BAB II
LANDASAN TEORI

2.1 *Image Based Authentication*

Dalam banyak hal yang kita jumpai, seringkali seorang pengguna diminta untuk mengidentifikasi diri mereka untuk mengakses suatu sistem atau layanan. Identifikasi adalah dimana tahap seorang pengguna menunjukkan bukti-bukti identitas nya. Bukti tersebut dapat berupa ID pengguna atau nomor *PIN* (*Personal Identification Number*). Setelah pengguna memasukkan suatu informasi, sistem akan masuk ke tahap otentikasi. Otentikasi adalah proses usaha pengecekan identitas seorang pengguna ketika mengakses ke dalam sebuah sistem (Pashalidis, 2005). Pengguna yang telah lolos pengecekan identitas adalah pengguna resmi pada sistem, orang yang memiliki otoritas atas sistem. Otorisasi adalah proses pengecekan bahwa pengguna yang dikenal memiliki kekuasaan untuk melakukan tindakan tertentu (Todorov, 2007).

Otentikasi dapat dikategorikan menjadi dua, yaitu *Text Authentication* (otentikasi teks) dan *Image based Authentication* (otentikasi berupa gambar). Lalu otentikasi gambar dapat dikategorikan menjadi dua teknik: *recall-based techniquess* and *recognition-based technicues* (Suo, Owen and Zhu, 2005; Wiedenbeck *et al.*, 2005).

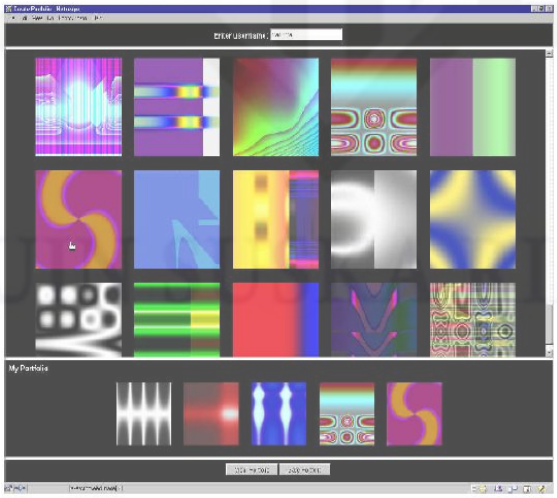
2.1.1 Recognition-based Techniques

Pada teknik *recognition based techniques* ini dengan memanfaatkan kemampuan mengingat manusia untuk mengenali sesuatu yang telah dilihat sebelumnya untuk otentikasi pengguna (Dunphy, 2012). Pengguna akan ditampilkan beberapa gambar, ikon atau simbol ketika registrasi dan pengguna melewati tahap otentikasinya dengan mengenali atau mengidentifikasi pilihan mereka yang sesuai proses registrasi. Pada saat registrasi, pengguna setidaknya mengetahui *k password* gambarnya. Dan ketika otentikasi, pengguna diharuskan melakukan pencarian visual untuk mengenali *password* gambarnya di antara *d*

gambar perangkap lainnya yang disediakan sebagai tantangan otentikasi (Elftmann, 2006; Towhidi and Masrom, 2009) .

Contoh skema yang berdasarkan teknik ini adalah Passcode penelitian yang dilakukan oleh Jansen dan sistem Dejavu yang didesain oleh Rachna Dhamija

Gambar 2. 1 Passcode (Jansen, 2007)

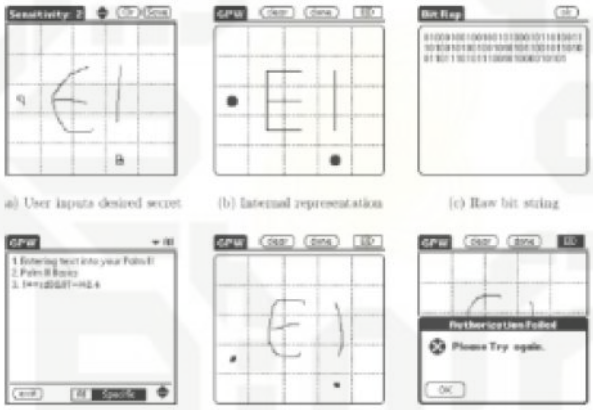


Gambar 2. 2 Dejavu (Perrig and Dharmija, 2000)

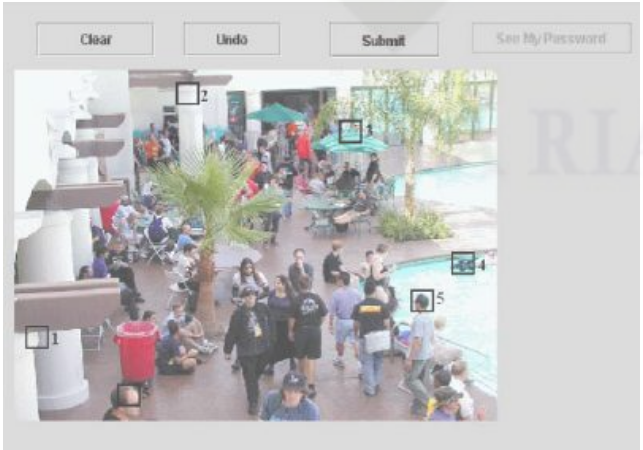
2.1.2 Recall-based Techniques

Penggunaan *recall-based techniques* lebih di titik beratkan kepada daya ingat seorang pengguna untuk kembali menciptakan sesuatu yang telah dia buat (*pure recall*) atau memilih yang telah dipilih pada proses registrasi (*cued recall*) (Dunphy, 2012).

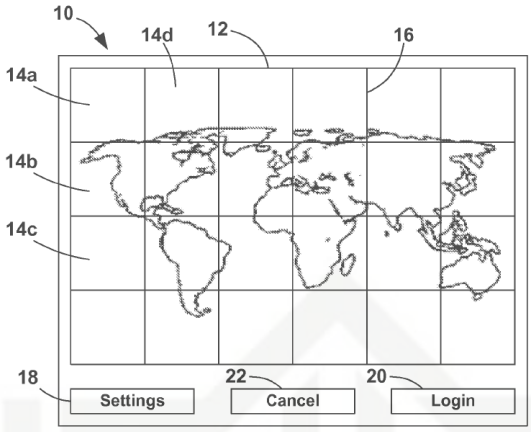
Contoh yang menggunakan teknik ini adalah penelitian yang dilakukan oleh Jozsef Patvarczki et al, Draw a Secret yang didesain oleh Jermyn et al. , dan Passpoint oleh Susan Wiedenbect et al



Gambar 2. 3 Draw a Secret (Jermyn *et al.*, 1999)



Gambar 2. 4 Passpoint (Wiedenbeck *et al.*, 2005)



Gambar 2. 5 Graphical *Password* Jozsef (Patvarczki, Kornafeld and Tamas, 2012)

2.2 Macam – macam Format Gambar

Format *file* dari gambar ada beberapa macam, berikut adalah penjelasannya yang dapat dilihat pada tabel 2.1 (Kadir and Susanto, 2012).

Tabel 2. 1 Deskripsi Format Gambar

Format Gambar	Ekstensi	Keterangan
TIFF	.tif , .tiff	*Tagged Image File Format* merupakan format citra yang mula-mula dibuat oleh Aldus. Kemudian dikembangkan oleh *Microsoft* dan *Adobe.*
JPEG	.jpg , .jpeg	*Joint Photographic Expert Group* adalah format gambar yang dirancang agar dapat memapatkan data dengan rasio 1:16.
GIF	.gif	*Graphic Interface Format* merupakan format yang memungkinkan pemampatan data hingga 50%. Format ini mampu menyimpan gambar dua dimensi

Format Gambar	Ekstensi	Keterangan
		sehingga banyak digunakan untuk keperluan *web*, multimedia, dan elektronik.
BMP	.bmp	*Windows Bitmap* merupakan format grafis yang dapat menyimpan 1 – 24 bit pada *platformwindows*.
PNG	.png	*Portable Network Grahics* mendukung pemampatan data tanpa menghilangkan infomasi aslinya. Format ini mampu menyimpan file dalam *bitdepth* hingga 24 *bit* serta menghasilkan *background* yang transparan dan pinggiran yang halus.
XWD	.xwd	*XwindowDump*

2.3 *Single Sign-On*

Teknologi *Single Sign-On* (*SSO*) adalah teknik dimana seorang pengguna mengotentikasi dirinya hanya sekali saja untuk mendapatkan izin akses terhadap semua layanan yang terdapat di dalam jaringan (Pashalidis and Mitchell, 2005). *SSO* menghindari login ganda dengan mengidentifikasi pengguna secara ketat dan memperkenankan informasi otentikasi untuk digunakan dalam sistem atau kelompok sistem yang terpercaya.

Keuntungan yang diperoleh dari *SSO* (Guelph, 2008; Cakir, 2013) antara lain :

1. Mengurangi *password fatigue* (kejenuhan terhadap *password*) berupa *username* dan *password* yang berbeda-beda.

2. Mengurangi waktu yang terbuang karena memasukkan *password* yang sama berulang kali.
3. Mengurangi IT *Costs* yang berkaitan dengan banyaknya pertanyaan mengenai *password*.
4. Keamanan pada semua level masuk, keluar, dan akses terhadap sistem tanpa meminta ulang *password*.

2.3.1 Kategori *Single Sign On*

Single Sign-On dapat dikategorikan ke dalam 2 kategori (Pashalidis, 2005), yaitu :

1. *Single Sign*-On for *Authentication*

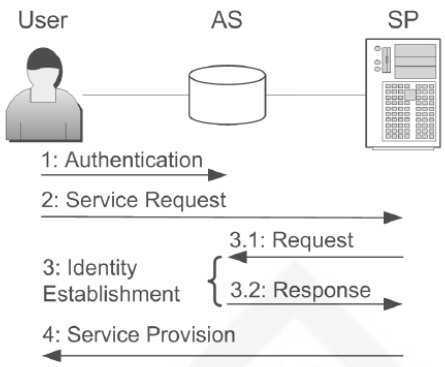
Sistem *SSO* berbasis otentikasi ini dimana *SSO* servernya hanya memberikan *service* apakah pengguna A telah terotentikasi apa belum, *SSO* server tidak melakukan proses otorisasi pada pengguna yang telah aktif tersebut. Proses otorisasi dilakukan pada tiap aplikasi nya.

2. *Single Sign-On for Authorization*

SSO Server pada sistem *SSO* berbasis otorisasi ini akan terus mengambil alih kendali pengguna yang telah terotentikasi tersebut.

2.3.2 Cara Kerja *Single Sign On*

Hampir semua perancangan *Single Sign-On* yang sudah ada menggunakan otentikasi sesi. Aliran data informasi dapat dilihat pada gambar 2.6. Awalnya untuk memulai sesi, pengguna akan memasukkan data identifikasi diri nya ke *Authentication Service* (*AS*). Ketika data yang dimasukkan telah benar dan diterima, barulah sesi dijalankan (langkah 1) (Pashalidis and Mitchell, 2005)..



Gambar 2.6 Jalannya informai pada skema *SSO* (Pashalidis, 2005)

Pada gambar 2.6 selama sesi belum berakhir, ketika seorang pengguna akan mengakses satu *Service Provider* (*SP*) lain, maka *AS* akan langsung secara otomatis telah masuk ke dalam *SP* yang diminta pengguna tersebut. Suatu waktu ketika pengguna menghentikan penggunaan layanan, maka sesi juga akan diakhiris. Pengguna yang akan menggunakan layanan harus melakukan identifikasi seperti awal mengali satu sesi baru.

2.3.3 Protokol Single Sign On

1. ***SAML* (*Security Assertion Markup Language*)**

Security Assertion Markup Language diciptakan oleh OASIS *Security Service Technical Committee* dan versi terbaru nya adalah *SAML* 2.0 pada tahun 2005 (OASIS, 2005). SAML adalah sebuah *framework XML*(*Extensible Markup Language*) untuk identitas yang portabel sebagai salah satu token keamanan yang dikenali oleh *WS-Security*. *SAML* 2.0 menjadi sebuah standar dalam pertukaran otentikasi dan otorisasi berbasis pada protokol *XML* dan menggunakan *security* token yang berisi *assertions* untuk melewati informasi antar otoritas *SAML*. *SAML* 2.0 memungkinkan skenario otentikasi dan otorisasi berbasis *cross domain single sign-on* (Raharjo, 2013).

2. ***OAuth* (*Open Authorization*)**

Sejak dipublikasikan pada tahun 2012 *OAuth* 2.0 menjadi salah satu *framework* yang banyak digunakan. *OAuth* menggunakan *Access Token* sebagai

mekanisme untuk izin akses pada sumber daya. Otentikasi ini dapat langsung otomatis karena sebuah *access token* mengindikasikan pengguna yang telah benar (Heijmink, 2015)

.

3. CAS (Central Authentication Service)

Central Authentication Service merupakan sebuah *framework* untuk *Single Sign-On* yang dibuat dengan menggunakan bahasa *Java*. Cara kerja *CAS* menggunakan *Ticket Granting* dimana ketika pengguna melakukan *login*, maka akan diberikan tiket yang tersimpan di dalam *cookies* dan nantinya digunakan untuk melakukan otentikasi pada setiap aplikasi (Raharjo, 2013).

4. Open Single Sign-On

Open Single Sign-On (*Open SSO*) adalah sebuah infrastruktur yang mendukung layanan berbasis identitas (*Identitity Management*) yang dikembangkan oleh Sun, dan implementasi solusi dari *SSO* transparan sebagai komponen keamanan dalam infrastruktur jaringan (Cakir, 2013).

2.4 *Open Authorization (OAuth)*

Open Authorization (*OAuth*) merupakan sebuah protokol yang memungkinkan aplikasi pihak ketiga memperoleh akses terbatas terhadap layanan HTTP, baik atas nama pemilik *resource* dengan membuat sebuah kesepakatan interaksi antara pemilik *resource* dan layanan HTTP, atau dengan membiarkan aplikasi pihak ketiga untuk memperoleh akses atas nama sendiri (Kaur and Aggarwal, 2013).

OAuth 1.0 (dikenal sebagai RFC 5849), diterbitkan pada tanggal 4 Desember 2007, direvisi pada tanggal 24 Juni 2009, dan diselesaikan pada bulan April 2010 yang memberikan pengaruh penting pada perkembangan keamanan web *API* (*Application Programming Interface*) tanpa harus pengguna berbagi *username* dan *password* mereka. Adapun pencipta dan pengagas utama dari otentikasi *OAuth* berbasis API ini adalah E.Hammer-Lahav, Ed (Heijmink, 2015).

OAuth 2.0 adalah *project* lanjutan dari protokol *OAuth 1.0*. *OAuth* 2.0 lebih menekankan pada kemudahan *client* sebagai pemilik dan pengembang aplikasi dengan memberikan otorisasi khusus di berbagai aplikasi. *OAuth* berada dalam pengembangan IETF *OAuth* WG dan didasarkan pada usulan WRAP *OAuth*. *WRAP* (*Web Resource Authorization Protocol*) adalah profil *OAuth* yang memiliki sejumlah kemampuan penting yang tidak tersedia di versi *OAuth* sebelumnya. Spesifikasi terbaru dari *OAuth* disumbangkan kepada IETF *OAuth* WG dan merupakan dasar dari terciptanya *OAuth* versi 2.0 (Heijmink, 2015) .

Protokol *OAuth* 2.0 mengarahkan proses otorisasi yang terbagi dan memisahkan peran dari *client* (pengguna) dari *resource owner*. Pada *OAuth*, pengguna meminta akses ke *resource owner* dan diatur pada *resource server*, dan mengeluarkan *credential* yang berbeda sesuai *resource owner* nya. Pengguna memperoleh *Access Token* yang merupakan pengguna pihak ketiga yang terotorisasi server dengan persetujuan *resource owner*. *Access Token* digunakan untuk mengakses *protected resource* pada *resource server* (Kaur and Aggarwal, 2013).

2.5 *Geometric Dimensioning and Tolerancing*

Dimensioning geometris dan *tolerancing* (GD & T) adalah sistem untuk mendefinisikan dan mengkomunikasikan *engineering tolerances*. *True Position* didefinisikan sebagai variasi total yang diperbolehkan dimana sebuah fitur dapat memiliki dari posisi "benar"nya. Ini mungkin berlaku untuk segala sesuatu dari titik sampai sumbu ke bidang untuk seluruh fitur (Geiss and Derr, 2015) .

True Position dari fitur yang dibuat dengan terlebih dahulu menentukan titik saat ini yang direferensikan dan kemudian membandingkan bahwa ke semua permukaan untuk menentukan seberapa jauh pusat dari fitur ini berada. Hal ini disederhanakan seperti toleransi dimensi tetapi dapat diterapkan untuk zona toleransi diameter daripada koordinat X-Y sederhana (Geiss and Derr, 2015)

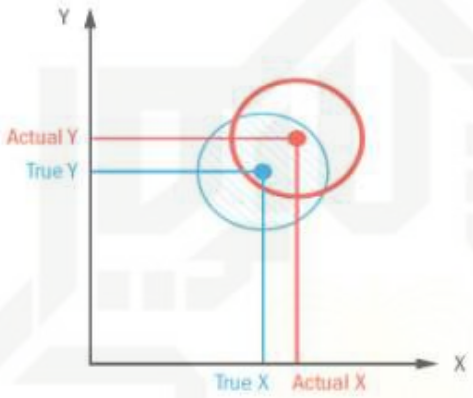
Circular Tolerance memiliki persamaan sebagai berikut:

$$\sqrt{(x - x')^2 + (y - y')^2} \leq r \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

x : actual X x' : true X

y : actual Y y' : true Y

r : toleransi



Gambar 2. 7 Persamaan *CircularityTolerance* (Geiss and Derr, 2015)

2.6 Pengujian *User Acceptance Testing*

Pengujian *User Acceptance Testing* (UAT) merupakan proses verifikasi bahwa solusi yang dibuat dalam sistem sudah sesuai untuk pengguna. Proses ini berbeda dengan pengujian sistem (memastikan software tidak crash dan sesuai dengan dokumen permintaan pengguna), melainkan memastikan bahwa solusi dalam sistem tersebut akan bekerja untuk pengguna (yaitu, tes bahwa pengguna menerima solusi di dalam sistem) (Hamblerg and Goethem, 2013) .

User Acceptance Testing umumnya dilakukan oleh pengguna akhir, biasanya tidak fokus pada identifikasi masalah sederhana seperti kesalahan ejaan, maupun di cacat seperti *crash* perangkat lunak. Penguji dan pengembang mengidentifikasi dan memperbaiki masalah ini selama tahap awal pengujian fungsionalitas, pengujian saat integrasi dan pada tahap sistem testing.

Pengujian *User Acceptance Testing* dengan memberikan kuisioner pernyataan kepada responden sesuai standar yang telah dilakukan oleh Eljetlawi dalam hal pembuatan otentikasi gambar (Eljetlawi, 2010; Zangooei, Welch and

Mansoori, 2012). Dimana atributnya adalah Easy to Use, Easy to Create, Easy to Memorize, Easy to Learn, dan Design Layout.

2.7 *Cronbachs Alpha*

Cronbachs Alpha adalah koefisiensi *alpha* yang dikembangkan oleh Cronbach pada tahunn 1951 sebagai ukuran umum dari konsistensi internal skala multiitem untuk mengukur keandalan indikator-indikator yang digunakan dalam kuesioner penelitian. Koefisien *Alpha* memiliki nilai berkisar dari nol sampai satu (Jr. *et al.*, 2011).

Rumus untuk menghitung koefisien reliabilitas instrument dengan menggunakan *Cronbachs Alpha* adalah sebagai berikut :

$$\mathrm{r} = \left[\frac{k}{(k-1)}\right] \times \left[1 - \frac{\sum \sigma_b^2}{\sigma_t^2}\right] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

Keterangan :

r = koefisien reliabilitas instrument (*Cronbachs Alpha)*

k = banyak bulir pertanyaan atau soal

$\sum \sigma_b^2$ = total varians bulir

σ_t^2 = total varians

Nilai tingkat keandalan *Cronbach's Alpha* dapat ditunjukan pada tabel 2.2 berikut ini.

Tabel 2. 2 Tingkat keandalan Cronbachs Alpha (Jr. *et al.*, 2011)

Nilai Cronbachs Alpha	**Tingkat Keandalan**
0.0 – 0.20	Kurang Handal
>0.20 – 0.40	Agak Handal
>0.40 – 0.60	Cukup Handal
>0.60 – 0.80	Handal

2.8 Macam- macam Serangan

Keamanan dari sebuah sistem otentikasi secara langsung berhubungan dengan tingkat kesulitan dari sulitnya sebuah *password* dibobol. Ada beberapa penyerangan dalam otentikasi, seperti dijelakan di bawah ini :

2.8.1 Keylogger

Keylogger adalah sebuah perangkat baik perangkat keras atau perangkat lunak yang digunakan untuk memantau penekanan tombol keyboard. Sebuah keylogger biasanya akan menyimpan hasil pemantauan penekanan tombol keyboard tersebut ke dalam sebuah berkas log/catatan/rekaman (Dunphy, 2012). Penyerangan seperti rekaman seperti ini menjadi salah satu ancaman bagi sebuah sistem. Contoh aplikasi Keylogger adalah

1. Familly Keylogger
2. KGB Keylogger
3. AK Monitor
3. BlazingTools Perfect Keylogger
4. Emissary Keylogger,

2.8.2 Sniffing Password

Sniffing adalah penyadapan terhadap lalu lintas data pada suatu jaringan komputer. Tindakan penyadapan yang dilakukan dalam jaringan dengan tujuan untuk dapat mencuri data-data pribadi ataupun account lain yang bersifat pribadi. Karena data yang mengalir pada suatu jaringan bersifat bolak-balik, maka dengan proses sniffing ini dapat menangkap paket yang dikirimkan dan terkadang menguraikan isi dari RFC (Request for Comments) (Elftmann, 2006; Towhidi and Masrom, 2009).. Pada saat ini, tindakan *sniffing* seringkali dilakukan dengan menggunakan bantuan *software Sniffer*. Beberapa contoh *software* yang terkenal misalnya : Cain & Abel, Ethereal, Wireshark, Tcpdump, Ettercap, Dsniff, Etherpeak, Airopeak, dll.

2.8.3 Brute Force Attack

Serangan *Brute Force* adalah sebuah ancaman penyusupan *password* dengan mencoba semua kemungkinan kombinasi *password* (Elftmann, 2006). Semakin kompleks sebuah *password* semakin aman dari serangan ini. Kemampuan dari sebuah brute force attack tergantung dari panjangnya cipher yang ingin dipecahkan , dan jumlah komputasi yang tersedia untuk penyerang. Contoh program yang menerapkan konsep ini adalah Cain and Abel, Brutus, Hydra, LastBit, Jhon the Ripper dan lain lain.

2.8.4 Shoulder Surfing Attack

Serangan Shoulder Surfing adalah sebuah ancaman yang sangat rentan dengan otentikasi gambar. Seorang penyerang tidak harus memiliki pengetahuan teknis untuk mengamati password pengguna tertentu. Sejak otentikasi menggunakan gambar untuk pengganti password-nya, mereka menyediakan ruang yang lebih besar di layar daripada teks, itu membuat pengamatan lebih mudah untuk penyerang (Towhidi and Masrom, 2009).

2.9 Penelitian Terkait

Penelitian yang dilakukan oleh Asraful dan Imam adalah melakukan kombinasi antara Recall dan Recognition Based Techniques (Haque and Imam, 2014). Ide ini menarik bagi penulis, dan mencoba mengembangkannya dengan output berbeda dan menerapkan pada sistem Single Sign On.

Penelitian *Single Sign-On* menggunakan *OAuth* 2.0 dilakukan oleh Priyo dan Yosrinal. Perbedaannya dengan penelitian ini membahas tentang merancang sebuah otentikasi gambar dan mengimplementasikan ke dalam sistem *SSO* menggunakan protokol *OAuth 2.0*, sedangkan dari penelitian yang sudah dilakukan membahas masalah *SSO* menggunakan protokol *OAuth 2.0* untuk layanan internet *proxy* (Nugroho, 2012) dan otentikasi *Oauth* pada aplikasi layanan *cafe online* (Yosrinal, 2011) dan tentunya memiliki hasil keluaran yang berbeda dengan penelitian ini.

Pada tugas akhir yang dilakukan oleh Guntoro dengan memanfaatkan otentikasi gambar pada sistem *Single Sign-On* yang dirancang nya berhasil. Pada tahap otentikasinya, pengguna diminta untuk memasukkan *username, password,* dan *upload* gambar sebagai kunci dari pengguna. Gambar yang di *upload* adalah gambar yang sudah disisipi kata rahasia menggunakan teknik steganografi dengan metode *Least Significant Bit* (Guntoro, 2011). Kelemahan pada sistem ini adalah pengguna diharuskan mengunggah gambar sebelum *login* yang diartikan gambar tersebut harus dibawa kemana-mana ketika akan login.

Takada dan Koike melakukan penelitian tentang teknik *password* grafis pada telepon genggam. Teknik ini mengizinkan pengguna untuk menggunakan gambar kesukaan mereka untuk otentikasi. Pengguna pertama mendaftarkan gambar mereka (memasukkan *password*) ke dalam sistem. Pada tahap identifikasi, pengguna harus melewati beberapa ronde verifikasi. Pada setiap ronde nya, pengguna memilih gambar *password* mereka diantara gambar tipuan yang lain atau tidak memilih apapun jika gambar *password* tidak ditampilkan. Sistem akan mengotorisasi pengguna yang berhasil melakukan semua tahap verifikasi. Pengguna diizinkan untuk mendaftarkan gambar mereka sendiri untuk mempermudah pengguna untuk mengingat gambar *password* nya (Takada and Koike, 2003).

Pada penelitian Rachna Dhamija diajukan satu perancangan otentikasi grafis berdasarkan teknik *Hash Visualization* (Perrig *et al.*, 1999). Pada sistem ini, pengguna diminta untuk memilih sejumlah gambar abstrak yang telah diatur acak oleh program. Kemudian, pengguna perlu untuk mengidentifikasi gambar yang telah dipilih sebelumnya untuk otentikasi dirinya. Hasil penelitian menunjukkan 90% partisipan sukses dalam otentikasi menggunakan teknik ini, sedangkan hanya 70% yang sukses menggunakan *password* berupa teks atau PIN (Perrig and Dharmija, 2000). Waktu yang diperlukan sedikit lebih lama dan cenderung membosankan menjadi kelemahan pada sistem ini.

Perancangan sistem *graphical password* pertama kali diperkenalkan oleh Blonder yang menampilkan sebuah gambar grafis penuh, lalu pengguna diminta

untuk memilih daerah / area pada gambar yang merupakan data identifikasi diri nya untuk tahap otentikasi (Blonder, 1996). Pada penelitiannya yang dilakukan Passlogix, pengguna diminta memilih beberapa urutan item yang ada pada satu gambar grafis yang berarti memasukkan *password* nya (Paulson, 2002). Tapi rincian cara yang digunakan untuk teknik ini tidak tersedia.