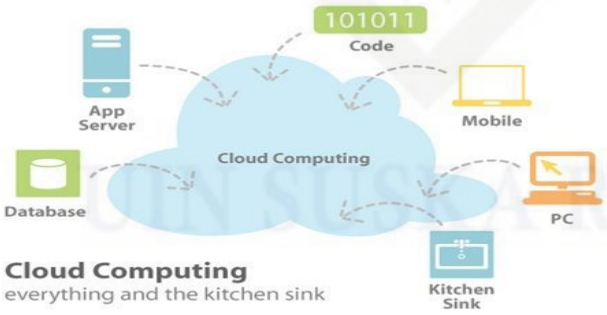
BAB II
LANDASAN TEORI

2.1 *Cloud Computing*

Teknologi *cloud computing* dihadirkan sebagai upaya untuk memungkinkan akses sumber daya dan aplikasi dari mana saja melalui jaringan internet. *Cloud computing* menggunakan konsep virtualisasi yang semuanya dapat diakses melalui internet sehingga dapat mengurangi biaya teknologi informasi dan menyederhanakan pengelolaan layanan teknologi informasi, dan layanan yang diberikan bersifat *multi-tenant* sehingga sumber daya komputasi dapat digunakan secara bersama-sama dan disesuaikan dengan kebutuhan pengguna (Haryani, Nugroho, 2015). *Cloud computing* merupakan sebuah model komputasi dengan skalabilitas yang tinggi dan memungkinkan penggunaanya untuk menggunakan sumber daya (*network, server, storage, applications,* dan *services*) yang ada dalam jaringan *cloud* sehingga dapat dibagi dan digunakan bersama. Secara sederhana dapat dikatakan bahwa para pengguna komputer bisa menggunakan *resource* tanpa perlu membeli, memiliki atau menginstal program, namun cukup dengan menyewa sumber daya dari server inti sesuai dengan kebutuhan yang diperlukan.



Gambar 2.1 *Cloud Computing* (NIST,1995)

Awan (*cloud*) adalah metafora dari internet, sebagaimana awan yang sering digambarkan di diagram jaringan komputer, *cloud* dalam *cloud computing* juga merupakan abstraksi dari infrastuktur kompleks yang disembunyikan, yang

merupakan suatu metode komputasi dimana kapabilitas terkait teknologi informasi yang disajikan sebagai suatu layanan (*as a service*) sehingga pengguna dapat mengaksesnya melalui internet (Kusnandar, 2012). Virtualisasi adalah tahap awal menuju *cloud computing.* Virtualisasi "melepaskan" ikatan antara *hardware* dan fungsi kendali manajemennya, sehingga menjadi *virtual machine* (Hakim, 2016).

2.1.1 Karakteristik *Cloud Computing*

Ada lima karakteristik penting dari *cloud computing*, yaitu (Sakurai, 2011):

1. *On-demand self service.* Konsumen dapat menentukan kemampuan komputasi secara sepihak, seperti *server time* dan *network storage*, secara otomatis sesuai kebutuhan tanpa memerlukan interaksi manusia dengan masing-masing penyedia layanan.
2. *Broad network access.* Kemampuan yang tersedia melalui jaringan dan diakses melalui mekanisme standar yang mengenalkan penggunaan berbagai *platform*.
3. *Resource Pooling.* Penyatuan sumberdaya komputasi yang dimiliki penyedia untuk melayani beberapa konsumen virtual yang berbeda, ditetapkan secara dinamis dan ditugaskan sesuai dengan permintaan konsumen.
4. *Rapid Elasticity.* Kemampuan dapat ditetapkan dan dirilis secara elastis dalam beberapa kasus dilakukan secara otomatis untuk menghitung keluar dan masuk dengan cepat sesuai dengan permintaan. Untuk konsumen, kemampuan yang tersedia yang sering kali tidak terbatas dan kuantitasnya dapat disesuaikan setiap saat.
5. *Meausred Service.* Sistem *cloud computing* secara otomatis mengawasi dan mengoptimalkan penggunaan sumber daya dengan memanfaatkan kemampuan pengukuran (*metering*) pada beberapa tingkat yang sesuai dengan jenis layanan (penyimpanan, pemrosesan, *bandwidth*, dan *account* pengguna aktif). Penggunaan sumber daya dapat dipantau, dikendalikan, dan dilaporkan sebagai upaya memberikan transparansi bagi penyedia dan konsumen dari layanan yang digunakan.

2.1.2 Model Layanan Cloud Computing

Terdapat tiga model layanan dari *cloud computing* (Sakurai, 2011) yaitu :

1. *Cloud Software as a Service (SaaS).* Kemampuan yang diberikan kepada konsumen untuk menggunakan aplikasi penyedia dapat beroperasi pada infrastruktur *cloud.* Aplikasi dapat diakses dari berbagai perangkat klien melalui antarmuka seperti *web browser* (contohnya, email berbasis web). Konsumen tidak mengelola atau mengendalikan infrastruktur cloud yang mendasar termasuk jaringan,server, sistem, operasi, penyimpanan, atau bahkan kemampuan aplikasi individu, dengan kemungkinan pengecualian terbatas terhadap pengaturan konfigurasi aplikasi pengguna tertentu.
2. *Cloud Platfrom as a Service (PaaS).* Kemampuan yang diberikan kepada konsumen untuk menyebarkan aplikasi yang dibuat konsumen atau diperoleh ke infrastuktur *cloud computing* menggunakan bahasa pemograman dan peralatan yang didukung oleh *provider*. Konsumen tidak mengelola atau mengendalikan infrastuktur *cloud* mendasar termasuk jaringan, *server*, sistem operasi, atau penyimpanan, namun memiliki kontrol atas aplikasi yang disebarkan dan memungkinkan aplikasi melakukan *hosting* konfigurasi.
3. *Cloud Infrastucture as a Service (IaaS)*. Kemampuan yang diberikan kepada konsumen untuk memproses, menyimpan, berjaringan, dan sumber komputasi penting yang lain. Dimana konsumen dapat menyebarkan dan menjalankan perangkat lunak secara bebas, yang dapat mencakup sistem operasian aplikasi. Konsumen tidak mengelola atau mengendalikan infrastruktur *cloud* yang mendasar tetapi memiliki kontrol atas sistem operasi, penyimpanan, aplikasi yang disebarkan, dan mungkin kontrol terbatas komponen jaringan yang dipilih.

2.1.3 Model Penyebaran *Cloud Computing*

Terdapat empat model penyebaran *cloud computing* (Sakurai, 2011) yaitu :

1. *Private cloud.* Infrastuktur *cloud* yang semata-mata dioperasikan bagi suatu organisasi. Ini mungkin dimiliki, dikelola dan dijalankan oleh suatu organisasi, pihak ketiga atau kombinasi dari beberapa pihak dan mungkin ada pada *on premis* atay *off premis.*
2. *Community cloud.* Infrastruktur *cloud* digunakan secara bersama oleh beberapa organisasi dan mendukung komunitas tertentu yang telah berbagi *concerns.*

Kemungkinan dikelola oleh organisasi atau pihak ketiga dan mungkin ada pada *on premis* atau *off premis.*

3. *Public cloud.* Infrastuktur *cloud* yang disediakan untuk umum atau kelompok industri besar dan dimiliki oleh sebuah organisasi yang menjual layanan *cloud*.
4. *Hybrid cloud.* Infrastruktur *cloud* merupakan komposisi dari dua atau lebih *cloud* yang masih entitas namun terikat bersama oleh standar atau kepemilikan teknologi yang menggunakan data dan portabilitas aplikasi.

2.2 Keamanan Informasi

Keamanan informasi merupakan suatu upaya untuk mengamankan aset informasi terhadap ancaman yang mungkin timbul. Sehingga keamanan informasi secara tidak langsung dapat menjamin kontinuitas bisnis, mengurangi resiko-resiko yang terjadi, mengoptimalkan pengembalian investasi (return on invesment).

Informasi atau data adalah aset bagi suatu perusahaan maupun instansi. Semakin banyak informasi yang disimpan, dikelola dan di s*haring* makan semakin besar pula resiko terjadinya kerusakan, kehilangan, atau terekspos nya data ke pihak yang tidak diinginkan.

Keamanan informasi terdiri dari perlindungan terhadap aspek-aspek berikut (Syafrizal, 2007) :

1. *Confidentiality* (kerahasian) : aspek yang menjamin kerahasian data atau informasi, memastikan bahwa informasi hanya dapat diakses oleh orang yang berwenang dan menjamin kerahasian data yang dikirim, diterima dan disimpan.
2. *Integrity* (integritas) : aspek yang menjamin bahwa data tidak dirubah tanpa ada ijin pihak yang berwenang, menjaga keakuratan dan keutuhan informasi serta metode prosesnya untuk menjamin aspek integritas.
3. *Availability* (ketersedian) : aspek yang menjamin bahwa data akan tersedia saat dibutuhkan dan memastikan user yang berhak dapat menggunakan informasi dan perangkat terkait.

2.3 Security Standard Mapping

Berikut adalah *roadmap* standar dalam keamanan *cloud computing* (Sakurai, 2011) yaitu :

1. *Authentication* dan *Authorization*

Authentication adalah proses untuk memastikan bahwa pelau adalah benar-benar pelaku. Proses ini memastikan untuk jika terdapat orang lain yang mengakses maka akan terdeteksi sebagai orang lain bukan pelaku. Demikian juga apabila pelaku yang mengakses adalah benar yang bersangkutan maka proses juga dapat memastikan bahwa yang mengaku sebagai pelaku benar-benar sebagai pelaku. Seteleah proses *authentication* selesai atau berhasil maka proses selanjutnya adalah *Authorization.* Dalam proses *authorization* ini akan ditentukan menu apa saja yang bisa dijalankan oleh *user* tersebut. Biasanya setiap *user* sudah diberikan *rules* tertentu dalam menjalankan aplikasi yang telah dibangun.

Proses *authentication* dan *authirization* merupakan proses yang sangat penting dalam pengendalian aplikasi maupun proses bisnis dalam suatu perusahaan atau instansi. Berikut adalah tabel pemetaan standar keamanan yang terdapat didalam kategori *authentification* dan *authorization.*

Tabel 2.1 Standar Keamanan pada *Authentification* dan *Authorization*

Categorization	***Available Standards***	***SDO***	***Status***
Authentication and Authorization	*RFC 5246 Secure Socket Layer (SSL)/ Transport Layer Security (TSL)*	*IETF*	*Approved Standard Market Acceptance*
	RFC 3820: X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile	*IETF*	*Approved Standard Market Acceptance*
	RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Cerficate Revocation List (CRL) Profile	*IETF*	*Approved Standard Market Acceptance*
	RFC 5849 Oauth (Open Authorization Protocol)	*IETF*	*Approved Standard Market Acceptance*
	ISO/IEC 9594-8:2008 | X.509 Information technology – Open Systems Interconnection – The	*ISO/IEC & ITU-T*	*Approved Standard Market Acceptance*

Categorization	*Available Standards*	*SDO*	*Status*
	Directory: Public key and attribute certificate frameworks		
	FIPS 181 Automated Password Generator	*NIST*	*Approved Standard Market Acceptance*
	FIPS 190 Guideline for the Use of Advanced Authentication Technology Alternative	*NIST*	*Approved Standard Market Acceptance*
	FIPS 196 Entity Authentication Using Public Key Cryptography	*NIST*	*Approved Standard Market Acceptance*
	OpenID Authentication	*OpenID*	*Approved Standard Market Acceptance*
	eXtensible Access Control Markup Language (XACML)	*OASIS*	*Approved Standard Market Acceptance*
	Security Assertion Markup Languange (SAML)	*OASIS*	*Approved Standard Market Acceptance*

2. *Confidentiality*

Aspek yang menjamin kerahasian data atau informasi, memastikan bahwa informasi hanya dapat diakses oleh orang yang berwenang dan menjamin kerahasian data yang dikirim, diterima dan disimpan. Bocornya informasi dapat berakibat batal suatu proses bisnis tertentu. Akses terhadap informasi juga harus dilakukan dengan melalui mekanisme *authorization* yang ketat. Tingkat keamanan dari mekanisme *authorization* bergantung pada tingkat kerahasian data yang diinginkan.

3. *Integrity*

Integrity merupakan aspek yang menjamin bahwa data tidak boleh berubah tanpa izin dari pihak yang bersangkutan (*authorized*).

4. *Indentity Management*

Indentity management memberikan solusi untuk pengaturan identias didalam sebuah organisasi yang mencakup *lifecycle* dari sebuah identitas didalam sebuah perusahaan yang mencakup *authentificationi, authorization, audit dan user administration.*

5. *Security Monitoring and Incident Response*

Security Monitoring and incident response merupakan peraturan yang membahas tentang keamanan dalam jaringan komputer dan pengamanan dalam keamanan jaringan komputer apabila terjadi serangan-serangan didalam jaringan.

6. *Security Policy Management*

Merupakan sebuah infrastruktur keamanan yang harus dimiliki oleh sebuah organisasi atau perusahaan maupun instansi yang ingin melindungi aset informasi terpentingnya. Dengan adanya kebijakan, selain akan membantu dalam mengamankan aset penting tersebut juga menghindari adanya insiden atau tuntutan hukum akibat organisasi, perusahaan ataupun instansi lalai dalam melakukan pengelolaan aset informasi atau hal-hal yang terkait dengan tata kelola informasi yang berada dalam lingkungannya.

7. *Availability*

Aspek yang menjamin bahwa data tersedia ketika dibutuhkan. dan memastikan user yang berhak dapat menggunakan informasi dan perangkat terkait.

2.4 *National Institute of Standard and Technology* (NIST)

NIST (National Institute Of Standard and Technology atau Badan Nasional Standar dan Teknologi Amerika Serikat) yang dulunya dikenal sebagai The National Bureau of Standards - NBS (Biro Standar Nasional) adalah sebuah badan non-regulator dari bagian Administrasi Teknologi dari Departemen Perdagangan Amerika Serikat. Satu dari fungsi utama NIST adalah untuk mengembangkan, memelihara, dan mempertahankan keaslian dari standar nasional pengukuran, dan menyediakan sarana dan metode standar yang digunakan untuk membandingkan ilmu pengetahuan, rekayasa, manufaktur, perdagangan, industri, dan pendidikan dengan standar yang ditetapkan atau diakui oleh Pemerintah Federal. Sebagai sebuah agen dari pemerintahan Amerika, Departemen Perdagangan Teknologi

Informasi, NIST melibatkan ilmuwan-ilmuwan dan insinyur-insinyur secara langsung dalam mengembangkan standar yang diperlukan untuk teknologi yang maju saat ini. Fasilitas penelitian NIST berlokasi Gaithersburg, MD 20899, dan di Boulder, CO 80303 (Guttman & Roback, 1995).

2.5 *Automated Password Generator*

Automated Password Generator (APG) merupakan program perangkat keras atau perangkat lunak yang mengambil masukan dari bilang acak secara otomatis menghasilkan kata sandi. Kata sandi acak dapat dihasilkan secara manual dengan menggunakan sumber acak sederhana seperti dadu atau koin, atau bisa dihasilkan menggunakan komputer.

APG terdiri dari prosedur utama yang memiliki tiga komponen utama yaitu tabel unit, tabel diagram, dan prosedur angka acak. Random password generator bekerja untuk membentuk suku kata ejaan dan menggabungkannya sehingga membentuk sebuah kata. Aturan-aturan pengejaan disimpan dalam sebuah tabel pada setiap unit dan setiap pasangan unit-unit (diagram). Aturan tersebut digunakan untuk menentukan apakah unit yang diberikan legal atau ilegal yang didasarkan pada posisi unit terhadap suku kata dan unit yang berdekatan. Kebanyakan aturan tersebut dan pemeriksaanya berorientasi ejaan dan tidak bergantung pada ejaan selainnya.

Kode implementasi APG generator terdiri dari prosedur angka acak DES, prosedur aktual DES, dan kode untuk menghasilkan kunci pseudorandom. Dalam implementasi *password generator* dapat dibaca didalam kode. Sebenarnya implementasi nilai kunci dan masukannya harus rahasia yang secara acak dihasilkan oleh aplikasi.

2.5.1 Tabel Unit

Tabel unit menentukan unit (karakteristik alfabtik) dan menjelaskan aturan penyinggungan pada individu unit yang digunakan dalam menghasilkan kata secara acak. Contohnya lokasi huruf mati (vowel) pada kata yang dihasilkan ditentukan oleh aturan ini.

```
static struct unit  rules[] =
{{"a", VOWEL},
 {"b", NO_SPECIAL_RULE},
 {"c", NO_SPECIAL_RULE},
 {"d", NO_SPECIAL_RULE},
 {"e", NO_FINAL_SPLIT | VOWEL},
 {"f", NO_SPECIAL_RULE},
 {"g", NO_SPECIAL_RULE},
 {"h", NO_SPECIAL_RULE},
 {"i", VOWEL},
 {"j", NO_SPECIAL_RULE},
 {"k", NO_SPECIAL_RULE},
 {"l", NO_SPECIAL_RULE},
 {"m", NO_SPECIAL_RULE},
 {"n", NO_SPECIAL_RULE},
 {"o", VOWEL},
 {"p", NO_SPECIAL_RULE},
 {"r", NO_SPECIAL_RULE},
 {"s", NO_SPECIAL_RULE},
 {"t", NO_SPECIAL_RULE},
 {"u", VOWEL},
 {"v", NO_SPECIAL_RULE},
 {"w", NO_SPECIAL_RULE},
 {"x", NOT_BEGIN_SYLLABLE},
 {"y", ALTERNATE_VOWEL | VOWEL},
 {"z", NO_SPECIAL_RULE},
 {"ch", NO_SPECIAL_RULE},
 {"gh", NO_SPECIAL_RULE},
 {"ph", NO_SPECIAL_RULE},
 {"rh", NO_SPECIAL_RULE},
  {"sh", NO_SPECIAL_RULE},
  {"th", NO_SPECIAL_RULE},
  {"wh", NO_SPECIAL_RULE},
  {"qu", NO_SPECIAL_RULE},
  {"ck", NOT_BEGIN_SYLLABLE}
};
```

Gambar 2.2 Tabel Unit

2.5.2 Tabel Diagram

Tabel diagram menentukan aturan-aturan mengenai semua kemungkinan pasangan unit dan pendekatan unit yang digunakan. Tabel diagram berisi satu input untuk setiap pasangan unit (diagram) dan diuji, apakah pasangan tersebut diperkenankan atau tidak.

Random word generator memastikan aturan-aturan yang ditetapkan dalam tabel diagram dipenuhi untuk setiap dua unit berurutan dalam pembentukan kata. Berikut adalah sebagian kode tabel diagram :

```
static int  digram[][RULE_SIZE]
=
{
  {/* aa */ ILLEGAL_PAIR,
   /* ab */ ANY_COMBINATION,
   /* ac */ ANY_COMBINATION,
   /* ad */ ANY_COMBINATION,
   /* ae */ ILLEGAL_PAIR,
   /* af */ ANY_COMBINATION,
   /* ag */ ANY_COMBINATION,
   /* ah */ NOT_BEGIN | BREAK |
NOT_END,
   /* ai */ ANY_COMBINATION,
   /* aj */ ANY_COMBINATION,
   /* ak */ ANY_COMBINATION,
   /* al */ ANY_COMBINATION,
   /* am */ ANY_COMBINATION,
   /* an */ ANY_COMBINATION,
   /* ao */ ILLEGAL_PAIR,
   /* ap */ ANY_COMBINATION,
   /* ar */ ANY_COMBINATION,
   /* as */ ANY_COMBINATION,
   /* at */ ANY_COMBINATION,
   /* au */ ANY_COMBINATION,
   /* av */ ANY_COMBINATION,
   /* aw */ ANY_COMBINATION,
   /* ax */ ANY_COMBINATION,
   /* ay */ ANY_COMBINATION,
   /* az */ ANY_COMBINATION,
   /* ach */ ANY_COMBINATION,
   /* agh */ ILLEGAL_PAIR,
   /* aph */ ANY_COMBINATION,
   /* arh */ ILLEGAL_PAIR,
   /* ash */ ANY_COMBINATION,
   /* ath */ ANY_COMBINATION,
   /* awh */ ILLEGAL_PAIR,
   /* aqu */ BREAK | NOT_END,
   /* ack */ ANY_COMBINATION},
  {/* ba */ ANY_COMBINATION,
   /* bb */ NOT_BEGIN | BREAK |
NOT_END,
   /* bc */ NOT_BEGIN | BREAK |
NOT_END,
   /* bd */ NOT_BEGIN | BREAK |
NOT_END,
   /* be */ ANY_COMBINATION,
```

Gambar 2.3 Tabel Diagram

2.5.3 Prosedur Algoritma

Berikut adalah beberapa prosedur dari algoritma dari *auto password generator* :

1. Periksa apakah variabel minlen > maxlen. Jika iya, maka terjadi kesalahan.
2. Periksa kata yang memiliki panjang nol. Secara teknik ini bukan kesalahan sehingga hanya mengembalikan kata null dengan panjang nol.
3. Temukan passoword :
   a. Insialisasi array yang menyimpan unit kata. Ketika diketahui panjang kata, hanya dibutuhkan salah satu panjang. Metode ini lebih baik digunakan pada array yang statik, karena memperkenankan fleksibilitas pemilihan panjang kata yang berubah-ubah. Karena sebuah kata dapat berisi satu ejaan, harus dibuat unit ejaan, array tersebut memiliki unit analog untuk tiap ejaan dengan panjang sama. Tidak ada aturan eksplisit yang membatasi panjang ejaan, tetapi aturan diagram dan heurististik melakukannya secara tidak langsung.
   b. Temukan ejaan sampai kata masukan terbangun.
      i. Dapatkan ejaan dan tentukan panjangnya.
      ii. Gabungkan unit ejaan dengan unit kata
      iii. Jika kata tersebut tidak tepat dibentuk, keluarkan ejaan tersebut. Pemeriksaan yang dilakukan disini adalah kata yang harus dibentuk dalam basis kata. Pengujian lainnya dilakukan sepenuhnya didalam ejaan tersebut. Sebaliknya, gabungkan ejaan tersebut pada kata dan gabungkan ejaan tersebut pada hasil kata.
      iv. Periksa kata tersebut tidak berisi kombinasi ilegal yang dapat menjangkau ejaan. Khususnya yaitu :
         1. Sebuah pasangan ilegal dari unit-unit antara ejaan.
         2. Tiga unit vowel berurutan.
         3. Tiga unit konsonan berurutan.
      v. Modifikasi silabel untuk angkat atau simbol kapital yang dibutuhkan. Dilakukan setelah kualitas kata diperiksa.
      vi. Tetap menjaga tracking waktu yang telah dicoba untuk mendapatkan ejaan. Jika melebihi threshold, maka insialisasi kembali variabel pwlen dan word_size, bersihkan array kata dan mulai dari awal.

      Prosedur utamanya sebagai berikut :

```
int gen_pron_pass (char *word, char
*hyphenated
_word, USHORT minlen,
USHORT maxlen, unsigned int pass_mode)
{
    int     pwlen;
    if (minlen > maxlen || minlen >
 APG_MAX_PASSWORD_LENGTH ||
        maxlen > APG_MAX_PASSWORD_LENGTH)
       return (-1);

    if (maxlen == 0)
    {
     word[0] = '\0';
     hyphenated_word[0] = '\0';
     return (0);
    }

    pwlen = gen_word (word,
 hyphenated_word, get_random
 (minlen, maxlen), pass_mode);
    return (pwlen);
}
```

Gambar 2.4 Prosedur Algoritma

2.5.4 *Random Number Generator*

Random number generator menggunakan prosedur DES untuk menghasilkan nilai presisi antara 0 dan 1. Angka ini digabungkan oleh variabel sebuah program n yang bertipe integer. Operasi ini menghasilkan integer acak antara 0 dan (n-1). Angka acak yang dibuat oleh rutin DES menghasilkan output yang nantinya digunakan sebagai input bagi random word generator. Prosedur ini menghasilkan sejumlah angka yang akan dipanggil oleh word generator setiap waktu sebuah karakter (unit) dibutuhkan. Tidak semua karakter yang dihasilkan akan diterima oleh word generator pada setiap posisi dalam kata. Setiap karakter diperiksa untuk ketepatan dan kesesuaiannya dengan aturan yang didefinisikan oleh tabel unit dan diagram. Oleh karenanya prosedur random number generator akan dipanggil berulang sampai menghasilkan karakter yang dapat diterima. Pemanggilan dibatasi sampai 100 kali. Jika telah mencapai nilai 100 maka kata tersebut ditolak dan program memulai kembali dari awal.

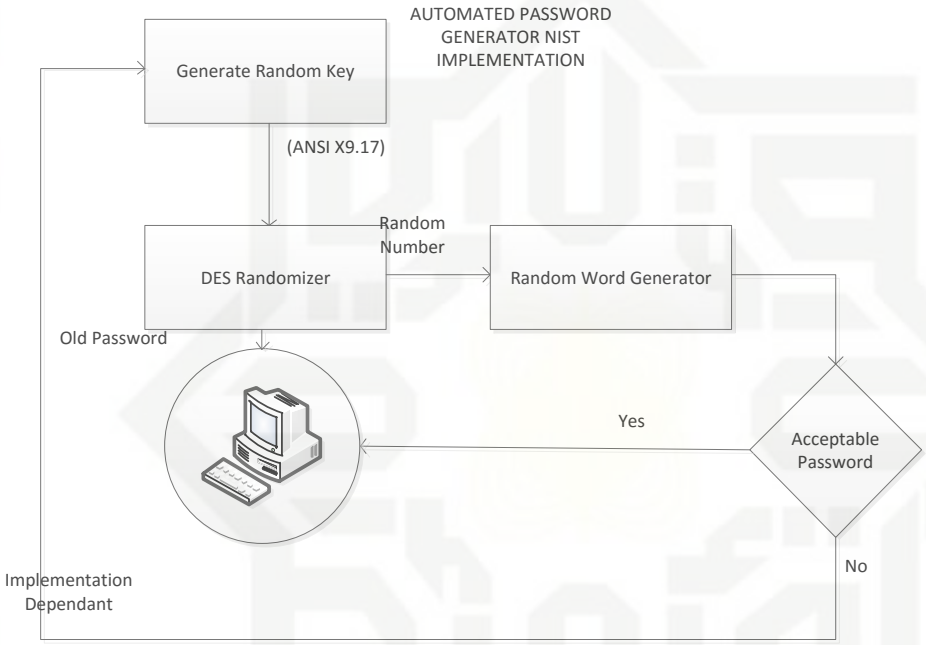
Distiribusi unit yang legal berbeda untuk setiap posisi bagian kata, untuk setiap unit, tergantung pada unit-unit sebelumnya menurut tabel unit dan diagram. Prosedur random number sendiri tidak melakukan pengujian unit yang telah legal. DES menerima input dua bits blok data. Salah satunya terdiri dari *password* lama atau string data; lainnya merupakan 64 bit (56 bits + 8 parity bits) kunci pseudorandom yang diturunkan dengan menggunakan prosedur standar. *Password* lama dimasukan secara manual dari *keyboard*. *Input* array dibuat dari bytes pertama dari delapan karakter *password* atau *input string*. Program akan menerima *string null* (*carriage return*). Karakter kedelapan diabaikan. Jika *input* blok kurang dari delapan karakter maka akan ditambahkan elemen ekstra. DES mode *electronic code book* (ECB) digunakan untuk mengenkripsi *input* data. *Output*nya adalah angka acak 64 bit yang dienkripsi bentuk *input*nya.

Fungsi pertama dalam struktur DES adalah setkey(), yang mengkonversi kunci pseudorandom menjadi sebuah format yang digunakan. DES dalam melakukan enkripsi. *Input* bagi fungsi setkey pada *command line* adalah (0,0,*key*). Nol yang pertama diset sehingga setkey() tidak menghasilkan pariti; nol yang kedua memberitahu setkey() bahwa enkripsi (bukan deskripsi) dibutuhkan. Hasil keluaran array, dikirm ke sebuah fungsi, answer(), yang mengembalikan angka final yang dibutuhkan. Fungsi answer() berada pada alamat dari array *output* sebagai *pointer* char *unsignet* dan integer n yang memiliki nilai 0 sampai (n-1) yang dibutuhkan oleh program kata acak. Fungsi ini membuat variabel sum, didefinisikan sebagian integer asli. Untuk mendapatkan nilai angka dari keluaran array karakter, tambahkan nilai ASCII pada tiga elemen pertama dalam keluaran array dan menyimpan penjumlahannya pada variable sum, bertipe integer. Untuk mendapatkan sebuah angka dengan rentang dari 0 sampai n-1 dari penjumlahan, fungsi tersebut mencari modulus sum dan n. Nilai ini kemudian dikembalikan pada fungsi pemanggil dalam program random word.

2.5.5 *Random Word Generator*

Algortima kata acak digunakan untutk menghasilkan kata acak sudah tetap dan tidak dapat dimodifikasi tanpa perubahan logik dari program. Fungsi dari

algoritma tersebut digunakan untuk menentukan apakah unit acak, dapat digabungkan pada akhir kata yang dibentuk sejauh ini. Aturan pengejaan disimpan dalam tabel unit dan diagram dijelaskan sebelumnya. Aturan tersebut digunakan untuk memeriksa jika unit yang diberikan legal atau ilegal. Jika illegal, unit ditolak dan prosedur unit acak dipanggil kembali. Ketika unit diterima, berbagai macam status variabel di *update* dan sebuah unit untuk posisi berikutnya dalam kata dicoba.



Gambar 2.5 Automated Password Generator NIST *Implementation* (NIST, 1995)

Gambar 2.5 adalah diagram blok implementasi NIST dari algoritma *automated password generator*. *Personal Computer* (PC) yang digunakan oleh NIST untuk menunjukkan standard. NIST mengganti nomor Unix secara acak dengan fungsi "DES Randomizer" dan "Generate Random Key". DES menerima kata sandi lama dan kunci pseudorandom menghasilkan sebuah angka acak. Angka acak tersebut digunakan oleh Random Word Generator untuk membentuk *password*. Sebagai *password* yang dihasilkan, setiap kelompok huruf akan di cek terlebih dahulu tata bahasa dan semantik untuk menentukan apakah sbeuah kata yang telah dibuat dapat diterima. Jika sudah dicek, maka kata sandi baru akan ditampilakn ke PC. DES (*Data Encryption Standar*) memiliki *input* yaitu *password* lama atau karakter string dari *user* dan sebuah kunci pseudorandom. Perubahan

sekecil apapun terhadap kunci atau input data string akan menyebabkan DES menghasilkan angka acak yang berbeda. Setiap kali perubahan ini terjadi, password generator menghasilkan password acak yang baru.

Password yang dibuat oleh APG ini terdiri dari 26 karakter alfabet inggris. Meskipun angka-angka dan karakter spesial tidak dimasukkan, ruang password yang merupakan sebuah fungsi jumlah karakter dalam *password* sangatlah besar. Diperkirakan 18 juta 6-karakter, 5.7 milyar 8- karakter, dan 1.6 triliun 10-karakter *password* dapat diubah oleh program tersebut. *User* harus memilih sebuah ruang *password* sepadan dengan *level* keamanan yang dibutuhkan bagi informasi agar terlindungi. Algoritma *password* tersebut tidak menghalangi menghasilkan kata-kata yang ditemukan dalam kamus kata bahasa inggris. Jika dibutuhkan, kamus kata terkomputerisasi harus digunakan untuk memeriksa kata bahasa inggris dan implementasinya harus memasukan pengujian *test* untuk mencegah dari tawaran kepada *user* sebagai *password* .

Gambar 2.5 merupakan diagram blok implementasi algoritma APG. Program ini menggantikan fungsi acak angka Unix dalam versi aslinya dengan “DES Randomizer” dan fungsi “Generate Random Key”. DES Randomizer menerima *password* lama dan kunci pseudorandom dan menghasilkan sebuah angka acak. Angka ini digunakan oleh generator huruf acak untuk membentuk sebuah *password.* Setelah *password* telah dihasilkan oleh setiap kelompok huruf kemudian diuji *grammer* dan semantiknya untuk menentukan jika kata yang diterima telah dibuat. Jika semua proses telah dilalui, maka *password* akan dikeluarkan ke PC. Dalam implementasinya, nilai *minlen* dan *maxlen* (yaitu panjang *password*), diset menjadi 5 dan 8. *User* membutuhkan *password* dengan panjang tetap harus menset variabel ini pada nilai yang spesifik.

2.6 Algoritma DES

Secara umum, algoritma DES terbagi menjadi 3 kelompok dimana kelompok yang satu dengan yang lain saling berinteraksi dan terkait antara satu dengan yang lain. Kelompok-kelompok tersebut adalah Pemrosesan kunci, enkripsi data 64 bit, dan deskripsi data 64 bit (Andri, 2009). Algoritma DES dirancang untuk

menulis dan membaca berita blok data yang terdiri dari 64 bit di bawah kontrol kunci 64 bit. Dalam pembacaan berita harus dikerjakan dengan menggunakan kunci yang sama dengan waktu menulis berita, dengan pejadwalan alamat kunci bit yang diubah sehingga proses membaca adalah kebalikan dari proses menulis.

Sebuah blok ditulis dan ditunjukan pada permutasi dengan insial IP, kemudian melewati perhitungan dan perhitungan tersebut sangat tergantung pada kunci kompleks dan pada akhirnya melewati permutasi yang invers dari permutasi dengan insial IP^{-1}. 64 bit dari blok input yang dienkripsi adalah subjek pertama dari permutasi yang disebut permutasi dengan inisial IP. Perhatikan Tabel 2.2 insial IP dibawah.

Tabel 2.2 Initial Permutation

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Cara membaca tabel atau matriks dua *entry* ujung kiri atas (58 dan 50) berarti “pindahkan bit ke-58 ke posisi 1”, “pindahkan bit ke-50 ke posisi bit 2”, dst. *Output* dari perhitungan ini, disebut *preoutput* dan *output* ini akan diteruskan pada permutasi berikutnya yang merupakan kebalikan dari permutasi inisial. Perhatikan tabel 2.3 kebalikan dari permutasi inisial IP yaitu IP^{-1}

Tabel 2.3 IP^{-1}

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29

36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Kunci *eksternal* 64 bit, dikompresi terlebih dahulu menjadi 56 bit menggunakan matriks permutasi kompresi PC-1. Dalam permutasi tiap bit kedelapan dari delapan *byte* kunci akan diabaikan. Sehingga akan ada pengurangan delapan bit dari 64 bit awal kunci *eksternal*. Matriks permutasi kompresi PC-1 :

Tabel 2.4 PC-1

PC-1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	45	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Proses selanjutnya adalah kedua bagian digeser ke kiri sepanjang satu atau dua bit tergantung pada tiap putaran. Putaran ini bersifat *wrapping* atau *round-shift*. Berikut adalah tabel 2.5 jumlah pergeseran bit pada tiap putaran.

Tabel 2.5 Jumlah Pergeseran Bit

Putaran ke-i	Jumlah Pergeseran
1	1
2	1
3	2
4	2
5	2
6	2
7	2

Putaran ke-i	Jumlah Pergeseran
8	1
9	2
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Untuk setiap hasil C($_i$) dan D ($_i$), kunci pada iterasi ke-*i* didapatkan dengan cara melakukan permutasi kembali pada C($_i$) dan D($_i$). Permutasi itu dikenal dengan nama Permuted Choice (PC-2). Berikut hasilnya :

Tabel 2.6 PC-2

PC-2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

a. Enciphering

Proses enciphering terhadap blok plainteks dilakukan setelah permutasi awal. Setiap blok plainteks mengalami 16 kali putaran enciphering. Setiap putaran enciphering merupakan jarinagan Feistel yang secara matematis dinyatakan sebagai:

$$L_i = R_{i-1} \qquad (2.1)$$

$$R_i = L_{i-1} \oplus f(R_{1-1}, Ki) \quad (2.2)$$

E adalah fungsi ekspansi yang memperluas blok R_{i-1} yang panjangnya 32-bit menjadi blok 48 bit. Fungsi ekspansi direalisasikan dengan matriks ekpansi sebagai berikut :

Tabel 2.7 Tabel Ekpansi Permutasi

Tabel Ekspansi Permutasi					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	33

Selanjutnya, hasil ekspansi, yaitu $E(R_{i-1})$, yang panjangnya 48 bit di XOR-kan dengan K_i yang panjangnya 48 bit menghasilkan vektor *A* yang panjangnya 48-bit :

$$E(Ri_{-1}) \oplus K_i = A \quad (2.3)$$

Tabel 2.8 P-box

P-box							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Vektor *A* dikelompokkan menjadi 8 kelompok, masing-masing 6 bit, dan menjadi masukan bagi proses subsitusi. Proses subsitusi dilakukan dengan menggunakan delapan buah kotak-S (*S-box*), S_1 sampai S_8. Setiap kotak-*S* menerima ukuran 6 bit dan menghasilkan keluaran 4 bit. Kelompok 6-bit pertama menggunakan S_1, kelompok 6-bit kedua menggunakan S_2, dan seterusnya. Berikut adalah Tabel 2.9 kotak-*S* dari $S_1 - S_8$.

Tabel 2.9 kotak-*S* $S_1 - S_8$

Substitution Box 1 (*S1*)																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
Substitution Box 2 (S_2)																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
Substitution Box 3 (S_3)																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	2	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
Substitution Box 4 (S_4)																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	9	12	7	2	14
Substitution Box 5 (S_5)																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	15	10	11	14	1	7	6	0	8	13	12
Substitution Box 6 (S_6)																

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	9	7	5	10	6	1
1	10	15	4	2	7	12	9	5	6	1	5	12	2	15	8	6
2	9	14	15	5	2	8	12	3	7	0	6	8	0	5	9	2
3	4	3	2	12	9	15	10	11	14	1	0	15	14	2	3	12

Substitution Box 7 (S_7)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

Substitution Box 8 (S_8)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

2.7 Pengujian Sistem

Didalam pengujian sistem, dilakukan pengujian dengan menggunakan *black box.* Pengujian *Black Box* atau dikenal dengan *Black Box Testing* merupakan metode pengujian perangkat lunak yang akan melakukan pengujian secara fungsionalitas dari aplikasi atau sistem yang dibangun apakah bertentangan dengan proses kerja atau tidak. *Black box* dapat menemukan kesalahan dalam kategori berikut (Rouf, n.d) :

1. Fungsi-fungsi yang tidak benar
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses basis data *eksternal*
4. Inisialisasi dan kesalahan terminasi
5. *Validitas* fungsional
6. Kesensitifan sistem terhadap nilai input tertentu
7. Batasan dari suatu data.

2.8 Penelitian Terkait

Penilitian-penilitian sebelumnya berguna sebagai referensi dalam pengerjaan penelitian ini. Penilitian terkait tentang *cloud computing* dapat dilihat pada tabel 2.10 di bawah ini sebagai berikut :

Tabel 2.10 Penelitian Terkait

No	Judul	Penulis	Tahun	Metode	Kesimpulan
1	Pengaruh Dimensi *Trust*, Keamanan dan Privasi Terhadap Kepercayaan Pengguna Untuk Layanan *Cloud Computing* Berbasis *Software as a Service*	Prita Haryani, Eko Nugroho, Dani Adhipta	(2015)	*Partial Least Square*	Adanya jaminan keamanan dan privasi data pengguna, maka akan menumbuhkan kepercayaan (*trust)* pengguna terhadap penyedia layanan. Hasil dari penelitian ini menemukan bahwa faktor yang secara signifikan mempengaruhi kepercayaan pengguna adalah faktor *ability*, *security* dan *privacy* dari penyedia layanan.
2	Model Implementasi *Centralized Authentication Service* pada	Muhammad Arfan	(2014)	*Single Sign On*	Sistem otentikasi *single sign-on* ini dapat digunakan pada layanan *Cloud Software as a Service* sehingga

	sistem *Software as a Service*				keamanan dan kemudahan dalam proses otentikasi dapat diperoleh pengguna dan pengelola layanan. Disisi pengguna kemudahan dalan memperoleh akses aplikasi dan pengelola mampu mengidentifikasi kesalahan dan serangan pada aplikasi yang masuk kedalam *cloud.* Proses pengamanan satu pintu juga menjadi ancaman jika pengelola tidak menyediakan sistem *backup* dan mitigasi yang baik
3	Penerapan AES Untuk Otentikasi Akses Cloud Computing	Imanah, Arif Djunaidy, Muhammad Husni	(2010)	*AES (Advance Encryption Standard)*	Hasil penelitian menunjukkan bahwa sistem otentikasi yang dikembangkan dalam penelitian ini layak untuk

					diaplikasikan dalam lingkungan cloud computing
4	Analisis Teknik-Teknik Keamanan Pada *Cloud Computing dan* NEBULA (*Future Cloud*)	Beny Nugraha	(2016)	*Attack Centric, Proof of Consent (*PoC*), Proof of Path* (PoP), dan teknik kriptografi ICING.	*Cloud computing* saat ini tidak fleksibel, Teknik keamanan yang ada pada *cloud computing* saat ini belum cukup untuk mengatasi seluruh serangan keamanan yang diteliti, Serangan *traffic analysis* dapat diatasi dengan pemakaian *onion routing* yang dapat diimplementasikan dengan cepat pada nebula
5	Data Security in Cloud Computing using RSA Algorithm	Parsi Kalpana, Sudha Singaraju	(2012)	*RSA Algotithm*	Apabila terdapat seseorang yang bukan berhak mengakses data tersebut maka data yang diambil atau dicuri akan otomatis terenkripsi sehingga tidak dapat mendeskripsikan dan mendapatkan

					keaslian data yang dicuri tersebut.
6	Manajemen Resiko Pada Implementasi SaaS (*Software as a Service*)	Toni Kusnandar	(2012)	*Analytical Network Process and Analytical Hierarchy Process, COTS Software Evaluation Methods, Procurement Oriented Requirements Engineering*	Analisis risiko membantu mengurangi ketidakpastian dalam pengambilan keputusan rasional dan mengumpulkan informasi yang relevan untuk mengurangi ketidakpastian
7	Enkripsi Data Menggunakan Steganografi Untuk Keamanan Data Pada Cloud	Imamah	(2015)	Algoritma *Embedding*	Metode steganografi tidak merubah bentuk cover image. Hal ini berarti penyisipan file yang dilakukan tidak merubah data yang disembunyikan tidak terlihat dan sangat tepat diterapkan pada cloud computing yang memiliki keterbatasan sumber daya

8	Security Issue For Cloud Computing	Kevin Hamlen, Murat Kantarcioglu, Latifur Khan, Bhavani Thuraisingham	(2010)	FIPS PUB 140	Karena kompleksitas *cloud*, Akan sulit untuk mencapai keamanan end-to-end. Namun, tantangan yang dihadapi adalah memastikan operasi yang lebih aman bahkan jika beberapa bagian dari *cloud* gagal untuk banyak aplikasi, tidak hanya perlu jaminan informasi saja.