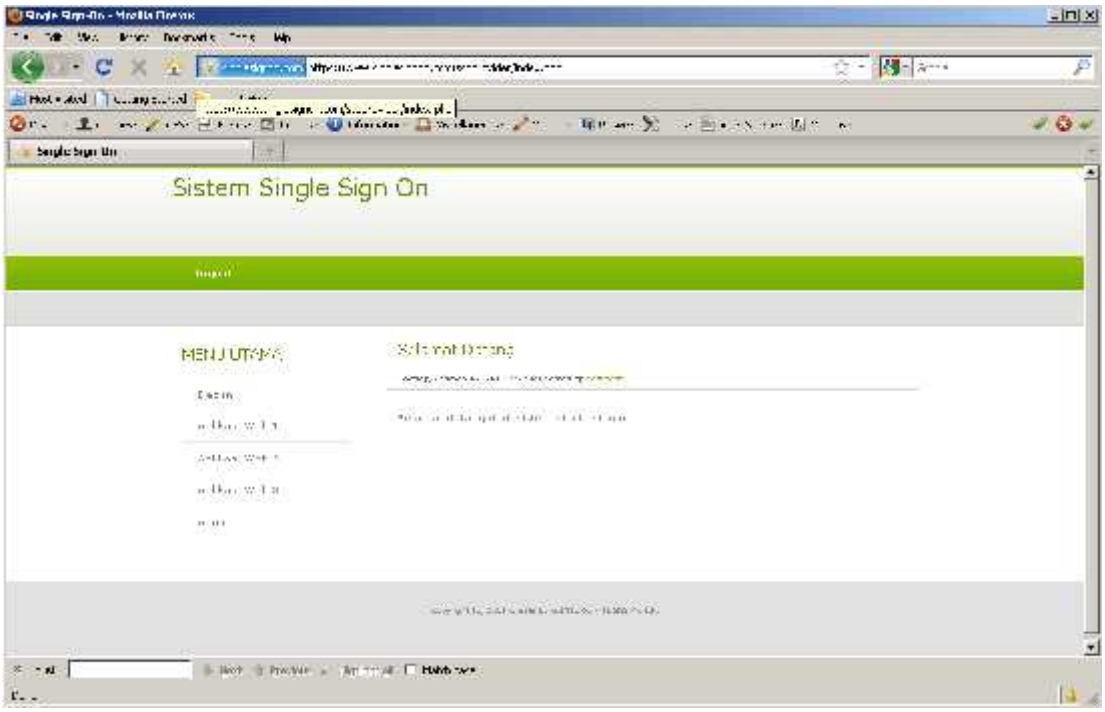
1. *Member* melakukan *login* pada *server single sign-on* dengan memasukkan *username, password* dan gambar.



Gambar 5.2 Halaman *Login* Sistem *Single Sign-On*

Pengujian sistem *single sign-on* dengan memasukkan *username, password* serta gambar, jika *username, password* dan gambar sesuai dengan di *server* maka *login* telah berhasil, dan langsung menuju ke halaman utama *web portal.*

2. *Member* berhasil melakukan *login* dan memasuki halaman *web portal*



Gambar 5.3 Halaman Utama Sistem *Single Sign-On*

3. *Member* mengakses Aplikasi *Web* 1 dan berhasil melakukan *login* tanpa memasukkan *credential.*

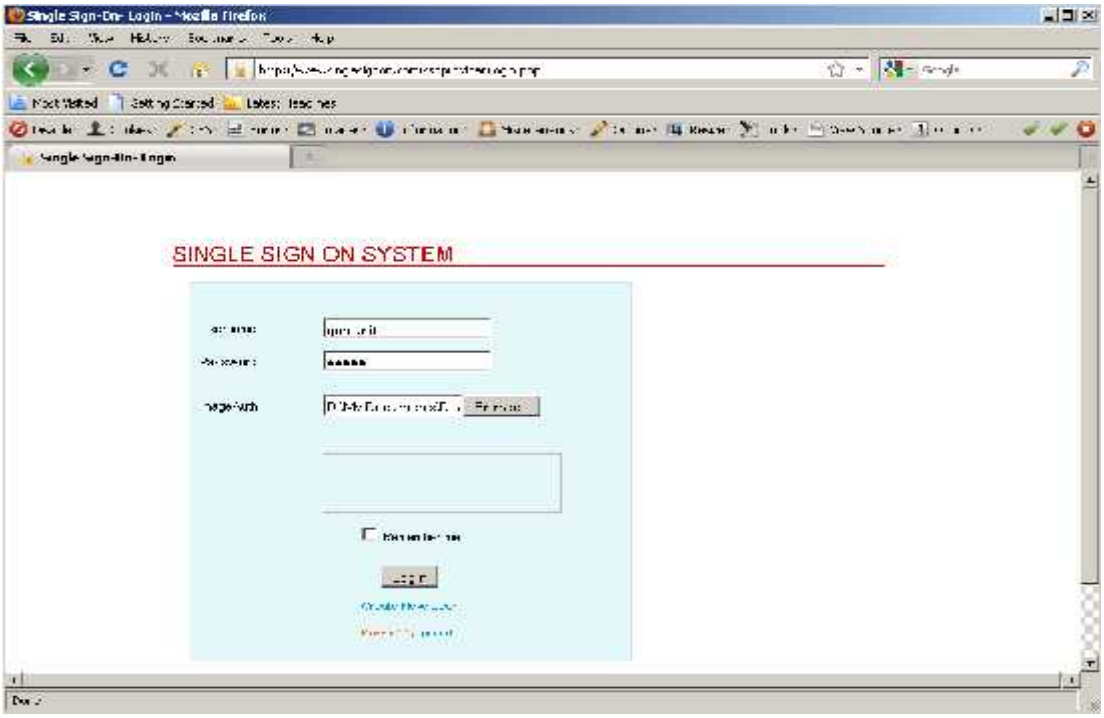
Gambar 5.4 Aplikasi *Web* 1

Ketika *member* mengakses aplikasi *web* yang lain maka secara otomatis *web* sudah dalam keadaan *login,* tanpa harus melakukan *login* untuk kedua kalinya dan begitu pula selanjutnya. Begitu pula sebaliknya, ketika seorang member melakukan *logout* dari salah satu situs *web* maka secara otomatis situs yang lain akan *logout* dengan sendirinya tanpa perlu menekan tombol *logout* kedua kalinya. Pengujian sistem secara rinci dilihat pada lampiran C.

5.2.3 Pengujian Berdasarkan Prosedur Sistem *Single Sign-On*

Pada tahap ini dilakukan pengujian terhadap sistem *single sign-on* yang telah dibangun, apakah sudah memenuhi beberapa prosedur-prosedur dalam membangun sistem *single sign-on*. Beberapa prosedur yang akan diujikan adalah *Authentication, Strong Authentication* dan *Authorization*.

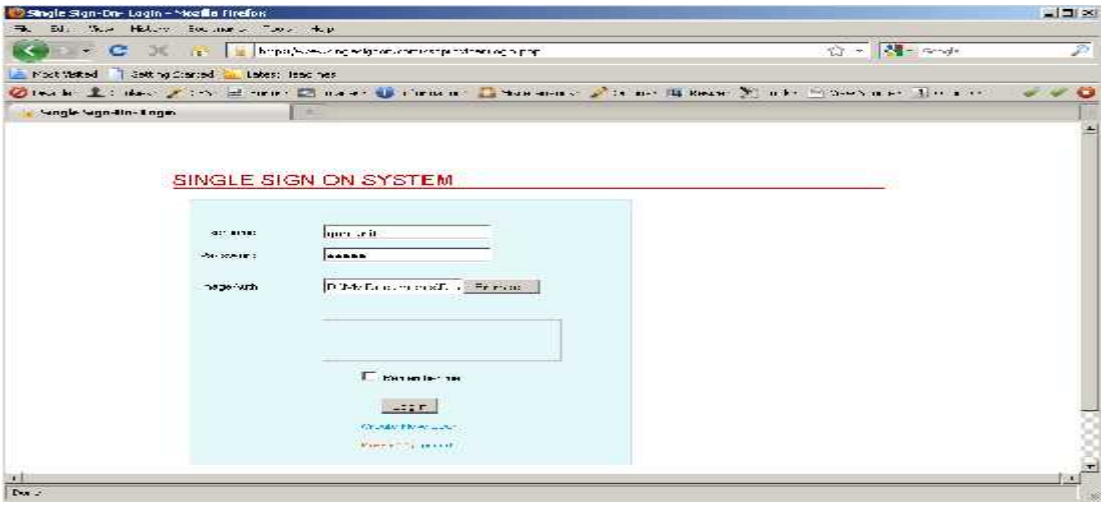
5.2.3.1 Pengujian *Authentication*



Gambar 5.5 Pengujian *Authentication*

Dari gambar 5.5 dapat dilihat bahwa *pengujian authentication* yaitu dengan menggunakan *username* dan *password* telah berhasil diujikan.

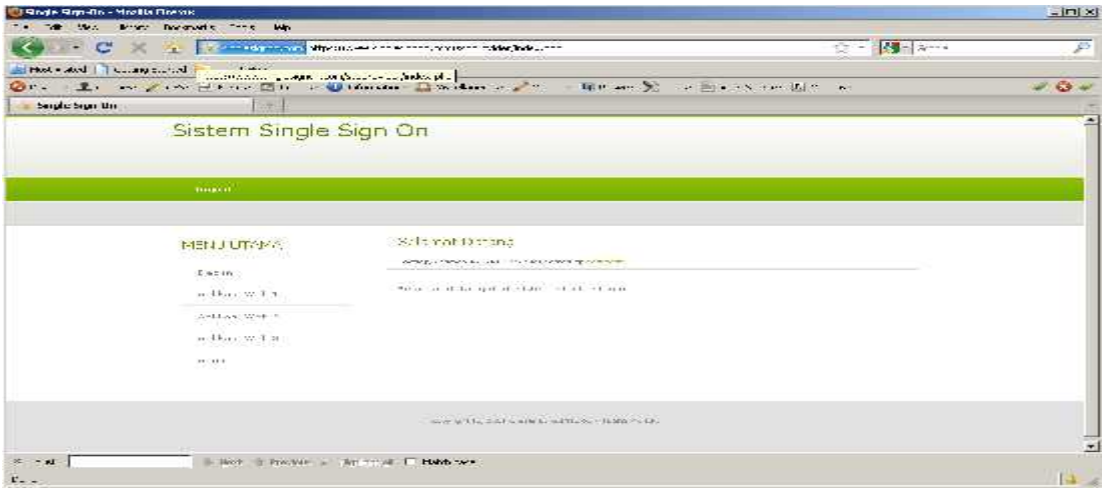
5.2.3.2 Pengujian *Strong Authentication*



Gambar 5.6 Pengujian *Strong Authentication*

Dari gambar 5.6 dapat dilihat bahwa *pengujian strong authentication* yaitu dengan menggunakan autentikasi gambar telah berhasil diujikan

5.2.3.3 Pengujian *Authorization*



Gambar 5.7 Pengujian *Authorization*

Gambar 5.8 Pengujian *Authorization* Aplikasi Web 1

Dari gambar 5.7 dan 5.8 dapat dilihat bahwa sistem *single sign-on* yang dibangun dapat melakukan otorisasi, ketika aplikasi *web* 1 dibuka maka secara otomatis sudah *login*, tanpa harus melakukan autentikasi. Begitu juga dengan aplikasi *web* yang lain.

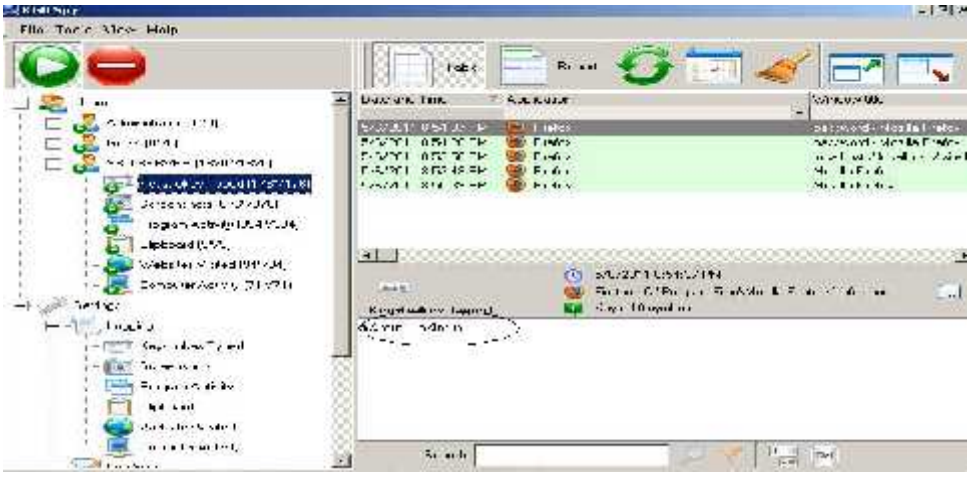
5.2.4 Pengujian Keamanan Sistem *Single Sign-On*

Pada tahap ini di lakukan pengujian terhadap keamanan sistem *sigle sign-on* yang telah di bangun dengan menggunakan aplikasi *keylogger,* aplikasi *sniffer.* Aplikasi *keylogger* dan *sniffer* yang akan digunakan pada pengujian ini adalah sebagai berikut :

Tabel 5.3 Spesifikasi Aplikasi Pengujian

Spesifikasi	***KGB Employee Monitor***	***Wireshark***
Version	4.5.4	1.2.10
Jenis Aplikasi	*Keylogger*	*Sniffer*
Size	5.2 MB	7.5 MB

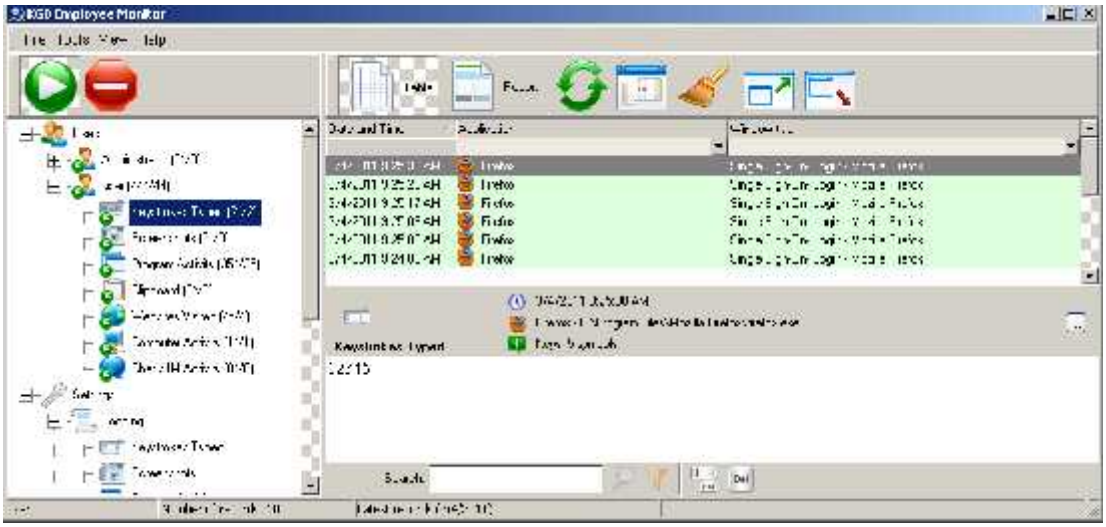
1. **Pengujian Sistem Tanpa *Single Sign-On* dengan Aplikasi *Keylogger***



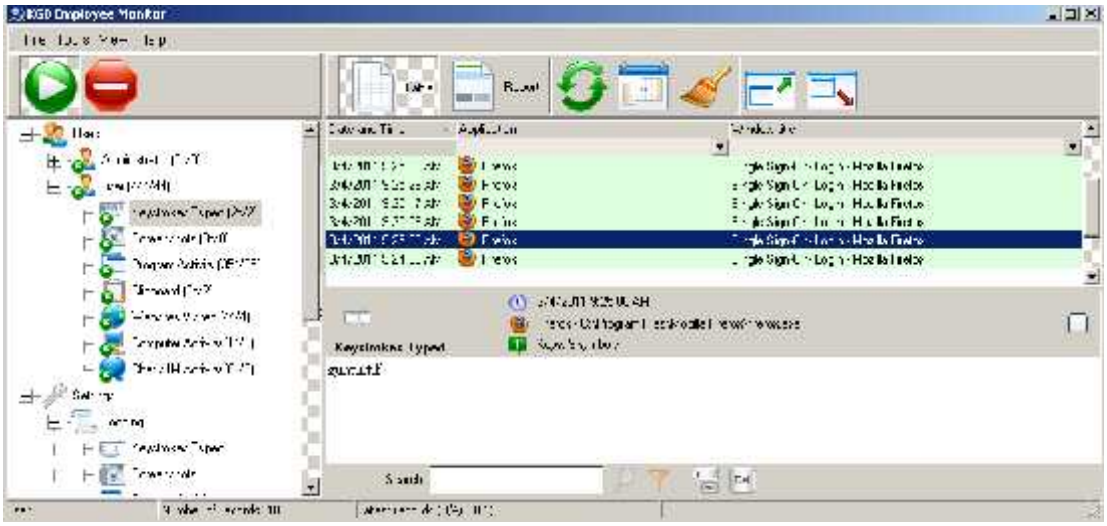
Gambar 5.9 Pengujian Sistem Tanpa *Single Sign-On* dengan *Keylogger*

Dari gambar 5.9 dapat terlihat bahwa autentikasi berbasis teks yaitu *username* dan *password* ter-*capture* oleh aplikasi *Keylogger.*

2. **Pengujian Sistem *Single Sign-On* dengan aplikasi *keylogger* KGB *Employee Monitor***



Gambar 5.10 Pengujian dengan Aplikasi *Keylogger* 1



Gambar 5.11 Pengujian dengan Aplikasi *Keylogger* 2

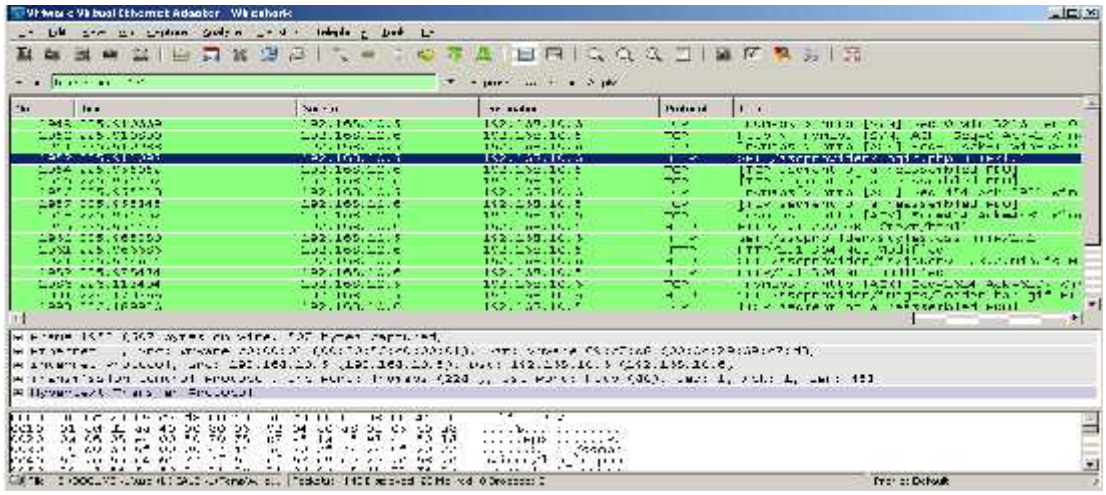
Pada pengujian gambar 5.10 dan gambar 5.11 dilakukan *login* dengan mengetikkan *username, password* serta gambar, dengan menggunakan aplikasi *keylogger* tersebut menunjukkan bahwa gambar tidak ter-*capture* oleh aplikasi *keylogger.*

3. Pengujian Sistem Tanpa *Single Sign-On* dengan Aplikasi *Wireshark*

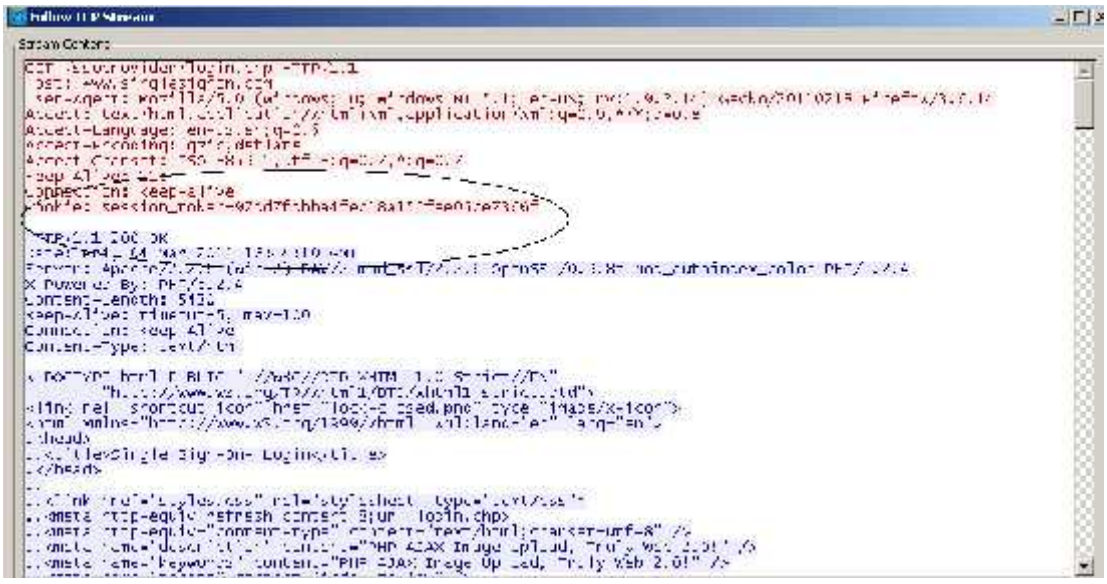
Gambar 5.12 Pengujian Sistem Tanpa *Single Sign-On* dengan *Wireshark*

Dari gambar 5.12 dapat terlihat bahwa autentikasi berbasis teks yaitu *username* dan *password* ter-*capture* oleh aplikasi *Wireshark.*

4. Pengujian *Single Sign-On* dengan aplikasi *Wireshark*



Gambar 5.13 Pengujian dengan Aplikasi *Wireshark*



Gambar 5.14 *Capture* Sistem *Single Sign-on* dengan *Wireshark*

Pada pengujian gambar 5.14, *member* mengakses sistem *single sign-on server* dan melakukan *login* dengan memasukkan *username, password* serta gambar, dengan

menggunakan aplikasi tersebut menunjukkan bahwa *username, password* dan gambar tidak ter-*capture* oleh aplikasi *wireshark* .

5.2.5 Kesimpulan Pengujian

Secara umum, hasil pengujian yang diperoleh adalah sebagai berikut :

1. Sistem *Single Sign-On* yang telah dibangun dapat berjalan dengan baik, seorang *member* tidak perlu *login* untuk kedua kalinya, *member* cukup sekali *login.*
2. Pengujian berdasarkan prosedur sistem *single sign-on* telah berhasil diujikan.
3. Autentikasi dengan menggunakan gambar tidak ter-*capture* oleh aplikasi *keylogger*, sehingga dapat menambah keamanan sistem *single sign-on.*
4. Seluruh aplikasi yang di implementasikan dapat berjalan.

BAB VI
PENUTUP

Pada bab ini akan dipaparkan kesimpulan yang didapat dari pelaksanaan Tugas Akhir. Selain itu, disampaikan beberapa saran yang berguna untuk kelanjutan pengembangan topik yang diambil.

6.1 Kesimpulan

Beberapa hal yang dapat disimpulkan dari pelaksanaan tugas akhir ini adalah :

1. Berhasil membangun sebuah sistem *single sign-on* berbasis *web*, dimana seorang *user* hanya sekali memasukkan *credential* untuk beberapa situs *web* tanpa harus memasukkan *username* dan *password* untuk kedua kalinya.
2. Autentikasi *login* menggunakan gambar merupakan salah satu solusi untuk keamanan sistem *login* terutama sistem *single sign-on,* karena sesuai dengan pengujian, *file* gambar tidak ter-*capture* oleh aplikasi *keylogger*.
3. Steganografi *Least Significant Bit* dapat diimplementasikan dengan baik untuk melakukan pengamanan gambar yang digunakan sebagai autentikasi *login*.
4. Sistem autentikasi menggunakan gambar pada aplikasi *single sign-on* hanya digunakan pada aspek *non-repudiation*.
5. Sistem autentikasi gambar menggunakan format PNG.
6. SSL diperlukan oleh sistem *single sign-on* dalam melakukan enkripsi *session.* Sehingga dapat meminimalisir terjadinya proses pencurian *password member.*

6.2 Saran

Saran-saran yang berkaitan dengan pelaksanaan tugas akhir ini adalah :

1. Aplikasi sistem *Single Sign-On* ini perlu di uji pada kasus yang lebih besar dan lingkungan yang bervariasi agar dapat diketahui kinerja aplikasi. Pada Tugas Akhir ini, sistem *Single Sign-On* hanya diuji pada jaringan *virtual* atau jaringan komputer lokal.

2. *Database* yang digunakan dalam aplikasi ini masih terlalu sederhana jika dibandingkan dengan aplikasi yang lain. Sehingga perlu dilakukan konfigurasi lebih lanjut untuk menghadapi masalah ini.
3. Untuk meningkatkan keamanan gambar sebagai autentikasi, dapat menggunakan teknik keamanan gambar yang lain.

DAFTAR PUSTAKA

Alkhatib Ghazi, dan Rine David, " *Integrated Approaches in Information Technology and Web Engineering: Advancing Organizational Knowledge Sharing*", IGI Global, 2009.

Ardagna Claudio Agostino, Frati Fulvio, Gianini Gabriele, "*Open Source in Web-Based Applications : A Case Study on Single Sign-On*". Chapter VI, IGI Global, 2009.

Ariyus Dony, "*Computer Security*", Penerbit Andi Yogyakarta. 2006.

Bettini, C., Jajodia, S., Sean Wang, X., & Wijesekera, D. *Provisions and obligations in policy management and security applications*. *In Proceedings of the 28th VLDB Conference*, HongKong, China. 2002.

Bulger Brad, Greenspan Jay, and Wall David, "*MySQL/PHP Database Applications, Second Edition*", Wiley Publishing, Inc., Indianapolis, Indiana. 2004.

Hursti Jani, " *Single Sign-On*.", *Department of Computer Science Helsinki University of Technology,* 1997.

Nitin, Sehgal Kumar Vivek, Chauhan Sigh Durg, Sood Munish, and Hastir Vikas, "*Image Based Authentication System with Sign-In Seal*" *Proceedings of the World Congress on Engineering and Computer Science,* 2008.

Pashalidis and C. J. Mitchell, "*A Taxonomy of Single Sign-On Systems, in Information Security and Privacy, 8th Australian Conference, ACISP 2003, Wollong, Australia, July 9-11, 2003, Proceedings*, ser. *Lectures Notes In Computer Science*, R. Safavi-Naini and J. Seberry, Eds., vol 2727. Springer-verlag, pp 249-264. *July* 2003

Kanda, " *Integrasi Single Sign-On OpenID pada Website berbasis PHP"*, [Online] Available http://www.kandar.info, diakses 10 Januari 2011

Shawn Riley, "*Why You Should Consider a Single Sign On Product in Health Care"*, [Online] Available http://www.healthtechnica.com, diakses 09 Desember 2010

_____, "*Single Sign-On",* [Online] Available http://www.id.wikipedia.org/, diakses 20 November 2010