BAB II

LANDASAN TEORI

2.1 Algoritma

Dalam bahasa Indonesia, kata *algorithm* diserap menjadi **algoritma**. Mulanya kata *algorism* berasal dari nama penulis buku arab yang terkenal, yaitu **Abu Ja'far Muhammad ibnu Musa Al-Khuwarizmi**. *Al-Khuwarizmi* oleh orang barat dibaca menjadi al-gorism. Al-Khuwarizmi adalah penulis buku yang berjudul **Kitab Al-Jabar Wal-Muqabala** (*The Book of Restoration and Reduction*). Pada akhirnya algorism berubah menjadi algoritm, sehingga akhiran -*sm* berubah menjadi –*thm*.

Defenisi Algoritma, *"Algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis"*. Sedangkan menurut Kamus Besar Bahasa Indonesia, terbitan Balai Pustaka, 1988. "*Algoritma adalah urutan logis pengambilan putusan untuk pemecahan masalah"*.

Kata **logis** merupakan kata kunci. Langkah-langkah yang disusun tersebut harus logis, artinya nilai kebenarannya harus dapat ditentukan, benar atau salah. Berikut ini adalah contoh menuliskan algoritma TUKAR ISI BEJANA;

Algoritma **TUKAR_ISI_BEJANA**

Diberikan dua buah bejana, A dan B; bejana A berisi larutan berwarna merah, bejana B berisi larutan berwarna biru. Pertukarkan isi kedua bejana itu

sedemikian sehingga bejana A berisi larutan berwarna biru dan bejana B berisi larutan merah.

Deskripsi :

1. Tuangkan larutan dari bejana A ke dalam bejana C
2. Tuangkan larutan dari bejana B ke dalam bejana A
3. Tuangkan larutan dari bejana C ke dalam bejana B

2.1.1 Algoritma Kriptografi

Kriptografi adalah suatu ilmu yang mempelajari bagaimana cara menjaga agar data atau pesan tetap aman saat dikirimkan, dari pengirim ke penerima tanpa mengalami gangguan dari pihak ketiga. Menurut Bruce Scheiner dalam bukunya "Applied Cryptography", Kriptografi adalah ilmu pengetahuan dan seni menjaga message-message agar tetap aman (secure).

Konsep kriptografi sendiri telah lama digunakan oleh manusia misalnya pada peradaban Mesir dan Romawi walau masih sangat sederhana. Prinsip-prinsip yang mendasari kriptografi yakni:

- *Confidelity* (kerahasiaan) yaitu layanan agar isi pesan yang dikirimkan tetap rahasia dan tidak diketahui oleh pihak lain (kecuali pihak pengirim, pihak penerima / pihak-pihak memiliki ijin). Umumnya hal ini dilakukan dengan cara membuat suatu algoritma matematis yang mampu mengubah data hingga menjadi sulit untuk dibaca dan dipahami.

- *Data integrity* (keutuhan data) yaitu layanan yang mampu mengenali/mendeteksi adanya manipulasi (penghapusan, pengubahan atau penambahan) data yang tidak sah (oleh pihak lain).
- *Authentication* (keotentikan) yaitu layanan yang berhubungan dengan identifikasi. Baik otentikasi pihak-pihak yang terlibat dalam pengiriman data maupun otentikasi keaslian data/informasi.
- *Non-repudiation* (anti-penyangkalan) yaitu layanan yang dapat mencegah suatu pihak untuk menyangkal aksi yang dilakukan sebelumnya (menyangkal bahwa pesan tersebut berasal darinya).

Berbeda dengan kriptografi klasik yang menitikberatkan kekuatan pada kerahasiaan algoritma yang digunakan (yang artinya apabila algoritma yang digunakan telah diketahui maka pesan sudah jelas "bocor" dan dapat diketahui isinya oleh siapa saja yang mengetahui algoritma tersebut), kriptografi modern lebih menitikberatkan pada kerahasiaan kunci yang digunakan pada algoritma tersebut (oleh pemakainya) sehingga algoritma tersebut dapat saja disebarkan ke kalangan masyarakat tanpa takut kehilangan kerahasiaan bagi para pemakainya.

Berikut adalah istilah-istilah yang digunakan dalam bidang kriptografi :

- **Plaintext** adalah pesan yang hendak dikirimkan (berisi data asli).
- **Ciphertext** adalah pesan ter-enkrip (tersandi) yang merupakan hasil enkripsi.
- **Enkripsi** adalah proses pengubahan plaintext menjadi ciphertext.

- **Dekripsi** adalah kebalikan dari enkripsi yakni mengubah ciphertext menjadi plaintext, sehingga berupa data awal/asli.
- **Kunci** adalah suatu bilangan yang dirahasiakan yang digunakan dalam proses enkripsi dan dekripsi.

Dalam kriptografi terdapat dua konsep utama yakni enkripsi dan dekripsi. Enkripsi adalah proses dimana informasi/data yang hendak dikirim diubah menjadi bentuk yang hampir tidak dikenali sebagai informasi awalnya dengan menggunakan algoritma tertentu. Dekripsi adalah kebalikan dari enkripsi yaitu mengubah kembali bentuk tersamar tersebut menjadi informasi awal.

Algoritma kriptografi berdasarkan jenis kunci yang digunakan dapat dibedakan menjadi dua jenis yaitu :



Gambar 2.1 Alur Algoritma *Simetris*

a. Algoritma *simetris*

Dimana kunci yang digunakan untuk proses enkripsi dan dekripsi adalah kunci yang sama(**K**).



Gambar 2.2 Alur Algoritma *Asimetris*

b. Algoritma *asimetris*

Dimana kunci yang digunakan untuk proses enkripsi (**K1**) atau sering disebut dengan *public key* dan dekripsi (**K2**) atau sering disebut dengan *private key* menggunakan kunci yang berbeda.

Sedangkan berdasarkan besar data yang diolah dalam satu kali proses, maka algoritma kriptografi dapat dibedakan menjadi dua jenis yaitu :

a. Algoritma *block cipher*

Informasi/data yang hendak dikirim dalam bentuk blok-blok besar (misal 64-bit) dimana blok-blok ini dioperasikan dengan fungsi enkripsi yang sama dan akan menghasilkan informasi rahasia dalam blok-blok yang berukuran sama.

b. Algoritma *stream cipher*

Informasi/data yang hendak dikirim dioperasikan dalam bentuk blok-blok yang lebih kecil (byte atau bit), biasanya satu karakter persatuan waktu proses, menggunakan tranformasi enkripsi yang berubah setiap waktu.

Penjelasan tersebut dimaksudkan sebagai perbandingan dengan algoritma yang lain, karena Algoritma RC4 termasuk ke dalam algoritma *simetris* dan algoritma *stream chipher*.

2.1.1.1 Mengenal Algoritma RC4

Dalam kriptografi RC4 juga dikenal sebagai ARC4 atau ARCFOUR adalah nol atau kode aliran (*stream*) perangkat lunak yang digunakan secara luas dan digunakan di dalam protokol populer seperti Secure Sockets Layer (SSL) untuk melindungi lalu lintas Internet dan WEP untuk mengamankan jaringan nirkabel. Sementara itu, RC4 luar biasa sederhana dan prosesnya cepat dalam perangkat lunak, RC4 mudah untuk diserang pada saat awal *output keystream* tidak dibuang, atau *keystream* tunggal yang digunakan dua kali. Beberapa cara untuk menggunakan RC4 bisa mendorong ke arah yang tidak aman dalam *cryptosystems* seperti WEP.

2.1.1.2 Sejarah Algoritma RC4

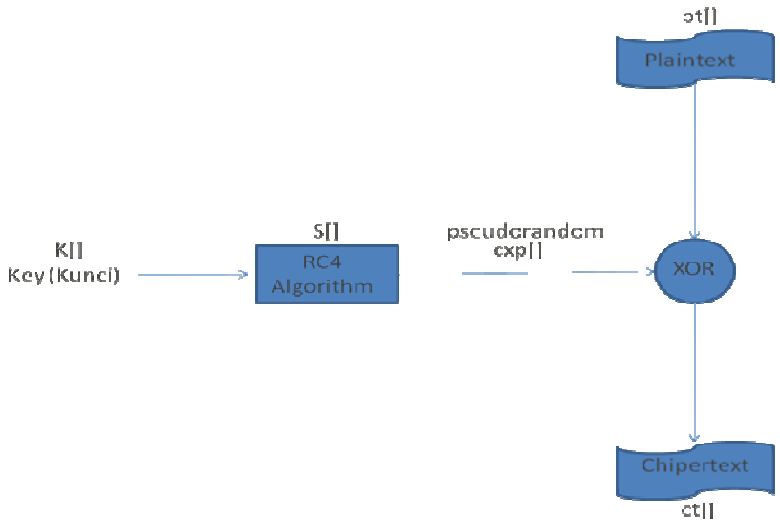
RC4 dirancang oleh Ron Rivest dari RSA Security pada tahun 1987. Sementara itu akhirnya telah diresmikan "Rivest Cipher 4", singkatan dari RC adalah alternatif untuk dipahami yang maksudnya “Ron Code”.

RC4 pada awalnya sebagai rahasia perniagaan, tetapi pada bulan September 1994 sebuah uraian kejanggalan diposkan pada Cypherpunks.

Kejanggalan tesebut segera dan telah poskan kepada kelompok berita yaitu sci.crypt, kemudian dari sana disebarluaskan ke berbagai tempat atau alamat di Internet. Kode telah dibocorkan lalu mencari dan menetapkan yang asli sebagai outputnya. Output tersebut ditemukan untuk dibandingkan dengan perangkat lunak menggunakan kepemilikan RC4 resmi. Karena algoritma telah diketahui, itu tidak lagi menjadi sebagai rahasia perniagaan. Nama "RC4" adalah cap/merek dagang, bagaimanapun. RC4 sering disebut sebagai "ARCFOUR" atau "ARC4" (yang maksudnya bahwa “ARCFOUR” atau “ARC4” itu diduga keras sebagai RC4, karena RSA tidak pernah secara resmi melepaskan algoritma RC4 tesebut), untuk menghindari permasalahan merek dagang. Itu telah menjadi bagian yang umum digunakan untuk protokol enkripsi dan standar, termasuk WEP dan WPA untuk *wireless card* dan TLS.

2.1.1.3 Uraian Algoritma RC4

RC4 membangitkan suatu aliran bit (*stream*) *pseudo-random* (*keystream*), untuk enkripsi, dan digabungkan dengan *plaintext* menggunakan XOR, kemudian didekripsi dengan melakukan cara yang sama.



Gambar 2.3 Deskripsi Algoritma RC4

(Hal ini serupa dengan *Vernam chiper* kecuali *pseudorandom bits,* dari pada bit acak, yang digunakan.) Untuk membangkitkan *keystream*, nol atau kode (*chiper*) membuat penggunaan dari kondisi internal yang bersifat rahasia yang terdiri dari dua bagian:

- Sebuah permutasi dari keseluruhan kemungkinan 256 byte (ditandai "S" di bawah).
- Dua 8-bit index-pointers (ditandai "i" dan "j").

Permutasi diinitialisasi dengan suatu kunci yang panjangnya bervariasi, secara tipikal antara 40 dan 256 bit, dengan menggunakan algoritma mengatur jadwal-kunci (*key-scheduling algorithm* KSA). Jika sudah terselesaikan, aliran (*stream*) dari bit ditetapkan dengan menggunakan algoritma generasi pura-pura-acak cxp[] (*pseudo-random generation algorithm* PRGA).

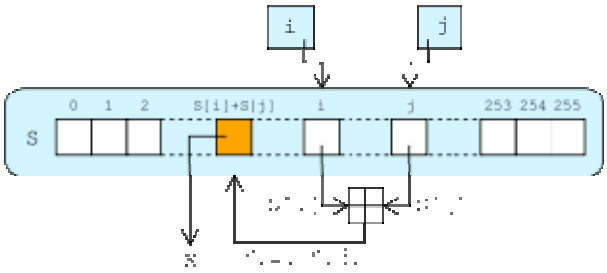
2.1.1.4 Algoritma mengatur jadwal-kunci (KSA)

Algoritma mengatur jadwal-kunci digunakan untuk menginitialisasi permutasi dalam larik (*array*) "S". "*keylength*" (K[])yang didefinisikan sebagai jumlah byte dalam kunci dan yang kemungkinannya terjadi dalam jangkauan antara 1 = *keylength* = 256. Larik (*array*) "S" adalah initial pada permutasi identitas. S kemudian memproses untuk 256 iterasi dengan cara yang sama pada algoritma PRGA, juga membuat campuran dalam byte dari kunci pada waktu yang bersamaan.

```
key_scheduling()
{
        /* initialisation */
        for i = 0 to 255
        s[i] = i;

        /* scrambling */
        j = 0;
        for i = 0 to 255
        {
                j = j + K[i % k_size] + S[i];
                swap S[i] and S[j];
        }
}
```

2.1.1.5 Algoritma generasi pura-pura-acak (PRGA)



Gambar 2.4 PRGA secara umum

Langkah- langkah RC4.

Output byte menyeleksi dengan cara memandang nilai dari S[i] dan S(j), kemudian S[i] dan S[j] ditambahkan secara bersamaan (modulo 256), dan untuk melihat penjumlahan dalam S, S(S[i] + S[j]) digunakan sebagai kunci aliran byte (*keystream*) yang disebut K.

Untuk banyaknya iterasi sangat dibutuhkan, PRGA memodifikasi kondisi dan keluaran sebuah *byte* dari keseluruhan kunci aliran byte (*keystream)*. Pada setiap iterasi, PRGA meningkatkan *i*, menambahkan nilai S yang ditunjukan pada i dan j, dan menukarkan nilai dari S[i] dan S[j], keluaran nilai S di lokasi S[i] + S[j] (modulo 256). Setiap nilai S ditukar sedikitnya sekali setiap 256 iterasi dilakukan. Contoh *script*:

```
PRGA()

{

    i = 0;

    j = 0;
```

```
        while not end of stream
        {
                i = (i + 1) % 256;
                j = (j + S[i]) % 256;
                swap S[i] and S[j];
                t = S[i] + S[j];
                ct[i] = pt[i] xor S[t];
        }
}
```

2.1.1.6 Implementasi

Banyak *chiper stream* berdasarkan pada umpan balik linier pergeseran register (*linear feedback shift registers* LFSRs), yang sementara efisien dalam perangkat keras sampai dengan perangkat lunak. Perancangan RC4 menghindari dari penggunaan LFSRs dan ideal untuk implementasi perangkat lunak, sebagaimana diperlukannya hanya memanipulasi *byte*. Manipulasi byte tersebut menggunakan 256 byte memori untuk kondisi array (larik), S[0] melalui S[255], *k* byte memori untuk *key*, *key*[0] melalui *key*[k-1], dan variabel bilangan bulat (*integer*), *i*, *j*, dan *k*. Di bawah ini implementasi sederhana dalam Python:

```
class WikipediaARC4:
    def __init__(self, key = None):
        self.state = range(256) # Initialize state array with values 0 .. 255
        self.x = self.y = 0 # Our indexes. x, y instead of i, j
        if key is not None:
            self.init(key)

    # KSA
    def init(self, key):
        for i in range(256):
            self.x = (ord(key[i % len(key)]) + self.state[i] + self.x) & 0xFF
            self.state[i], self.state[self.x] = self.state[self.x], self.state[i]
        self.x = 0

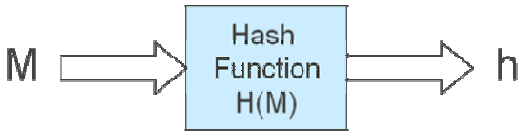
    # PRGA
    def crypt(self, input):
        output = [None]*len(input)
```

```
        for i in xrange(len(input)):
            self.x = (self.x + 1) & 0xFF
            self.y = (self.state[self.x] + self.y) & 0xFF
            self.state[self.x], self.state[self.y] = self.state[self.y], self.state[self.x]
            r = self.state[(self.state[self.x] + self.state[self.y]) & 0xFF]
            output[i] = chr(ord(input[i]) ^ r)
        return ''.join(output)

if __name__ == '__main__':
    test_vectors = [['Key', 'Plaintext'], \
                    ['Wiki', 'pedia'], \
                    ['Secret', 'Attack at dawn']]
    for i in test_vectors:
        print WikipediaARC4(i[0]).crypt(i[1]).encode('hex').upper()
```

2.1.2 Algoritma SHA

SHA singkatan dari *Secure Hash Algorithm*. Algoritma ini sebagai fungsi hash. Fungsi hash memiliki sifat searah, sehingga sering disebut *one way hash function*.



Gambar 2.5 Fungsi hash secara umum

Sembarang pesan *M* berukuran bebas dikompresi oleh fungsi *hash H* melalui persamaan

$$h = H(M)$$

Sifat-sifat fungsi *hash* adalah sebagai berikut

1. Fungsi *H* dapat diterapkan pada blok data berukuran berapa saja.
2. *H* menghasilkan nilai (*h*) dengan panjang tetap (*fixed-length output*).
3. H(x) mudah dihitung untuk setiap nilai x yang diberikan.
4. Untuk setiap h yang dihasilkan, tidak mungkin dikembalikan nilai x sedemikian sehingga H(x) = h. Itulah sebabnya fungsi H dikatakan fungsi hash satu-arah (one-way hash function).
5. Untuk setiap x yang diberikan, tidak mungkin mencari y ≠ x sedemikian sehingga H(y) = H(x).
6. Tidak mungkin mencari pasangan x dan y sedemikian sehingga H(x) = H(y).

Nilai fungsi hash satu arah biasanya berukuran kecil, sedangkan pesan berukuran besar. Fungsi hash memiliki algoritma yang iterative dan searah, yang dapat memproses pesan yang diberikan untuk menghasilkan representasi yang lebih pendek yang disebut message digest. untuk menghasilkan nilai message digest dari pesan besar, fungsi hash menerima masukan berupa isi blok pesan saat ini serta nilai hash dari blok pesan sebelumnya.

Dalam aplikasi skripsi ini fungsi hash yang dipergunakan untuk pembuatan password pada sejumlah pesan yang dimasukan, kemudian pesan tersebut diproses sehingga tidak menyerupai pesan. Fungsi hash terkadang juga disebut sidik jari (*finger print*), MIC(*message integrity check*), atau *manipulation detection code*. Supaya memenuhi syarat 'aman' secara kriptografis, harus memenuhi syarat:

1. Sulit membuat sebuah dokumen dengan yang menghasilkan sebuah hash yang sudah ditentukan sebelum dokumen itu dibuat.
2. Sulit mencari 2 buah dokumen yang menghasilkan sebuah message digest / hash yang sama.
3. Perubahan pada pesan (termasuk perubahan bit tunggal) akan berpengaruh, dengan kemungkinan yang sangat tinggi, hasil dalam intisari pesan yang seluruhnya berbeda.

2.1.2.1 Sejarah Algoritma SHA

Hingga saat ini ada lima algoritma SHA yaitu SHA-0, SHA-1, SHA-224, SHA-256, SHA-384, dan SHA-512. Varian SHA-0 dikenal dengan SHA-0 pada tahun 1993, Varian SHA-1 dikenal dengan SHA-1 pada tahun 1995, varian SHA-224, SHA-256, SHA-384, dan SHA-512 dikenal dengan SHA-2 pada tahun 2002.

SHA adalah fungsi hash (*Message Digest*) satu arah yang dibuat oleh NIST, dirancang oleh NSA (National Security Agency) dan digunakan bersama DSS (Digital Signature Standard). Oleh NSA, SHA dinyatakan sebagai standard fungsi hash satu-arah.

SHA-1 dikembangkan dalam beberapa aplikasi keamanan dan protokol digunakan secara luas, mencakup TLS dan SSL, PGP, SSH, S/MIME, dan IPsec. SHA-1 dianggap sebagai turunan dari MD5, dikenal lebih awal dari MD5, fungsi hash digunakan secara luas.

2.1.2.2 Uraian Algoritma SHA-256



Gambar 2.6 Fungsi Hash Dengan SHA-256

Algoritma SHA-256 dapat digunakan untuk menghitung nilai *message digest* dari sebuah pesan, dimana pesan tersebut memiliki panjang maksimum 2^{64}

bit. Algoritma ini menggunakan sebuah *message schedule* yang terdiri dari 64 elemen 32-bit *word*, delapan buah variabel 32-bit, dan variabel penyimpan nilai *hash* 8 buah *word* 32-bit. Hasil akhir dari algoritma SHA-256 adalah sebuah *message digest* sepanjang 256-bit.

Preprocessing dilakukan dengan menambahkan bit pengganjal, membagi-bagi pesan dalam block berukuran 512-bit, dan terakhir menginisiasi nilai hash awal. Misalkan panjang pesan M adalah l bit. Proses penambahan bit pengganjal adalah pertama tambahkan bit "1" pada akhir pesan, diikuti dengan bit "0" sejumlah k, dimana k adalah solusi dari persamaan:

$$l+1+k=448 \bmod 512$$

Kemudian tambahkan 64-bit block yang menyatakan panjang pesan semula (*l*) dalam representasi biner.

2.1.2.3 Deskripsi Algoritma SHA-256

Dalam proses komputasinya,SHA-256 menggunakan enam fungsi logika, dimana setiap fungsi beroperasi menggunakan tiga buah *word* 32-bit (x, y, dan z), dan keluarannya berupa sebuah *word* 32-bit. Berikut ini adalah fungsi-fungsi dalam SHA-256:

CH(x, y, z) = (x AND y) XOR ((NOT x) AND z)

MAJ(x, y, z) = (x AND y) XOR (x AND z) XOR (y AND z)

$\sum_0^{256}(x)$ = ROTR^2(x) XOR ROTR^13(x) XOR ROTR^22(x)

$\sum_1^{256}(x)$ = ROTR^6(x) XOR ROTR^11(x) XOR ROTR^25(x)

$\sigma_0^{256}(x)$ = ROTR^7(x) XOR ROTR^18(x) XOR SHR^3(x)

$\sigma_1^{256}(x)$ = ROTR^17(x) XOR ROTR^19(x) XOR SHR^10(x)

Nilai *hash* awal pada algoritma SHA-256 adalah sebagai berikut:

H0[0] = 6a09e667

H0[1] = bb67ae85

H0[2] = 3c6ef372

H0[3] = a54ff53a

H0[4] = 510e527f

H0[5] = 9b05688c

H0[6] = 1f83d9ab

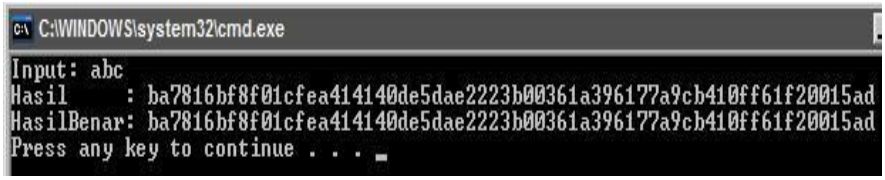
H0[7] = 5be0cd19

Dan konstanta dalam SHA-256 adalah sebagai berikut (64 buah):

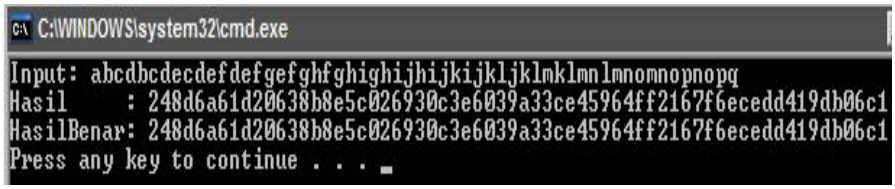
428a2f98 71374491 b5c0fbcf e9b5dba5 3956c25b 59f111f1 923f82a4 ab1c5ed5
d807aa98 12835b01 243185be 550c7dc3 72be5d74 80deb1fe 9bdc06a7 c19bf174
e49b69c1 efbe4786 0fc19dc6 240ca1cc 2de92c6f 4a7484aa 5cb0a9dc 76f988da
983e5152 a831c66d b00327c8 bf597fc7 c6e00bf3 d5a79147 06ca6351 14292967
27b70a85 2e1b2138 4d2c6dfc 53380d13 650a7354 766a0abb 81c2c92e 92722c85
a2bfe8a1 a81a664b c24b8b70 c76c51a3 d192e819 d6990624 f40e3585 106aa070
19a4c116 1e376c08 2748774c 34b0bcb5 391c0cb3 4ed8aa4a 5b9cca4f 682e6ff3
748f82ee 78a5636f 84c87814 8cc70208 90befffa a4506ceb bef9a3f7 c67178f2

2.1.2.4 Pengujian Algoritma SHA-256

Pengujian algoritma SHA-256, yaitu dengan string "*abc*" dan string "*abcdbcdecdefdefgefghfghighijhijkijkljklmklmnlmnomnopnopq*". Akan terlihat seperti gambar dibawah ini.



Gambar 2.7 Pengujian SHA-256 Pertama



Gambar 2.8 Pengujian SHA-256 Kedua

Dengan pengujian diatas dapat disimpulkan bahwa berapapun string yang dimasukan akan menghasilkan hasil yang berbeda. Maka dari itu perubahan dalam satu huruf saja akan menghasilkan perbedaan satu yang lainnya.[7]

2.1.2.5 Implementasi

Fungsi Transform() pada SHA-256 menerima dua parameter masukan, yaitu block pesan yang akan dikomputasi, dan panjang block pesan tersebut.

Untuk membangkitkan nilai message digest, dilakukan proses komputasi yang melibatkan 64 putaran untuk tiap block. Implementasinya adalah sebagai berikut:

```
void SHA256::Transform(unsigned
char* p_Message, unsigned int
p_BlockNum)
{
   unsigned int t_W[64];
   unsigned int t_WV[8];
   unsigned char* t_SubBlock;
   for (unsigned int i = 1; i <=
   p_BlockNum; i++)
   {
      t_SubBlock = p_Message + ((i - 1) << 6);
      for (int j = 0; j < 16;j++)
      {
         PACK32(&t_SubBlock[j << 2], &t_W[j]);
      }
      for (int j = 16; j < 64;j++)
```

```
        {
            t_W[j] =  SHA256_F4(t_W[j - 2]) + t_W[j - 7] + SHA256_F3(t_W[j - 15])
+ t_W[j - 16];
        }
        for (int j = 0; j < 8; j++)
        {
            t_WV[j] = m_H[j];
        }
        for (int j = 0; j < 64;j++)
        {
            unsigned int t_Temp1 =t_WV[7] + SHA256_F2(t_WV[4]) +
CH(t_WV[4], t_WV[5], t_WV[6]) + m_K[j] + t_W[j];
        unsigned int t_Temp2 = SHA256_F1(t_WV[0]) + MAJ(t_WV[0], t_WV[1],
t_WV[2]);
            t_WV[7] = t_WV[6];
            t_WV[6] = t_WV[5];
            t_WV[5] = t_WV[4];
            t_WV[4] = t_WV[3] + t_Temp1;
```

```
            t_WV[3] = t_WV[2];
            t_WV[2] = t_WV[1];
            t_WV[1] = t_WV[0];
            t_WV[0] = t_Temp1 + t_Temp2;
        }
        for (int j = 0; j < 8; j++)
        {
            m_H[j] += t_WV[j];
        }
    }
}
```

2.1.2.6 Perbandingan SHA-256 Dengan Algoritma Hash Lainnya

Dengan menggunakan perbandingan dibawah ini sekiranya dapat menyimpulkan dalam pemilihan algoritma hash untuk kedepannya. Dengan tujuan memperbaiki algoritma yang kurang memenuhi standar keamanan dalam hal ini sebagai fungsi hash.

Tabel 2.1 Daftar Informasi Algoritma Fungsi Hash [7].

Algoritma	Output size	Internal	Block	Length	Word	Collision
HAVAL	256/224/192/160/128	256	1024	64	32	Yes
MD2	128	384	128	No	8	Almost
MD4	128	128	512	64	32	Yes
MD5	128	128	512	64	32	Yes
PANAMA	256	8736	256	No	32	With flaws
RIPEMD	128	128	512	64	32	Yes
RIPEMD-	128/256	128/256	512	64	32	No
RIPEMD-	160/320	160/320	512	64	32	No
SHA-0	160	160	512	64	32	Yes
SHA-1	160	160	512	64	32	With flaws
SHA-256/224	256/224	256	512	64	32	No
SHA-512/384	512/384	512	1024	128	64	No
Tiger(2)-	192/160/128	192	512	64	64	No
VEST-4/8 (hash	160/256	176/304	8	80	1	No
VEST-16 (hash	320/512	424/680	8	88	1	No
WHIRLPOOL	512	512	512	256	8	No

Penjelasan diatas menggambarkan beberapa algoritma beserta output size, internal, block, Length, word dan Collision. Algoritma tersebut memiliki kelebihan masing-masing dan terdapat kekurangan pada algoritma tertentu karena terjadi tabrakan/bentrok (*collision*).

2.2 *USB Flash Drives*

Bicara tentang *usb (universal serial bus) flash drive,* sangat erat hubungannya dengan *flash disk* atau media penyimpanan yang dapat dibawa kemana saja. Pengertian *flash disk* adalah suatu alat media penyimpanan yang dihubungkan dengan *usb connector. Flash disk* secara tipikal dapat dipindahkan dan di program ulang, panjangnya 1 sampai 4 inci atau 25 sampai 102 mm, dan beratnya kurang dari 2 ons (56gram). Kapasitas *flash disk* berkisar antara 64 MB

sampai 32 GB atau bahkan lebih. *Flash disk* memungkinkan menullis satu juta data atau bahkan menghapus data dan dapat menyimpan data selama 10 tahun, dapat dikoneksikan dengan USB 1.1, USB 2.0 atau keduanya. Gambarnya seperti dibawah.



Gambar 2.9 *Flash disk / Flash drive*

Flash disk menawarkan banyak keuntungan pada alat media penyimpanan yang dapat dibawa kemana saja, sepertihalnya *floppy disk.* Keunggulan *flash disk* adalah lebih ringkas, lebih cepat, mampu menampung banyak data, lebih handal karena tidak ada bagian yang dipindahkan dan mempunyai desain yang lebih tahan lama. Apalagi pada masa sekarang ini semakin banyak digunakan oleh para pengguna komputer. *Flash disk* dapat digunakan dalam berbagai operasi sitem seperti Windows, Linux, Mac OS, dan sistem operasi unik yang lainnya. Masukan usb sudah terdapat pada setiap komputer dan laptop pada masa sekarang ini. *Flash disk* sudah menjadi umum maksudnya dapat digunakan oleh siapapun tanpa harus memiliki atau mengetahui komputer.

Secara otomatis media *flash disk* tampil pada saat dikoneksikan dengan sebuah masukan usb pada komputer tanpa harus mengatur untuk dimunculkan. Hal seperti Itu disebut "*drive*" karena tampil pada operasi sistem komputer yang mengidentifikasi *flash disk* tersebut.

Flash disk terdiri dari papan rangkain yang disertai komponen-komponen dengan lapisan plastik atau logam pada bagian luarnya dan baru-baru ini menggunakan bahan karet pada bagian luarnya untuk meningkatkan ketahanan pada *flash disk* tersebut, serta agar lebih kokoh untuk dibawa dan disimpan dalam saku, sebagai contoh sebuah gantungan kunci, atau digantung dengan sebuah tali. Hanya *usb connector protrudes*, dan itu untuk melindungi satu tipikal yang manapun, baik oleh satu selubung yang dipindahkan atau dengan cara menarik kembali ke badan dari *drive*. Sebagian besar *flash drive* menggunakan tipe sebuah *usb connection* mengizinkan mereka untuk menghubungkan secara langsung pada satu port di komputer pribadi.

2.2.1 Teknologi *Usb Flash Drive*

Flash memory adalah satu kombinasi dari sejumlah teknologi lebih tua, dengan murah, konsumsi daya rendah dan ukuran kecil sedang memungkinkan oleh perkembangan saat ini di bidang teknologi mikroprosesor. Tempat penyimpanan memory menjadi dasar di atas EPROM dan teknologi EEPROM yang lebih awal. Ini telah sangat membatasi kapasitas, sangat lambat untuk

keduanya, yaitu baca-tulis, tegangan tinggi yang kompleks diperlukan untuk merangkai *drive*, dan hanya dapat *re-written* setelah menghapus keseluruhan dari isi chip. Berikutnya EEPROM mengembangkan dimana daerah penghapusan atau menghancurkan menjadi "bidang" lebih kecil yang dihapus bisa menjadi dihapus secara individu tanpa mempengaruhi lainnya. Mengubah isi satu lokasi memori tertentu melibatkan pertama mengcopy keseluruhan bidang ke dalam satu memori '*buffer*' terlepas dari-chip, menghapus bidang, *re-writing* data kembali ke bidang sama, membuat perubahan yang diperlukan pada lokasi memori relevan sementara. Ini memerlukan dukungan komputer yang spesifik, dan EEPROM PC-Based sistem *memory flash* sering membawa sistem mikroprosesor khusus milik mereka sendiri. *Flash drive* adalah kurang lebih satu versi miniaturized dalam hal ini. Pengembangan antar muka (*interface*) data serial kecepatan tinggi seperti usb untuk pertama kali dibuat pada sistem memori tempat penyimpanan yang berturutan yang memungkinkan untuk diakses, dan pengembangan simultan kecil, kecepatan tinggi, sistem mikroprosesor kuat-rendah memungkinkan hal ini untuk disatukan ke dalam sistem yang benar-benar ringkas. Akses serial juga sangat mengurangi jumlah koneksi elektrik yang diperlukan untuk memori chip, yang telah memungkinkan manufaktur sukses membuat kapasitas *multi-giga byte*. (Setiap koneksi elektrik eksternal adalah satu sumber kegagalan manufaktur potensial, manufaktur tradisional, suatu titik dengan cepat dicapai dimana pendekatan hasil sukses nol). Sistem *memory flash* modern adalah suatu akses yang sangat mirip dengan hard disk, dimana sistem pengontrol mempunyai kendali penuh atas informasi yang mana benar-benar menyimpan. Penulisan

EEPROM dan proses penghapusan aktual, bagaimanapun masih persis seperti sistem lebih awal yang digambarkan di atas. Banyak *MP3 Players* sederhana yang hanya menambahkan perangkat lunak ekstra pada *memory flash* standar mengendalikan mikroprosesor sehingga itu bisa juga bertindak sebagai pemutar musik dekoder. Kebanyakan *MP3 Players* ini mungkin juga digunakan sebagai konvensionil *flash drive*.

2.2.2 Sejarah *Usb Flash Drive*

Dalam sejarah *usb flash drive* sampai saat ini terbagi kedalam 2 generasi dalam penemuannya yaitu, generasi pertama dan generasi kedua. Pada kedua generasi tersebut meiliki perbedaan yang jauh dalam hal kapasitas untuk menyimpan data dan kecepatan dalam mengakses.

2.2.2.1 Generasi Pertama dan Pemasaran



Gambar 2.10 Flash Drive Generasi Pertama.

Beberapa perusahaan mengklaim telah menemukan *usb flash drive*. Perjalanan perusahaan pertama untuk menjual *usb flash drive* adalah ThumbDrive di awal 2000. Bagaimanapun, hak paten mereka tidak menggambarkan usb *flash*

drive dengan cukup, itu menggambarkan alat tempat penyimpanan yang sangat besar yang bisa meliputi *usb flash drive*.

Perusahaan Israeli M-Systems (diakui oleh SanDisk pada bulan November 2006) masih sedang mengerjakan *usb flash drive* sejak 1998. Mereka mendaftarkan domain www.diskonkey.com pada tanggal 12 Oktober, 1999 menandakan niat mereka untuk menjual produk. Pada tahun 2000 Dan Harkabi bergabung dengan tim M-Systems dan memimpin pengembangan DiskOnKey. Desain industri dilakukan oleh Ziba dan produk memenangkan penghargaan IDE pada tahun 2001. hak paten dimiliki oleh M-Systems dengan kaku menggambarkan *usb flash drive* dan implementasi nya.

Menyingkapan dalam penemuan IBM RPS8-1999-0201 (Sept. 99) oleh Shimon Shmueli, adalah dokumen yang dikenali paling awal untuk menggambarkan USB-FD dengan teliti dan keseluruhan, dan hanya USB-FD. M-Systems melakukan manufaktur DiskOnKey untuk IBM, di akhir 2000 pertama untuk menjual produk di Amerika Utara. IBM 8 MB (8 MiB) USB Memory Key tersedia 15 Desember, 2000. Shmueli kemudian KeyNetica didirikan, perusahaan pertama untuk mempatenkan dan mengembangkan konsep alat tempat penyimpanan untuk telepon genggam yang cerdas dan semua kebutuhan untuk komputasi telepon genggam. Pelaksanaan saat ini dari konsep tersebut adalah U3 (bagian dari SanDisk, yang juga memiliki KeyNetica asli untuk mempatenkan) dan Ceedo.

Melakukan perjalanan teknologi yang diakui pertama kalinya untuk mengembangkan dan menciptakan ThumbDrive. Perjalanan memegang hak paten untuk ThumbDrive di Jepang, Taiwan, Korea Selatan, Inggris, Selandia Baru serta Singapura. Meskipun demikian, kepemilikan hak paten untuk alat ini secara luas diperdebatkan. Sesuai dengan laporan The Straits Times, perusahaan lain mulai melakukan pemasaran alat yang serupa. M-Systems, terdaftar pada NASDAQ pada waktu itu, perangkat DiskOnKey dan Diskey. Electec adalah importir M-Systems, dan FE Global adalah distributor tunggalnya di Singapura. Lexar bisa juga menuntut satu mempelopori produk *usb flash drive*. Pada tahun 2000 mereka memperkenalkan Compact Flash (CF) kartu mempunyai fungsi sebagai usb internal. Lexar menawarkan *card reader* dan kabel usb dimana mengeliminasi kebutuhan akan pusat (hub) usb.

Perjalanan untuk menggugat 4 perusahaan yang melanggar hak patennya. Mereka menggugat kembali Trek, dan menanyakan hak paten yang dimiliki Trek telah ditarik kembali dan tentu saja itu tidak sah.

Singapura Court Appeal menetapkan hak paten kebenaran dari Trek Technology untuknya ThumbDrive, pemanggilan itu dinamakan "*novel and inventive*" dalam keputusan yang diterbitkan di Straits Times. Pengadilan tertinggi negara juga membatalkan permohonan dari empat perusahaan—Israeli memastikan M-Systems Flash Disk Pioneers, Electec, FE Global Electronics dan Singapore-based Ritronics Components dan memesan mereka untuk menghentikan penjualan alat yang serupa. Keputusan diharapkan mempunyai efek riak terhadap penuntutan perkara serupa yang lain, dimana kelompok Trek telah

menunggu keputusan di Inggris, Jepang dan Taiwan. Sedikitnya keputusan di Inggris telah menangani titik serupa itu sebagai penanganan oleh Singapura Court Appeal, tetapi petugas mencapai kesimpulan berbeda dari Singapura Court Appeal, dan hak paten Trek ditarik kembali ke Inggris.

2.2.2.2 Generasi Kedua



Gambar 2.11 Flash Drive Generasi Kedua (Toshiba TransMemory).

Keadaannya belum diinstall dengan U3, membiarkan para pemakai untuk menangani aplikasi mereka, secara penuh diinstall dan operasional, pada desktop apapun. Modern *flash drives* mempunyai USB 2.0 konektivitas. Bagaimanapun, sekarang ini mereka tidak menggunakan 480 Mbit/s sehubungan dengan keterbatasan dukungan spesifikasi teknis tak yang terpisahkan NAND flash. *Drives* paling cepat sekarang ini tersedia menggunakan pengontrol jalur rangkap, walaupun mereka masih jatuh dengan penuh kekurangan pertimbangan *transfer rate* mungkin dari harddisk generasi saat ini, atau maksimum usb kecepatan tinggi.

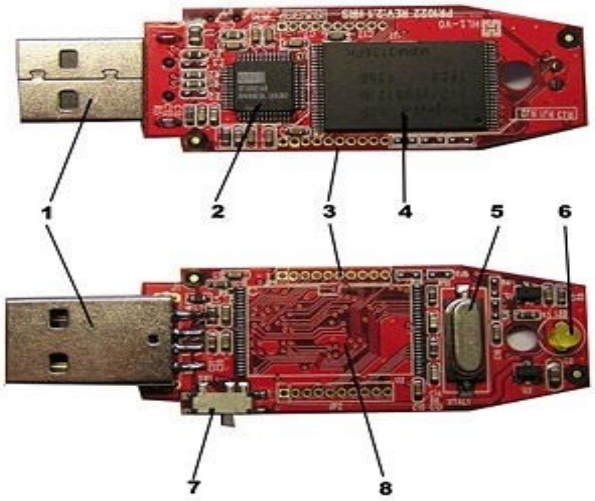
Tipikal keseluruhan pengiriman berkas (file transfer) kecepatannya adalah sekitar 3 MB/s. Arus tertinggi keseluruhan pengiriman berkas kecepatannya

adalah sekitar 10-60 MB/s. Lebih tua, "*full speed*" 12 Mbit/s adalah membatasi alat pada kecepatan maksimum sekitar 1 MB/s.

2.2.3 Desain dan Implementasi

Salah satu ujung alat dipasangkan dengan jenis tunggal atau *usb connector*. Di dalam casing plastik ada papan rangkaian dicetak kecil. Puncak dari papan ini, beberapa kekuatan kecil menggerakan rangkaian dan sejumlah kecil rangkaian terintegrasi di puncak permukaan atau *surface-mounted integrated circuits* (ICs). Secara tipikal, ICs ini menyediakan menghubung ke *usb port*, *flash memory* mengendalikan memori dalam papan tersebut.

2.2.3.1 Komponen-Komponen Penting



Gambar 2.12 Komponen-Komponen Flash Drive.

Secara tipikal terdapat empat bagian-bagian pada *flash drive*:

1. Male type-A USB connector — menyediakan menghubung ke komputer(host).
2. USB mass storage controller — mengimplementasikan (host) USB pengontrol. Pengontrol berisi mikroprosesor RISC kecil dan sejumlah ROM serta RAM dalam chip kecil.
3. Test point — point pengujian.
4. Memory flash NAND chip — menyimpan data. NAND flash secara tipikal juga digunakan dalam kamera digital.
5. Crystal oscillator — alat utama yang menghasilkan sinyal clock 12 MHz dan mengontrol alat keluaran data melalui tahap pengulangan kunci.
6. LEDs — menunjukan pengiriman data atau membaca data dan menulis data.
7. Write-protect switches — alat harus menunjukan, keadaan sedang cara "menulis-melindungi".
8. Space for second flash memory chip — ruang kosong untuk *chip flash memory* kedua.

2.2.3.2 Ukuran dan Bentuk



Gambar 2.13 Bentuk *Flash Drive* Ikura Sushi.

Beberapa pabrik membedakan produk mereka dengan menggunakan kolaborasi dengan alat perumahan, kadang sering besar sekali ukurannya dan membuat drive sulit untuk dihubungkan ke *usb port*. Karena *usb connector port* pada komputer rumah jaraknya sering berdekatan, memasukan flash drive ke dalam *usb port* dapat menghalangi *usb port* sebelahnya. Beberapa alat hanya dapat membawa logo usb jika menjual dengan bagian kabel terpisah.

Usb flash drive telah di integrasikan ke perangkat lain *(tools),* pada umumnya *usb flash drive* dibawa oleh seseorang seperti layaknya jam tangan, pena, pisau Swiss Army. Ukuran yang kecil, tidak mengganggu alam, dan *usb flash drive* secara relatif harganya murah membuat flash drive itu sendiri meningkat menjadi popular.

Kelebihan kapasitas atau tidak sesuai pengemasan pada *flash drive* bisa menyebabkan pemutusan hubungan dari komputer (*host*). Hal ini mungkin mengatasi dengan menggunakan kabel perluasan untuk membebaskan tekanan

terhadap *port*. Beberapa kabel USB-compatible, tetapi jangan menyesuaikan pada USB 1.0 standar.

2.2.3.3 *File System*

Sebagian besar flash drive berformatkan FAT atau FAT32. File system ini memungkinkan untuk bisa mengakses pada komputer (*host*). Secara virtual file yang terdapat pada *flash drive* baik rusak ataupun hilang data dapat dikembalikan kembali. Karena *flash drive* akan tampil pada komputer sebagai harddisk dengan koneksi *usb port.*

2.2.4 Penggunaan Umum

Kegiatan yang sering dilakukan dengan menggunakan flash *drive* yaitu:

- Pemindahan data pribadi.
- Mengamankan tempat penyimpanan data, aplikasi, dan perangkat lunak.
- Sebagai sistem administrator dalam hal menduplikasi data.
- Sebagai media penyimpanan sementara apabila komputer dalam masa perbaikan, baik dari virus atau kerusakan pada sistem operasi.
- Membawa aplikasi dengan mudah.
- Sebagai pelaksanaan hukum *Computer Online Forensic Evidence Extractor* (COFEE) yang dikeluarkan Microsoft tentang membawa aplikasi yang tidak resmi.

- Untuk *booting* (membuka data) operasi sistem.
- Untuk *audio players* seperti iPod Shuffle karena berdasarkan pada *usb flas drive*.
- Sebagai tempat menyimpan musik.
- Untuk membawa promosi produk.
- Untuk membackup(menduplikasi).

2.2.5 Keuntungan dan Kerugian *Flash Drive*

Keuntungan:

- Hampir tidak dapat dimasuki debu, tidak seperti compact disc, Zip-Disk, dan disket.
- Di desain padat serta tahan lama.
- Harga relatif murah.
- Kapasitas lebih besar dari pada sebuah DVD (4.7 GB) sedangkan pada saat ini *flash drive* sudah mencapai 64 GB.
- Konsumsi daya sangat rendah dari sebuah flash drive.
- Alternatif lain selain harddisk, dalam hal ini *flash drive* sering juga disebut hardisk portable.
- Tidak ada bagian yang bergerak, lain dengan harddisk yang harus merlukan beberapa motor untuk berputar sehingga dapat dibaca.

- *USB Flash drive* termasuk kelas alat media penyimpanan, mempunyai pengertian bahwa sistem operasi yang paling modern bisa membaca dan menulis untuk menbuka *drive* tanpa alat tambahan.
- Dapat digunakan sebagai sistem operasi.
- *Flash drive* jauh lebih toleran dari penyalahgunaan dibandingkan harddisk, tetapi masih bisa dirusak atau mempunyai data yang rusak akibat dari dampak fisik yang parah.
- Beberapa *flash drive* bisa mempertahankan memori setelah terendam di dalam air, atau melalui pencucian mesin. Hanya tinggal menunggu *flash drive* mengering seluruhnya sebelum dimasukan ke sebuah *usb port.*

Kerugian:

- Sebuah kelemahan pada ukurannya yang kecil karena salah meletakkan, meninggalkan, atau bahkan terjadi kehilangan *flash drive*.
- Sebagai perantara dalam hal penyebaran virus.
- Mahalnya komponen setiap unit dala media penyimanan ini dibandingkan dengan harddisk.
- Kapasitas terbatas, usb flash drive paling besar baru mencapai sekita 64 GB, sedangkan harddisk 10 kali lipat lebih besar dibandingkan *usb flash drive* atau bahkan lebih.

2.3 Bahasa Pemograman

Pada era globalisasi bahasa pemograman untuk bidang ilmu komputer sudah menjadi pengetahuan umum yang dapat dikenal oleh orang banyak. Bahasa pemograman terdiri dari beberapa macam yaitu pascal, C, C++, dan lain-lain. Bahasa pemograman tersebut diimplementasikan pada suatu program untuk membangun sebuah perangkat lunak yang fungsional. Beberapa bahasa pemograman yang diimplementasikan pada suatu program yang sering digunakan diataranya Delphi dan Visual Basic.

2.3.1 Sejarah Delphi

Berbicara mengenai bahasa pemrograman Delphi, rasanya sulit dipisahkan dari keberadaan nenek moyangnya, yaitu Turbo Pascal. Dikatakan demikian karena Delphi memang merupakan generasi lanjutan dari Turbo Pascal (yang diluncurkan tahun 1983 oleh *Borland International Incorporation*).

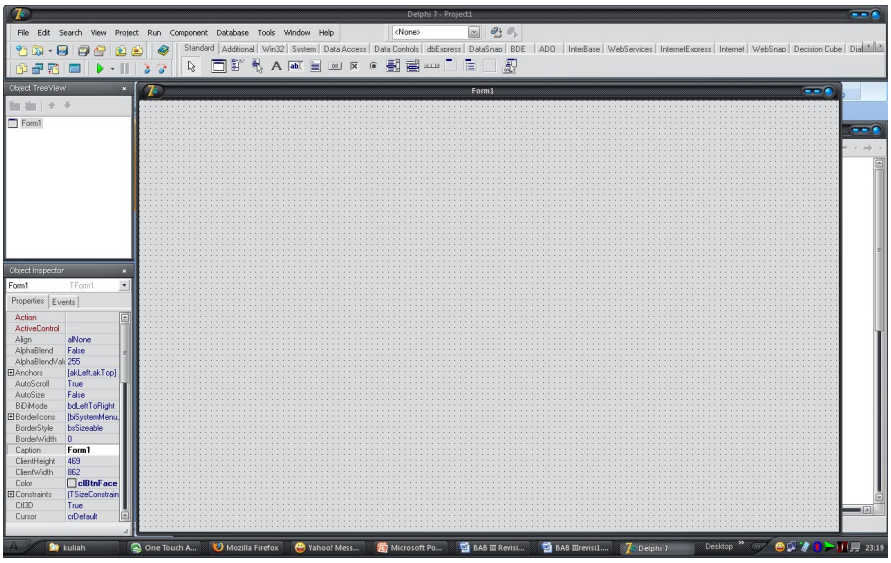
Sesuai dengan sistem operasi yang populer di masa itu, Turbo Pascal memang dirancang untuk dijalankan pada sistem operasi DOS. Kemudian seiring dengan perkembangan yang ada, *Borland International Incorporation* merilis Turbo Pascal untuk sistem operasi Windows, versi ini dijalankan pada Windows 3.X.

Penggabungan Turbo Pascal dengan Turbo Pascal For Windows, pada tahun 1992, melahirkan suatu bahasa pemrograman baru yang diberi nama Borland Pascal Versi 7. Namun masih terdengar keluhan mengenai sulitnya menggunakan bahasa pemrograman ini, sehingga mendorong pihak Borland untuk

membuat bahasa pemrograman visual yang berbasis bahasa Pascal. Hasilnya, pada tahun 1995 Borland Delphi diluncurkan kepasar pengguna komputer. Menyusul setahun kemudian dirilis Borland Delphi versi 2.0, versi 3.0 dan sampai dengan kini Borland Delphi versi 8.0.

2.3.2 Mengenal Delphi

Pada saat pertama kali menjalankan Delphi, maka sebuah jendela akan terbentuk atau Delphi akan menampilkan sebuah proyek baru, dan pada proyek ini terdapat sebuah *form* yang dapat dirancang dan disusun program aplikasi. Inilah yang disebut IDE (*Integrated Development Environment)*. Melalui IDE inilah pemrogram secara visual dapat merancang tampilan untuk pemakai (antarmuka pemakai) dan menuliskan programnya.



Gambar 2.14 Antarmuka Bahasa Pemrograman Delphi 7.0

2.3.2.1 Mengenal IDE Delphi

Ada terdapat banyak bagian pada IDE (*Integrated Development Environment*). Bagian-bagian IDE antara lain adalah *menu*, s*peedbar*, *component palette*, *object inspector*, *form designer*, *editor code*.

a. Menu

Semua perintah yang diperlukan selama merancang dan membangun program aplikasi tersedia dalam *menu bar*, yang terletak di bagian atas jendela utama Delphi. Apabila membuka sebuah *project*. Delphi menyediakan menu-menu: *File*, *Edit*, *Search*, *View*, *Project*, *Run*, *Component*, *Database*, *Tool*, dan menu *Help*.

b. Speedbar

Speedbar merupakan sekumpulan tombol yang digunakan untuk mengakses beberapa perintah dalam menu. Biasanya yang tersedia pada *speedbar* adalah perintah-perintah yang umum yang digunakan dalam proses perancangan program aplikasi. Misalnya perintah untuk membuka *project*, membuka *file*, menyimpan *file*, menjalankan program aplikasi dan sebagainya.

c. Component Palette

Component Palette adalah *tool* yang berupa kumpulan *tab*, dimana setiap *tab* atau halaman memuat berbagai tombol komponen yang digunakan sebagai elemen antarmuka program aplikasi.

d. Object Inspector

Object Inspector merupakan penghubung antara tampilan aplikasi dengan kode program yang menjadikan aplikasi yang dibuat dapat berjalan. Secara garis besar terdapat dua hal yang dapat dilakukan dalam *object inspector*, yakni masing-masing adalah penetapan *properti* bagi komponen-komponen dalam *form* (termasuk penetapan properti bagi *form* itu sendiri), dan penetapan prosedur-prosedur penanganan suatu kejadian.

e. Form

Form adalah jendela yang merupakan komponen dasar dari sebuah aplikasi. Selama perancangan dan penyusunan aplikasi, meletakkan sebuah komponen ke dalam *form*, membentuk antarmuka program aplikasi. *Form* dapat berfungsi sebagai jendela bagi program aplikasi atau sebuah kotak dialog.

f. Editor Code

Editor Code merupakan jendela editor yang penuh dengan rencana. Editor ini dapat mengakses dan memodifikasi kode-kode program aplikasi. Pada editor ini merupakan *reserved word*-nya Delphi.

2.3.2.2 Struktur Program Delphi

Program aplikasi yang dibuat dengan menggunakan Delphi tersimpan sebagai sebuah *project*. *Project* ini merupakan kumpulan dari *file* unit yang menyusun aplikasi. *Project* aplikasi Delphi terdiri dari :

1. *File Project*

File project ini tersimpan dalam ekstensi .DPR. Dalam satu *project* aplikasi hanya bisa tedapat satu *file project*.

2. *File Form*

File form merupakan *file* dengan ekstensi .DFM. *File* ini adalah merupakan *file binary* yang berisi gambar grafis dari *form*. Setiap *file* .DFM mempunyai pasangan sebuah *file* Unit (dengan ekstensi .PAS).

3. *File Unit*

File Unit (*File Source Code)* tersimpan sebagai *file* dengan ekstensi .PAS. Setiap *file* unit merupakan sebuah *file source code* Objek Pascal, tetapi *file* unit yang merupakan pasangan dari *file form* berbeda dengan tipe *file* unit yang lain, karena *file* ini tidak dapat dipisahkan dari *file form* .DFM-nya.

a. Project

Setiap memulai *project* baru, Delphi akan membentuk sebuah *file project* dan akan meng-*update*-nya selama proses pembuatan *project* tersebut. *File* inilah yang akan mengontrol suatu aplikasi Delphi, karenanya *file project* disebut juga sebagai program utama dari aplikasi. *File project* dapat dikenali dari ekstensinya, yaitu .DPR. Didalamnya tercatat unit-unit apa saja yang digunakan oleh suatu *project*.

Untuk *project* yang bernama *project*1, dimana di dalamnya terdapat *form* bernama *form*1 (yang *source code*-nya tersimpan sebagai *file* unit1.pas), Delphi membuat *source code*-nya seperti di bawah ini.

```
program Project1

uses

    Forms,

 Unit1 in 'Unit1.pas' {Form1};

                        {$R*.RES}

begin

Application.create(Tform, Form1);

Application.Run(Form1);

end.
```

File ini dimulai dengan reserved word program, diikuti nama *file* (dimana pada contoh diatas *project* ini bernama ***Project*1**). Setiap kali menambahkan *form* atau unit baru kedalam *project*, secara otomatis delphi akan menambahkannya kedalam *file project*, tepatnya dibagian **uses**.

b. *Form*

Form adalah titik utama dari suatu aplikasi Delphi. Meletakan kedalam *form*, meng-*edit properti*-nya atau menuliskan kode, itu semua merupakan proses *editing form*. *Form* tersimpan dalam dua buah *file*, yaitu:

1. *File form* (.DFM), menyimpan image binary dari *form*. Semua perubahan yang dilakukan pada properti *form*, seperti mengubah tinggi, lebar, warna, *border*, dan lain-lain yang berkaitan dengan tampilan *form*, oleh delphi disimpan dalam *file* .DFM.
2. *File* Unit (.PAS), menyimpan source kode dari *file* .DFM. Didalam *file* .PAS inilah dituliskan kode-kode event-handler, yang menentukan tentang bagaimana *form* serta komponen-komponen didalamnya harus bereaksi saat aplikasi dijalankan.

Delphi selalu menjaga *file* .DFM dan *file* .PAS untuk tetap sinkron selama proses penyusunan serta modifikasi *form*. Bila menambahkan *form* baru dalam *project*, secara otomatis delphi akan membuat *file* unit untuk *form* baru tersebut, dan menambahkannya kebagian *uses* dalam *file project*.

c. Unit

Unit dalam blok utama pembentuk aplikasi delphi. *File-file* unit ini berisi kode sumber bagi elemen-elemen aplikasi delphi yang dibuat. Ada *file* unit merupakan pasangan *file form* .DFM, dan ada pula *file* unit milik suatu *project*.

Tetapi bisa juga membuat dan menyimpan *file* unit sebagai *file stand-alone* yang dapat digunakan oleh beberapa *project* sekaligus.

Jadi tidak semua *file* unit merupakan pasangan *file* unit .DFM. Jika membuat sendiri prosedur, fungsi, atau lain-lain, maka *source code*-nya (kode sumber) juga akan disimpan dalam bentuk *file* .PAS. Tapi *file* .PAS semacam ini tidak terkait sedikitpun dengan *file form* .DFM.

Selama pembuatan aplikasi, dapat menambahkan *form-form* dan *file-file* unit yang terkait. Selain itu anda juga dapat menambahkan:

1. *Ffile-file* yang ditulis dalam Borland Pascal versi sebelumnya, atau bahasa pemrograman lain. *File-file* ini disertakan kedalam *project* delphi dengan menggunakan *compiler directives* yang akan memberitahu delphi tipe *file* yang anda sertakan.
2. *File-file* yang tidak dibuat dengan delphi, tetapi menggunakan *library* atua *source* lain (*file-file resource non-delphi*).

File project menggabungkan semua *file* ini bersama-sama, sehingga delphi mengetahui apa yang harus dikompilasi dan di-*link* untuk membentuk *file executable* (.EXE) atau *dynamic link library*.

2.4 Sistem

Suatu sistem terdiri dari elemen-elemen yang berhubungan melalui berbagai bentuk interaksi dan bekerjasama untuk mencapai tujuan tertentu. Elemen-elemen tersebut mempunyai sifat dalam menjalankan fungsinya tidak

dapat dipisahkan antara satu dengan yang lainnya, akan saling berhubungan dan saling mempengaruhi proses secara keseluruhan.

Dalam memahami suatu sistem diperlukan pemahaman mengenai sistem itu sendiri. Dalam mendefinisikan suatu sistem, terdapat beberapa pendekatan yang sering digunakan. Pendekatan pertama dalam mendefinisikan sistem lebih menekankan pada prosedur. Pendefinisian sistem dengan penekanan prosedur sebagai berikut :

"*Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu*". (H.M Jogiyanto,1993:1)

Pendekatan kedua yang sering dipergunakan dalam mendefinisikan sistem lebih menekankan pada elemen atau komponennya. Pendefinisian sistem dengan penekanan elemen sebagai berikut :

"*Sistem adalah kumpulan dari elemen-elemen yang berorientasi untuk mencapai suatu tujuan*". (H.M Jogiyanto,1993:2)

Secara umum, sistem adalah suatu kesatuan dari dua atau lebih komponen yang berinteraksi untuk mencapai suatu tujuan. Pendekatan sistem yang menekankan pada komponen lebih mudah untuk dipelajari dan diterapkan pada berbagai bidang terutama untuk pengembangan sistem lebih lanjut.

Perancangan sistem secara umum adalah suatu tahap dimana di dalamnya terdapat identifikasi komponen-komponen sistem informasi yang akan dirancang secara rinci yang bertujuan untuk memberikan gambaran kepada pengguna atau

user mengenai sistem yang baru. Sedangkan desain sistem secara terinci dimaksudkan untuk pembuat program komputer dan ahli teknik lainnya yang akan mengimplementasikan sistem.

Penggambaran dan rancangan model sistem informasi secara logika dapat dibuat dalam bentuk Diagram Alir Data (DAD) atau *Data Flow Diagram* (DFD). Arus data pada Diagram Alir Data atau *Data Flow Diagram* dapat dijabarkan dengan menggunakan kamus data (*Data Dictionary*).

2.4.1 Diagram Konteks

Keadaan sistem secara umum dan hubungan-hubungan sistem tersebut dengan komponen-komponen diluar sistem atau dengan sistem yang lain dapat digambarkan secara logika dengan diagram konteks atau *contex diagram*. Definisi diagram konteks adalah sebagai berikut :

"Diagram konteks adalah diagram yang tidak detail dari sebuah sistem informasi yang menggambarkan aliran-aliran data masuk dan keluar dari sistem. Diagram ini digambarkan dengan sebuah lingkaran yang menjelaskan tentang batasan sistem yang saling berhubungan dengan kesatuan luar (external entity) yang akan memberikan masukan dan menerima keluaran dari sistem tersebut yang dihubungkan dengan aliran yang menghubungkan sebuah sistem dengan lingkaran sistem".(Lani Sidharta, Sistem Informasi Bisnis, 1995 : 66)

2.4.2 Diagram Aliran Data

Diagram arus data merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur. Diagram arus data ini dapat menggambarkan sistem secara keseluruhan dan merupakan representasi grafik dari sebuah sistem yang menunjukan proses dan aliran data yang ke luar ataupun ke dalam sistem, namun tidak menunjukan dimana, bagaimana dan oleh siapa proses itu dilakukan.

Definisi *Data Flow Diagram* adalah sebagai berikut :

"Data Flow Diagram adalah representasi grafik yang menggambarkan arus data dari suatu sistem. Data Flow Diagram menggambarkan komponen-komponen tersebut, asal dan tujuan dan penyimpanan data." (Lani Sidharta, Sistem Informasi Bisnis, 1995 : 65)

Simbol yang digunakan pada diagram alir data atau data flow diagram antara lain :

1. Kesatuan luar *(External Entity)*

Kesatuan luar *(External Entity)* merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi, atau sistem lain. Lingkungan luar dan batas sistem dinotasikan dalam simbol yang sama.



Gambar 2.15 Notasi Kesatuan lingkungan

2. Arus Data (*Data Flow*)

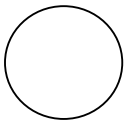
Arus data (*Data Flow*) di *DFD* diberi simbol suatu panah. Arus data ini mengalir diantara proses (*process*), simpanan data (*data store*) dan kesatuan luar (*external entity*).



Gambar 2.16 Notasi Arus Data

3. Proses (*Process*)

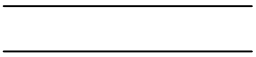
Proses adalah kegiatan yang dilakukan oleh manusia, mesin, atau komputer yang bertujuan mengolah data yang dimasukkan menjadi keluaran yang diinginkan.



Gambar 2.17 Notasi Proses

4. Simpanan Data (File)

Simpanan data disimbolkan dengan sepasang garis horizontal paralel. Simpanan data adalah tempat menyimpan atau untuk mengakses data yang diperlukan oleh sistem.



Gambar 2.18 Notasi Simpanan Data

2.4.3 Kamus Data

Kamus data merupakan kumpulan data-data. Tujuannya adalah untuk memberikan informasi mengenai definisi struktur, pemakaian masing-masing elemen. Elemen adalah unit data yang terkecil *(Tauri D. Mahyozir, Analisa dan perancangan system pengolahan data, 1995 : 44)*. Elemen-elemen yang dapat menyusun sebuah sistem tersebut, terdiri dari :

1. Tujuan, merupakan tujuan dari sistem tersebut
2. Batasan, merupakan batasan-batasan yang ada untuk mencapai tujuan dari sistem.
3. Kontrol, merupakan pengawas dari pelaksanaan pencapaian tujuan sistem
4. Input, merupakan bagian dari sistem yang memberikan data masukan ke dalam sistem.
5. Proses, merupakan bagian yang memproses masukan data menjadi informasi yang sesuai dengan keinginan penerima.
6. Output, merupakan keluaran atau tujuan akhir dari sistem.
7. Umpan Balik, merupakan elemen sistem yang mempunyai tugas untuk melihat kembali apakah sistem telah berjalan sesuai dengan yang diinginkan.

Kamus data atau *Data dictionary* berisi keterangan tentang arus data, alias, bentuk data, penjelasan, periode, volume dan struktur data yang merupakan katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi.

2.5 Microsoft Access XP

Microsoft access merupakan salah satu software yang memungkinkan kita membuat database. Dimana kita dapat mengelola seluruh data ke dalam file database. Microsoft Acces XP yang selanjutnya disingkat Acces XP merupakan salah satu program aplikasi basis data (database). Dengan menggunakan Acces XP kita dapat merancang, membuat dan mengelola database dengan mudah. Acces XP merupakan pengembangan dari Access versi sebelumnya, dengan harapan program aplikasi database ini lebih mudah dipakai, diintegrasikan dengan program aplikasi Microsoft Office XP itu sendiri ataupun dengan program aplikasi yang lainnya.

Komponen-komponen yang terlibat dalam pembuatan suatu database diantaranya adalah tabel. Table Merupakan tabel kumpulan data yang merupakan komponen utama dari sebuah database.