PERANCANGAN APLIKASI DIGITAL SIGNATURE DENGAN *AIPLER32*

Disusun dan Diajukan Untuk Memenuhi Persyaratan Ujian Akhir Dalam
Memperoleh Gelar Sarjana Komputer Pada Fakultas Sains dan Teknologi
Universitas Pembangunan Panca Budi
Medan

SKRIPSI

OLEH

NAMA	: KEVAN SEPTIAWAN
NPM	: 1514370017
PROGRAM STUDI	: SISTEM KOMPUTER

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
2019

ABSTRAK

NAMA: KIYAN SEPTIAWAN

PERANCANGAN APLIKASI *DIGITAL SIGNATURE* DENGAN *ADLER32*

TAHUN 2019/2020

Penelitian ini bertujuan untuk menjaga keaslian data teks untuk memberikan jaminan kepada penerima bahwa data tersebut bebas dari modifikasi yang dilakukan oleh pihak lain, dan jika terjadi suatu modifikasi terhadap data tersebut, maka penerima akan mengetahui bahwa data tersebut tidak terjaga lagi keasliannya. Untuk menjaga keaslian data teks digunakan Teknik *Digital Signature* dengan menggunakan *Adler32*. Data yang digunakan sebagai inputan dapat berupa data teks dengan format txt yang memiliki ukuran maksimal 100MB. Hasil penelitian menujukan bahwa dari kedua algoritma memberikan *digital signature* yang memiliki tingkat keamanan yang cukup tinggi dan dengan waktu proses yang cukup cepat.

Keywords: *Adler32, Digital Signature*. Kriptografi, Teks.

DAFTAR ISI

Halaman

KATA PENGANTAR

Bismillahirrohmanirohim

Assalamu'alaikum Wr. Wb.

Puji syukur atas kehadirat Allah SWT karena telah memberikan rahmat serta karunianya sehingga penulis dapat menyelesaikan skripsi ini dengan baik dan tepat pada waktunya. Shalawat dan salam dihaturkan kepada Nabi besar junjungan kita Nabi Muhammad S.A.W beserta sahabatnya.

Skripsi ini merupakan salah satu syarat wajib untuk menempuh gelar Strata satu (S-1) dalam menempuh kesarjanaan yang harus diselesaikan setiap mahasiswa. Oleh karena itu, penulis selaku Mahasiswa Universitas Pembangunan Panca Budi Fakultas Sains dan Teknologi mengajukan skripsi ini.

Sebagai rasa syukur karena telah selesainya skripsi dengan judul "Perancangan Aplikasi *Digital Signature* dengan *Adler32*". Penulis ingin mengucapkan banyak terimakasih kepada orang-orang yang telah membantu penulis dalam menyelesaikan penelitian skripsi ini diantaranya :

1. Kepada Bapak Dr.H. Muhammad Isa Indrawan, S.E, M.M selaku Rektor di Universitas Pembangunan Panca Budi.
2. Kepada Ibu Sri Shindi Indira, ST., M.Sc selaku dekan Fakultas Sains dan Teknologi yang telah mengikut sertakan saya dalam Seminar Proposal dan Sidang Meja hijau.
3. Kepada Bapak Eko Haryanto, S.kom., M.kom selaku Ka. Program Studi Sistem Komputer yang telah mengurus semua kebutuhan-kebutuhan mahasiswa/I dalam bidang akademik dan hal-hal yang berkaitan dengan kepentingan mahasiswa.
4. Dosen pembimbing I yaitu Bapak Andisyah Putra Siahaan, S.Kom, M,Kom, Ph.D. dan Dosen pembimbing II yaitu Ibu Ranti Eka Putri, S.Kom, M.Kom. yang telah memberikan arahan, bimbingan dan pembinaannya kepada penulis dalam menyelesaikan skripsi ini.
5. Kepada Orangtua tercinta Ibu Budi Wati dan Bapak Pairen yang telah mendukung dan memberikan curahan kasih sayang dan yang selalu menyemangati saya sampai detik ini. Kasih sayang dan senyuman yang kalian berikan membuat saya tetap semangat dalam menggapai cita-cita ini.
6. Terimakasih juga kepada Abah Suratno dan Mamak Asminah yang telah mendukung cucu nya dalam menggapai gelar kesarjanaan. Serta sering memotivasi saya dan selalu mencurahkan kasih sayang nya.
7. Terimakasih juga kepada Adik saya (Reza Riswandy) yang tak hentinya memberi saya semangat dalam menggapai cita-cita ini.
8. Kepada orang terdekat saya (Tia Monica S.Pd) yang telah menyemangati saya untuk menyelesaikan studi ini serta bersama-sama dalam menggapai gelar kesarjanaan.
9. Kepada sahabat saya (Syahrul Maulidin S.Kom, Imam Munandar Silalahi S.Kom, Syakila S.Pd, Nurul Hidayati S,Pd, Misbahul Latif Nst S.kom, dan Yudha Prasetyo) yang tak henti-henti nya memberi saya semangat serta

dukungan dalam menyelesaikan skripsi ini. Serta bersama-sama berjuang untuk mendapatkan gelar kesarjanaan. Terimakasih kepada kalian yang tak meninggalkan dan tetap bersama sampai detik ini.

Penuh harap semoga kebaikan kita semua dapat diterima Allah S.W.T dan tercatat sebagai amal shaleh.

Penulis juga menyadari masih banyaknya kekurangan dan kekhilafan dalam skripsi ini, maka dari itu penulis mengharapkan saran juga kritik yang bersifat membangun dari semua pihak demi kesempurnaan dan perbaikan skripsi ini. Akhirnya, besar harapan penulis semoga skripsi ini dapat bermanfaat bagi penulis dan bagi pembaca pada umumnya. Amin.

Medan, September 2019

Kiyan Septiawan

BAB I

PEDAHULUAN

1.1 Latar Belakang

Komunikasi merupakan faktor penting dalam kehidupan manusia. Kini manusia dipermudah oleh teknologi untuk menyampaikan informasi. Media komunikasi yang diciptakan manusia tersebut memang memudahkan dalam penyampaian informasi, tapi di sisi lain penyampaian pesan melalui media tertentu tidak menjamin keamanan terhadap integritas data. Keamanan telah menjadi aspek yang sangat penting dari suatu sistem informasi.

Menurut Aristo Jansen Sainlae, dalam jurnalnya yang Berjudul Analisis Kriptosistem Menggunakan *Digital Signature* Berbasis SHA (2012), dalam era masyarakat yang berbasis informasi saat ini, sebuah data *digital* juga merupakan komponen yang sangat vital, sehingga sangat memerlukan pengamanan yang baik saat didistribusikan atau saat disimpan.

Kemudian dikembangkan berbagai metode untuk mengatasi persoalan keamanan data dengan cara mengantisipasi supaya orang-orang yang tidak berhak tidak dapat membaca atau bahkan merusak data yang bukan ditunjukan kepadanya. Salah satu cara pengamanan data tersebut adalah dengan mengunakan kriptografi atau penyandian.

Digital signature memiliki fungsi sebagai penanda pada data yang memastikan bahwa data tersebut adalah data yang sebenarnya (tidak ada yang berubah). Dengan begitu, *digital signature* dapat memenuhi setidaknya ada dua syarat keamanan data dan teks, yaitu *Authenticity* dan *Nonrepudiation. Digital signature* dilakukan dengan menggunakan algoritma kunci-publik. Salah satunya adalah algoritma *Adler-32* dan dengan menggunakan fungsi *hash* sehingga proses pembentukan pengamanan dari pesan yang dikirim dapat dipriksa keasliannya.

Sehubungan dengan latar belakang tersebut, maka penulis akan merancang dan membangun suatu aplikasi kriptosistem menggunakan *digital signature* berbasis algoritma *Adler-32* dengan tujuan untuk memenuhi aspek-aspek keamanan dari kriptografi yaitu kerahasiaan, intergritas data, ontentikasi dan penyangkalan. Berdasarkan judul diatas tentang“ **Perancangan Aplikasi *Digital Signature* Dengan *Adler-32***” Metode ini diharapkan dapat membantu melindungi keaslian juga menjamin integritas data.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat disimpulkan rumusan masalah yaitu :

1. Bagaimana merancang aplikasi *digital signature* dengan *adler-32*?
2. Bagaimana mengimplementasikan aplikasi *digital signature* dengan *adler-32?*

1.3 Batasan Masalah

Berdasarkan latar belakang diatas, maka dapat di uraikan batasan masalah yaitu:

1. Merancang sebuah aplikasi *digital signature* dengan *adler-32*.
2. Pengimplementasi aplikasi *digital signature* dengan *adler-32* ditampilkan secara rinci proses perhitungan *plaintext* tersebut.

1.4 Tujuan Penelitian

Adapun tujuan penelitian ini sebagai berikut:

1. Membuat perancangan aplikasi integritas data teks *digital signature* dengan *adler-32*.
2. Menampilkan proses perhitungan *plaintext* agar menghasilkan bilangan desimal yang di konversikan dengan bilangan heksadesimal.

1.5 Manfaat Penelitian

Berikut merupakan manfaat yang diharapkan dalam penelitian ini :

1. Dapat dijadikan sumber ketika ada yang ingin mengembangkan aplikasi ini.
2. Diharapkan dapat memahami bagaimana cara kerja *digital signature* dengan *adler-32.*

1.6 Metodologi Penelitian

Metode yang digunakan dalam penelitian ini merupakan metode deskriptif. Berikut teknik pengumpulan data yang dilakukan :

1. *Study Literature*

Data yang dikumpulkan berdasarkan jurnal, buku, *paper*, artikel dan bacaan bacaan yang ada kaitannya dengan judul penelitian.

2. Kajian Pustaka

   Data yang didapat dengan cara menggunakan atau mengumpulkan beberapa sumber yang tertulis yaitu dengan cara membaca mempelajari dan mencatat hal-hal penting yang berhubungan dengan masalah yang sedang dibahas guna mendapat gambaran teoritis.

BAB II

LANDASAN TEORI

2.1 *Digital Signature*

2.1.1 Pengertian *Digital signature*

Digital signature merupakan sistem keamanan kriptografi sismetris (*symmetric crypthography / secret key crypthography*) atau *public key cryptography system* yang dikenal sebagai kriptografi simetris, menggunakan kunci yang sama dalam melakukan enkripsi dan deskripsi terhadap suatu pesan *(message),* disini pengiriman dan penerima menggunakan kunci yang sama sehingga mereka harus menjaga kerahasian *(secret)* terhadap kunci tersebut. Pada *digital signature* suatu data/pesan simetris yang diciptakan secara acak *(randomly generated symmetric key)* yang kemudian akan dienkripsi dengan menggunakan kunci *public* dari penerima. *Digital signature* juga merupakan salah satu teknologi yang digunakan untuk meningkatkan keamanan jaringan.

Digital signature juga merupakan tanda tangan *digital* yang digunakan melalui media elektronik yang memiliki fungsi yang sama seperti tanda tangan manual yang biasanya orang gunakan, tanda tangan digital memiliki kumpulan *bit* yang bisa melakukan fungsi elektronik yang memakai fungsi *hash* satu arah.

Pada intinya tanda tangan *digital* (*digital signature*) dari tiap-tiap dokumen berbeda dengan dokumen yang lainnya karena diambil dari dokumen sendiri, dan pastinya jika berganti dokumen tanda tangan *digital* pun akan ikut berubah, pada era *digital* seperti saat ini *digital signature* atau dalam bahasa Indonesia di sebut tanda tangan *digital* memiliki fungsi yang sama dengan

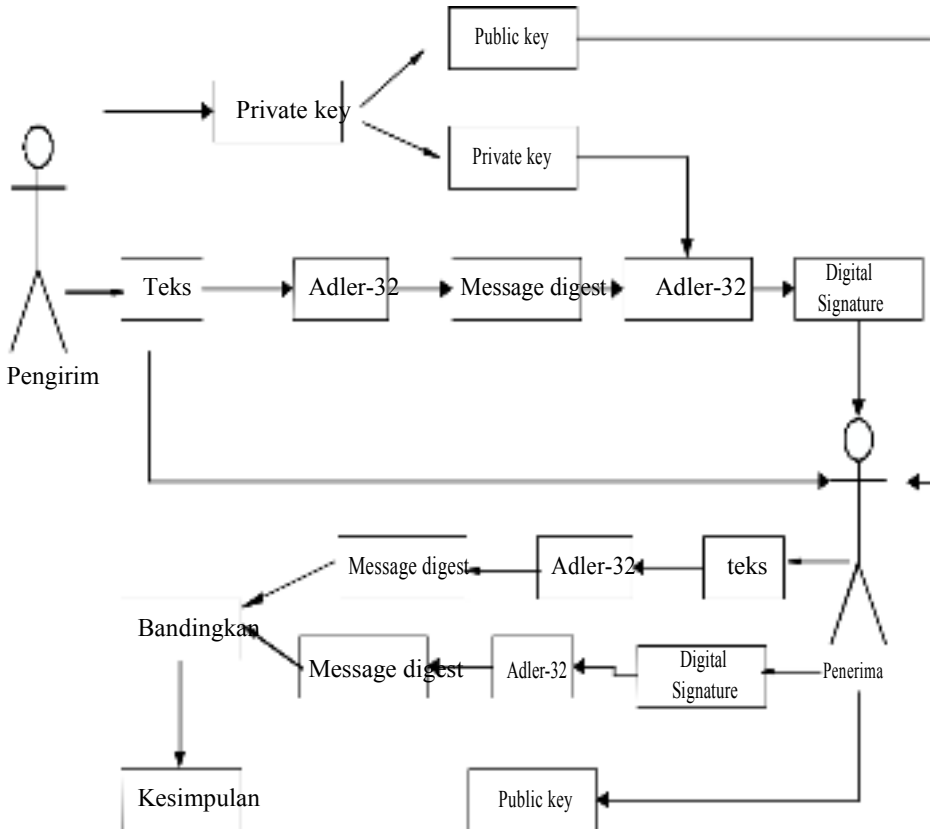
tanda manual yakni sebagai pengesahan suatu dokumen yang bisa dijadikan tanda persetujuan, tanda terima, dan lain-lain.

Penanda tanganan *digital* terhadap suatu dokumen dapat berupa sidik jari dari dokumen tersebut berserta *time stamp*-nya akan di enkripsi menggunakan kunci privat dari pihak yang menanda tangani dokumen tersebut. Tanda tangan *digital* memanfaatkan fungsi *hash* satu arah untuk menjamin bahwa tanda tangan *digital*/*digital signature* itu hanya berfungsi untuk dokumen atau pesan yang bersangkutan saja.

Fungsi *hash* satu arah adalah fungsi yang menerima *inputan* untuk dimodifikasi menjadi *output* yang mempunyai *size* yang tetap (biasanya ukuran akan lebih pendek dari *inputan*nya, *output* dari fungsi *hash* disebut / *message digest.*

Digital signature memiliki fungsi sebagai penanda pada data yang memastikan bahwa data tersebut adalah data yang sebenarnya (tidak ada yang berubah). Dengan begitu *digital signature* dapat memenuhi setidaknya dua syarat keamanan jaringan, yaitu *Authenticity* dan *nonrepudiation*.

Berdasarkan masalah yang telah diuraikan dalam pembangkitan tanda tangan *digital* setindaknya dibutuhkan sebuah algoritma fungsi *hash* dan sebuah algoritma kriptografi kunci publik yang akan dikombinasikan dan digunakan dalam membangkitkan tanda tangan *digital* yang aman dari suatu pesan. Model stadart dari digital signature dapat dilihat pada gambar berikut:



Gambar 2.1 Model Standart Digital Signature
(Sumber : Shaker dan Jumaa 2017)

Pada model diatas dapat dilihat bahwa penerima mendapatkan sebuah *public key* yang tidak dirahasia, sehingga siapa saja yang menyadap pesan yang diberikan oleh pengirim, maka penyadap mampu memeriksa keaslihan data. Jika pengirim pesan hanya menginginkan agar penerima pesan yang hanya dapat mengecek keaslian data maka model standart dari *digital signature* ini tidak lagi menutupi keinginan pengirim data. Agar hanya penerima saja yang dapat menguji keaslian data, maka dapat dibutuhkan pengembangan model *digital signature* yang baru dimana penyadap tidak mampu menguji keaslian data yang disadap. Sehingga penyadap tidaka akan pernah mengetahui apakah data yang dibajak merupakan data asli atau data modifikasi.

Menurut Mohamad Ihwani dalam *Internal Journal of Computer Science And Software Engineering* (IJCSSE) yang berjudul Model Keamanan Informasi Berbasis *Digital Signature* Dengan Algoritma RSA (2016). *Digital signature* merupakan suatu tanda tangan elektronik yang digunakan untuk membuktikan keaslian identitas pengiriman dari suatu pesan atau penandatangan dari suatu dokumen *digital*. Tanda tangan *digital* memastikan isi pesan atau dokumen *digital* yang dikirim tidak mengalami perubahan sampai ke tangan penerima. Dengan demekian penerima yakin bahwa pesan yang diterimanya benar-benar asli dari pihak pengirim.

Digital signature juga membutuhkan fungsi *hash*, fungsi *hash* merupakan fungsi yang menerima masukan *string* yang panjangnya sembarang dan mengkonversinya menjadi *string* keluaran yang panjangnya tetap (*fixed*), umumnya berukuran jauh lebih kecil dari pada ukuran *string* semula, hasil dekripsi dari proses *digital signature* untuk otentikasi dan integritas dari keaslian pesan.

Digital signature banyak dipublikasikan untuk keamanan informasi berupa *file digital*, hal tersebut dilakukan untuk mencegah pemalsuan pengiriman suatu *file* atau pesan digital. Teknologi informasi yang semangkin berkembang saat ini menuntut perusahaan termasuk harus melakukan otentikasi pengirim dan kepercayaan pada *file* atau pesan *digital* yang akan dikirimkan.

Pesan tersebut sudah tidak asli lagi. Apalagi pesan M tidak berasal dari orang yang sebenarnya, maka *message digest* (MD) yang dihasilkan berbeda

dengan *message digest* (MD') yang dihasilkan pada proses verifikasi karena kunci *public* yang digunakan oleh penerima pesan tidak berkoresponden dengan kunci privat pengirim. Bila MD = MD' maka pesan yang diterima adalah pesan asli (*message authentication*) dan orang yang mengirim merupakan orang yang sebenarnya *user authentication*.

proses pemberian tanda tangan digital (*signing*) dimulai dari menghitung *message digest* (MD) dari pesan yang diperoleh dengan mentransformasikan pesan M dengan menggunakan fungsi *hash* satu arah H.

$$MD = H(M) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Messages Digest (MD) dienkripsi dengan algoritma kunci *public* dan menggunakan kunci *private* (SK), hasil enkripsi ini dinamakan dengan tanda-tangan *digital* S.

$$S = Esk(MD) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

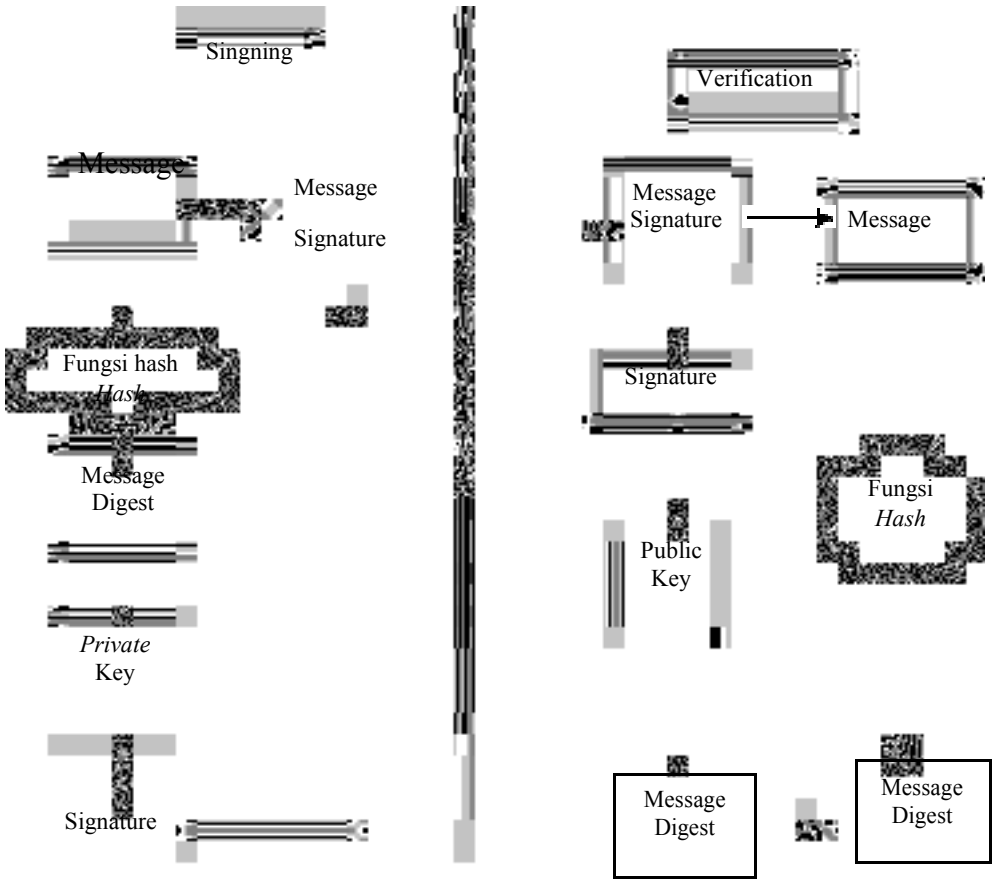
Tanda tangan digital S diletakan ke pesan M dengan cara *append* S, dan dikirim melalui media komunikasi, dalam hal ini dapat dikatakan bahwa pesan M sudah ditanda tangani oleh pengirim dengan tanda tangan *digital* S.

Untuk membuktikan keontetikannya maka sipenerima melakukan verifikasi, dimana tanda tangan digital S di dekripsi dengan menggunakan kunci *public* (PK) sehingga menghasilkan *messages digest* (MD).

$$MD = Dpk(S) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

Penerima kemudian merubah M menjasi *messages digest* MD' menggunakan fungsi *hash* satu arah yang sama dengan fungsi *hash* yang

digunakan pengirim, jika MD' = MD berarti tanda tangan yang diterima otentik dan berasal dari pengirim yang benar.



Gambar 2.2. Proses *Authentication Digital signature*
(Sumber: Mohamad Ihwani, 2016)

Cara kerja *Digital signature* adalah dengan memanfaatkan dua buah kunci, yaitu kunci *public* dan kunci *private*. Kunci *public* digunakan untuk mengenkripsi data, sedangkan kunci *private* digunakan untuk mendekripsi data. Pertama, dokumen di-*hash* menghasilkan *Message Digest*. Kemudian, *Message Digest* dienkripsi oleh kunci *public* menjadi *Digital signature.*

Untuk membuka *Digital signature* tersebut diperlukan kunci *private*. Bila data telah diubah oleh pihak luar, maka *Digital signature* juga ikut berubah

sehingga kunci *private* yang ada tidak akan bisa membukanya. Ini merupakan salah satu syarat keaman jaringan, yaitu *Authenticity*. Artinya adalah, keaslian data dapat terjamin dari perubahan-perubahan yang dilakukan pihak luar.

Dengan cara yang sama, pengirim data tidak dapat menyangkal data yang telah dikirimkannya, bila *Digital signature* cocok dengan kunci *private* yang dipegang oleh penerima data, maka dapat dipastikan bahwa pengirim adalah pemegang kunci *private* yang sama. Ini berarti *Digital signature* memenuhi salah satu syarat keamanan jaringan, yaitu *Nonrepudation* atau non-penyangkalan. Dengan demikian dapat disimpulkan bahwa *Digital signature* sangat bermanfaat sebagai salah satu teknologi pengamanan jaringan. Fungsinya untuk memastikan keaslian data sangat berguna di saat pembajakan dan pemalsuan data semangkin marak.

2.1.2 Pembangkitan Kunci *Digital Signature*

Dalam membangkitkan kunci *Digital Signature* memiliki beberapa proses. Adapun proses ini digunakaan untuk membangkitkan *public key* dan *private key*, berikut prosesnya adalah :

1. p adalah bilangan prima dengan panjang L, bit, dimana $2^{L1} < p < 2^{L}$ dengan $512 \leq L \leq 1024$ dan L adalah kelipatan 64. q, bilangan prima 160 bit, factor dari $p-^{1}$ dimana $2^{159} < q < 2^{160}$. Parameter p bersifat *public*.

a. $g = h^{(p-1)q} \bmod p$, dimana $1 < h < p-^{1}$ sehingga $g > 1$. Parameter g bersifat publik.

b. x bilangan bulat yang dibangkitkan random atau pseudorandom dimana $0 < x < q$ dengan panjang 160 bit. Parameter x bersifat *private*.

c. y = gx mod p adalah kunci *public*.

d. M adalah pesan yang akan diberikan tanda tangan.

e. k = bilangan bulat yang dibangkitkan random atau pseudorandom dimana $0 < k < q$.

Parameter p, q dan g bersifat *public* dan dapat digunakan Bersama dalam sekelompok orang, parameter p, q dan g juga bernilai tetap untuk priode/waktu tertentu. Parameter x dan k hanya digunakan untuk pembangkitan tanda tangan dan harus dijaga kerahasiaannya, parameter k harus berbeda untuk setiap tanda tangan.

1. Pembangkitan Sepasang Kunci

   a. pilih bilangan prima p dan q, dimana $(p-^{1})$ mod q = 0

   b. 2 hitung $g = h^{(p-1)/q}$ mod p, dimana $1< h < p-^{1}$ dan g > 1

   c. Tentukan kunci *private* x < q

   d. Hitung kunci *public* $y = g^{x}$ mod

2. Pembentukan Sidik *digital* (*signing*)

   a. Ubah pesan m menjadi *message digest* dengan fungsi *hash* H.

   b. Tentukan bilangan acak K< q.

   c. Sidik *digital* dari pesan m adalah bilangan r dan s hitung r dan s sebagai berikut:

$$r = (g^{k} \bmod p) \bmod q$$

$$s = (k^{-1} (H(m) + x * r)) \bmod q$$

   d. Kirim pesan m dan sidik *digital* r dan s.

3. Verifikasi Keabsahan Sidik *Digital* (*verifying*)

   a. Hitung

$$W = s^{-1} \bmod q$$

$$U_1 = (H(m) * w) \bmod q$$

$$U_2 = (r * w) \bmod q$$

$$V = ((g^{u1} * y^{u2}) \bmod p) \bmod q)$$

   b. Jika v = r, maka sidik digital sah, yang berarti bahwa pesan masih asli dan dikirim oleh sipengirim yang benar.

Maka di dapatkan kunci *public* (p,q,g,y) dan kunci *private* (p,q,g,x).

1. Perhitungan *Digital Signature*:

   a. Pilih bilangan prima p dan q dengan $(p-^{1}) \bmod q = 0$ yaitu = 59419 dan q = 3301 (memenuhi 3301,18 = 59419_1).

   b. Hitung g = h (p-1)/q mod p, dimana 1 < h < p-1 dan g > 1, yaitu (ambil h = 100) g =100 (54919-1)/3301 mod 59419 = 18870.

   c. Tentukan kunci rahsia x bilangan bulat <q, ambil x =3223.

      Hitung kunci *public* y = gx mod p =18870 3223 mod 59419 = 29245.

2. Pembentukan Sidik *Digital* (*Signing*)

   a. Hitung nilai *hash* dari pesan misalkan H(m) = 4321

   b. Tentukan bilangan acak k < q

$$K = 997$$

$$k^{-1} = 2907 \pmod{3301}$$

c. Hitung r dan s sebagai berikut:

$$r = (g^{k} \bmod p) \bmod q = 848$$

$$s = (k^{-1} (H(m) + x * r)) \bmod q$$

$$= 7957694475 \bmod 3301 = 183$$

3. Kirim pesan m dan sidik *digital* r dan s.

a. Verifikasi Keabsahan Sidik *Digital*

b. Hitung

$$s^{-1} = 469 \pmod{3301}$$

$$w = s^{-1} \bmod q = 469$$

$$u_1 = (H(m) * w) \bmod q \; 2026549 \bmod 3301 = 3036$$

$$u_2 = (r * w) \bmod q = 397712 \bmod 3301 = 1592$$

$$v = ((g^{u1} * y^{u2}) \bmod p) \bmod q) = 848 \bmod 3301 =$$

848 Karena v = r, maka sidik *digital* sah.

2.1.3 Manfaat *Digital signature*

Tanda tangan *digital* memanfaatkan teknologi kunci *public (public key)*. Kunci *public private* dibuat unik dan tidak ada pasangannya. Kunci *private* disimpan oleh pemiliknya, dan dipergunakan untuk membuat tanda tangan *digital*. Sedangkan kunci *public* dapat diserahkan kepada siapa saja yang ingin memeriksa tanda tangan *digital* yang bersangkutan pada suatu dokumen. Proses pembuatan dan pemeriksaan tanda tangan ini melibatkan sejumlah teknik kriptografi seperti *one-way hashing* dan enkripsi asimetris. Adapun manfaat dari *Digital Signature* yaitu sebagai berikut :

1. Efisien, Fleksibel dan Praktis

   Tidak perlu pena, tintas dan kertas. Tanda tangan *digital* takkan pernah bisa luntur seperti halnya tanda tangan biasa diatas kertas yang seiring waktu bisa luntur dan pudar. Tanda tangan *digital* ini juga bisa dikirimkan dari jarak jauh, inilah yang membuat tanda tangan digital ini lebih bersifat Efisien dan praktis.

2. Handal dan Akurat

   Jika menggunakan tanda tangan biasa maka terkadang antara satu tanda tangan dengan lainnya sulit untuk sama persis. Pasti selalu ada yang berbeda meskipun sangat kecil kelihatannya. Dengan aplikasi *digital signature* ini maka akan selalu dihasilkan tanda tangan yang identik.

3. Tidak bisa dipalsukan

   Sudah sering terdengar jika tanda tangan biasa bisa dipalsukan, terlihat sebuah kelemahan dari tanda tangan biasa yaitu sangat mudah untuk dipalsukan. Jadi perlu sebuah urgensi penggunaan *digital signature* apalagi dilingkungan pemerintahan, dengan adanya *digital signature* ini tanda tangan tidak akan mudah dipalsukan lagi karena angka yang keluar dari *digital signature* tidak akan pernah bisa digunakan lagi.

2.1.4 Fungsi Digital Signature

Tanda tangan *digital* dapat digunakan dengan segala jenis pesan, apakah itu dienkripsi atau tindak. Fungsi lain dari tanda tangan *digital* ini adalah penerima dapat yakin dengan identitas pengirim dan pesan tersebut diterima secara utuh. Tanda tangan *digital* mempersulit penandatangan untuk

menyangkal telah menandatangani sesuatu, dengan asumsi kunci privat mereka belum dikompromikan, karena tanda tangan *digital* unik untuk dokumen dan penandatangan dan mengikat mereka Bersama-sama. Properti ini disebut nonrepudiation.

Sebagian besar program *email* modern saat ini mendukung penggunaan tanda tangan *digital* dengan sartifikat *digital.* Tanda tangan *digital* juga digunakan secara luas untuk memberikan bentuk keaslian intrigritas data, penolakan komunikasi dan transaksi yang lakukan melalui internet.

2.1.5 Tujuan *Digital signature*

Salah satu cara yang digunakan untuk memastikan surat tersebut adalah dengan mengecek tanda tangan yang ada didalam surat tersebut dan stempel yang menunjukan keaslian pengirim surat. Tanda tangan *digital* atau lebih dikenal dengan *digital signature* mempunyai fungsi yang sama dengan tanda tangan analog yang ditulis diatas kertas.

Tujuan dari pengguna *digital signature* ini adalah untuk membuktikan keaslian indentitas pengiriman dari suatu pesan atau penandatangan dari suatu dokumen, dan untuk memastikan isi yang asli dari pesan atau dokumen itu yang sudah dikirim tanpa perubahan. Dengan mudah dapat dipindahkan, tidak bisa ditiru oleh orang lain, dan dapat secara otomatis dilakukan *time-stamp*.

Tanda tangan *digital* harus unik sehingga dapat membedahkan pengirim yang satu dengan yang lainnya. Tanda tangan *digital* harus sulit untuk ditiru dan dipalsukan sehingga intergritas dan keabsahan pesan dapat terjaga. Dengan

demikian diharapkan pencatutan identitas ketika pesan atau *email* tersebut dikirim dapat dihindari.

2.2 Proses Enkripsi

Enkripsi adalah proses mengamankan suatu proses dengan membuat informmasi tersebut tidak dapat dibaca tanpa bantuan pengetahuan khusus. Adapun langkah-langkah proses enkripsi dalam *digital signature* adalah sebagai berikut:

1. Contoh si A mengirimkam kunci *public key* (n,e) nya untuk si B dan menyimpan secara rahasia *private key*-nya.
2. Si B ingin mengubah teks M pada si A.
3. Si B kemudian merubah M menjadi kode *ascii* (berupa integer).
4. Menghitung *chipertext* c (nilai yang telah terenkripsi) dengan menggunakan *public key* yang dikirimkan oleh si A kepadanya.
5. Kemudian B mengirimkan nilai c kepada A untuk di-dekripsi dengan menggunakan *private key* miliknya.
6. Nilai M harus lebih besar dari 0, dan harus lebih kecil dari nilai n (dari *public key*).

2.3 Proses Dekripsi

dekripsi tidak jauh berbeda dengan proses enkripsi yang berbeda hanya nilai yang dimasukan . dalam proses dekripsi , nilai M diganti dengan nilai C *chiphertext* (hasil enkripsi) dan nilai e dari *public key* diganti dengan nilai d

dari *private key*, sedangkan nilai n dari *public key* selalu sama dengan nilai n dari *private key*.

Adapun langkah-langkah proses dekripsi dalam *digital signature* adalah sebagai berikut:

1. Contoh si A mengirimkan *private key* (n,d) nya untuk si B, dan menyimpan secara rahasia *public key*-nya.
2. Si B ingin mengubah teks c pada si A.
3. Si B kemudian merubah c menjadi kode *ascii* (berupa integer).
4. Menghitung M (nilai yang telah di dekripsi) dengan menggunakan *public key* yang dikirimkan oleh si A kepadanya.
5. Kemudian B mengirimkan nilai M kepada si A dengan menggunakan *private key* miliknya.

2.4 Integritas

Dalam skenario banyak, pengiriman dan penerima pesan mungkin memiliki kebutuhan untuk yakin bahwa pesan belum diubah selama transmisi. Meskipun menyembunyikan enkripsi isi pesan, dimungkinkan untuk mengubah sebuah pesan terenkripsi tanpa memahaminya. (Algoritma enkripsi beberapa, yang dikenal sebagai *nonmalleable* yang, mencegah hal ini, tetapi yang lain tidak). Namun, jika pesan secara *digital* ditanda tangani, setiap perubahan dalam pesan setelah tanda tangan akan membatalkan tanda tangannya. Selain itu, tidak ada cara yang efisien untuk memodifikasi pesan dan tanda tangan untuk menghasilkan pesan baru dengan tanda tangan yang sah, karena ini

masih dianggap layak oleh sebagian besar komputasi fungsi *hash* kriptografi (lihat risistensi tabrakan).

Integritas adalah aspek yang memastikan bahwa data tidak boleh diubah tanpa izin dari yang berwenang kompeten. Untuk aplikasi pencitraan *digital* aspek integritas adalah yang terpenting berisi rahasia informasi akses data sering dicari oleh penyusup, data yang sudah dikirimkan tidak dapat diubah oleh pihak yang tidak berwenang pelanggaran ini akan mengakibatkan tidak berfungsinya validasi. Terutama dibidang Pendidikan, obat-obat, militer,dan lain-lain perlu membuktikan keasliannya dari konten, beberapa dari mereka digunakan sebagai bukti fakta.

Validasi integritas tidak hanya menyimpan hidup seseorang , tetapi bisa diimplementasikan pada keamanan sisi. Pengguna sistem pencitraan yang sangat mengarah kedata pertukaran melalui udara sambil melampirkan ke internasional jaringan. Sementara komunikasi sangat penting untuk memverifikasi pesan sehingga penyusup tidak dapat menggantikan dengan yang palsu.

Ada beberapa cara untuk menjaga intergritas data, yang pertama bisa dengan dilindungi di inputan *software,* bisa juga dijaga level sql/ *database.* Pencegahan gangguan integritas data di level *software* bisa dilakukan dengan cara memastikan inputan di form valid. Sedangkan integritas data *via SQL* bisa dilakukan dengan cara menambahkan *constrain check*.

2.5 Algoritma

Menurut Gun Gun Maulana dalam Jurnal yang berjudul Pembelajaran Dasar Algoritma Dan Pemrograman Menggunakan *EL-Goritma* Berbasis Web (2017) algoritma adalah metode efektif yang diapresikan sebagai rangkaian terbatas. Algoritma juga merupakan kumpulan perintah-perintah ini dapat diterjemahkan secara bertahap dari awal hingga akhir. Masalah tersebut dapat berupa apa saja, dengan syarat untuk setiap permasalahan memiliki kriteria kondisi awal yang harus dipenuhi sebelum menjalankan sebuah algoritma. Algoritma juga memiliki pengulangan proses (iterasi), dan juga memiliki keputusan hingga keputusan selesai.

Dalam cabang disiplin ini, algoritma dipelajari secara abstrak, terlepas dari *system* komputer atau pemrograman yang dipergunakan. Algoritma yang berbeda dapat diterapkan untuk suatu permasalahan dengan kriteria yang sama. Kompleksitas dari suatu algoritma merupakan ukuran seberapa banyak komputasi yang diterapkan pada algoritma tersebut untuk menyelesaikan permasalahannya. Secara informal, algoritma yang dapat menyelesaikan permasalahan dalam waktu yang relative singkat memiliki tingkat kompleksitas yang rendah, sementara untuk algoritma yang menyelesaikan permasalahan dalam waktu yang lebih lama memiliki tingkat kompleksitas yang lebih tinggi pula.

Dalam mata kuliah logika dan algoritma, kita telah mempelajari tentang algoritma dan penerapannya dalam pemrograman *computer*. Kesulitan yang dihadapi dalam permasalahan yang dihadapi, serta sulitnya membayangkan

struktur data yang akan digunakan. Dalam memahami penyelesaian suatu permasalahan, kita akan lebih mudah untuk mengingat dan memahaminya apabila permasalahan itu dapat ditampilkan dalam bentuk visual dan gambar, sehingga penyajiannya menjadi menarik.

2.6 Kriptografi

Secara etimologi, kriptografi (*cryptograph*) berasal dari bahasa Yunani dan terdiri dari dua suku kata, yaitu "*cyptos*" yang artinya rahasia (*secret*) dan "*graphein*" yang artinya menulis (*writing*). Sehingga kriptografi bisa diartikan sebagai tulisan rahasia (*secret writing*). Kriptografi kadang diartikan sebagai ilmu dan seni untuk menjaga sebuah pesan.

Pengertian yang lain, kriptografi merupakan ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, otentikasi entitas dan otentikasi keaslian data. Kriptografi tidak hanya menyediakan keamanan informasi, melainkan sebuah himpunan teknik-teknik, penggunaan kata "seni" didalam definisi di atas berasal dari fakta sejarah bahwa pada masa-masa awal sejarah kriptografi, setiap orang mungkin mempunyai cara yang unik untuk merahasiakan pesan.

Cara-cara unik tersebut mungkin berbeda-beda pada setiap pelaku kriptografi sehingga setiap cara menulis pesan rahasia mempunyai nilai estetika tersendiri sehingga kriptografi berkembang menjadi sebuah seni merahasiakan pesan. Suatu proses penyandian yang melakukan perubahan sebuah kode (pesan) dari yang bisa dimengerti atau *plainteks* menjadi sebuah

kode yang tidak bisa dimengerti atau *ciphertext* disebut enkripsi. Sedangkan proses kebalikannya untuk mengubah *ciphertext* menjadi *plainteks* disebut *dekripsi*.

Algoritma kriptografi disebut juga sebagai *cipher* yaitu aturan untuk *enkripsi* dan *deskripsi,* atau fungsi matematika yang digunakan untuk *enkripsi* dan *deskripsi.* Beberapa *cipher* memerlukan algoritma yang berbeda untuk *enkripsi* dan *deskripsi.* Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yaitu himpunan yang berisi elemen-elemen *plainteks* dan himpunan berisi *cipherteks.* Proses enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara kedua himpunan tersebut. Misalkan P menyatakan *plainteks* dan C menyatakan *cipherteks,* maka fungsi enkripsi memetakan P ke C.

E(P) = C

Dan fungsi dekripsi memetakan D ke C,

D(C) = P

Karena prose enkripsi kemudian dekripsi mengembalikan pesan ke pesan asal, maka belaku persamaan D (E(P)) = P.

2.7 Konsep Dasar Kriptografi

Rinaldi Munir dalam bukunya yang berjudul pengantar kriptografi (2006), menjelaskan bahwa kriptografi (*cryptography*) berasal dari Bahasa Yunani yaitu kriptografi (cryptograph) berasal dari bahasa Yunani dan terdiri dari dua suku kata, yaitu "*cyptos*" yang artinya rahasia (*secret*) dan "

graphein" yang artinya menulis (*writing*). Sehingga kriptografi bisa diartikan sebagai tulisan rahasia (*secret writing*).

Sedangkan Rakhmadi, A., & Nugroho dalam bukunya yang berjudul *Web-Based Application fo Single Elimination Tournament Using Linear Congruential Generator* (2016), menjelaskan bahwa kriptografi merupakan sebuah algoritma penyandian untuk menjaga kerahasia suatu data. Kriptografi dapat dilakukan dengan cara subtitusi (pergantian huruf) dan transposisi (perpindahan posisi) yang menghasilkan kode-kode rahasia dengan makna yang tidak mudah dipahami orang lain. Istilah-istilah yang sering muncul dengan kriptografi yang penting untuk dikethui yaitu:

1. *Plainteks*, pesan asli yang dapat dibaca dan mudah dipahami, *plainteks* dapat berupa gambar, pesan tulisan, maupun video bergambar.
2. *Ciphertext*, pesan yang sudah disandikan dari pesan asli menjadi kode-kode yang tidak dapat dipahami karena membentuk kata-kata yang tidak lazim
3. Enkripsi, proses mengubah *plainteks* menjadi *ciphertext*.
4. Dekripsi, merupakan kebalikan dari enkripsi yaitu proses mengubah *ciphertext* menjadi *plainteks.*

2.8 Fungsi *Hash*

Di dalam kriptografi terdapat sebuah fungsi yang sesuai untuk aplikasi keamanan seperti otentikasi dan integrasi pesan. Fungsi tersebut adalah fungsi *hash* yaitu fungsi yang menerima *string* dengan panjang sembarang dan mengkonversinya mejadi *string* keluaran yang panjangnya tetap (*fixed*).

Menurut Ainul Wardah Dalam Jurnal Teknik Informatika Yang Berjudul Perancangan Aplikasi Pembelajaran Kriptografi Pada Algoritma Fungsi *Hash* Menggunakan Metode *Computer Based Learning* (2017). Fungsi *hash* adalah seluruh pesan dan keluaran sebuah sidik pesan dan keluaran sebuah sidik pesan (*Message Fingerprint*). Sidik pesan yang sering di sebut *message digest.* Fungsi *hash* dapat digunakan untuk mewujudkan beberapa layanan keamanan jaringan misalnya, untuk keutuhan data dan otentikasi pesan.

Pengirim pesan dan penerima pesan dan penerima pesan memiliki cara sehingga keutuhan data dapat diselidiki. Fungsi *hash* dapat digunakan untuk mewujudkan layanan keutuhan data. Misalnya M merupakan h adalah fungsi *hash,* maka y = h (M) disebut dengan sidiki pesan x atau sering disebut dengan *message digest.* Sebuah *message digest* umunya berukuran pendek, yaitu sekitar 160 bit[5].

Dengan menggunakan sebuah fungsi *hash* h. Sebelum pesan M disebutkan/dikirimkan sebuah *massage digest* y lama = h (M) disimpan sebagai acuan. Misalnya didapatkan kemabali M setelah disebarkan apabila ingin menguji apakah M = M hitung kembali *massage digest* baru y baru = h (M) disimpulkan pesan tidak berubah bila y lama = y baru. Contoh: 25 mod 11 = 3

Jika key bernilai negatif, maka bagi key dengan N untuk mendapatkan sisa r:

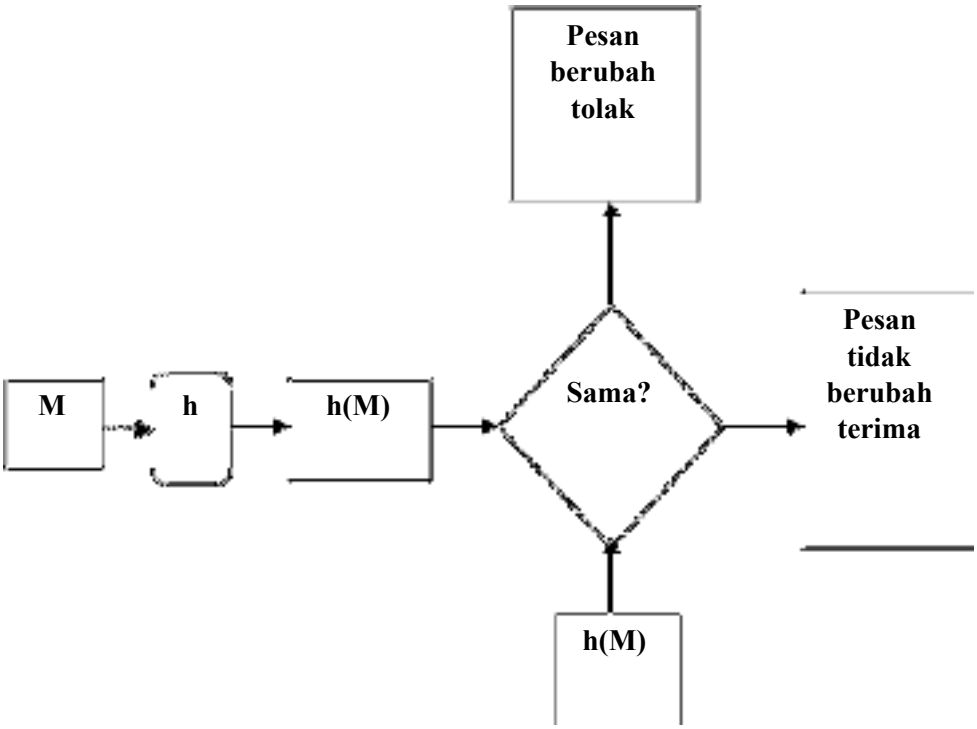
Untuk r = 0, maka key mod N = 0

Untuk r < > 0, maka key mod N = N_{-r}

Menurut Budi Maryanto Dalam Jurnal media informatika Berjudul Penggunaan Fungsi *Hash* Satu Arah Untuk Enkripsi Data (2008). Fungsi *hash* satu arah adalah fungsi yang bekerja satu arah yaitu pesan yang sudah diubah menjadi ringkasan pesan maka tidak dapat dikembalikan lagi menjadi pesan semula.

Berdasarkan pembahasan diatas Fungsi *hash* H satu arah mempunyai sifat sebagai berikut:

1. Jika diberikan M (M adalah suatu data dalam hal ini berupa pesan), maka H(M) = h mudah dihitung.
2. Untuk setiap h yang dihasilkan, tidak mungkin dikembalikan nilai M sedemikian sehingga H(M) = h itulah sebabnya fungsi H dikatakan fungsi *hash* satu arah.
3. Jika diberikan M, tidak mungkin mendapatkan M* sedemikian sehingga H(M) = H(M*). Bila diperoleh pesan M* semacam ini maka disebut tabrakan (*collision*). Maka sangat sulit menemukan dua pesan yang berbeda yang menghasilkan nilai *hash* yang sama.
4. Tidak mungkin mendapatkan dua pesan M dan M* sedemikian sehingga H(M) = H(M*).



Gambar 2.3. Pengujian keutuhan pesan dengan fungsi *hash*
(Sumber: Ainul Wardah, 2017)

Sebuah fungsi *hash* satu arah H(M) beroperasi pada prapeta pesan M dengan panjang sebarang dan mengembalikan nilai *hash* h yang memiliki panjang tetap. Fungsi *hash* dikembangkan berdasarkan ide sebuah fungsi kompresi. Fungsi satu arah ini menghasilkan nilai *hash* kata dari *hash* diberi label $H_0^{(i)}$, $H_1^{(i)}$, ,……, $H_4^{(i)}$ yang akan menampung nilai *hash* awal $H^{(0)}$ untuk

diganti dengan nilai *hash* yang berurutan setelah blok pesan diperoses $H^{(N)}$.

Ada pun beberapa fungsi *hash* satu arah yang pernah dibuat, antara lain:

1. MD2(*Message digest2*)
2. MD4 (*Message digest4*)
3. MD5 (*Message digest5*)
4. *Secure Hash Function* (SHA)

5. *Adler32*
6. Snefru; N-has; RIPE-MD, dan lain sebagainya.

Ada pun manfaat dari fungsi *hash* satu arah, antara lain:

1. Menjaga Intergritas Data

   Fungsi *hash* sangat peka terhadap perubahan 1 bit pada pesan. Pesan berubah 1 bit maka nilai *hash* berubah sangat signifikan.

2. Menghemat Waktu Pengiriman

   Misalnya untuk memverifikasi sebuah salinan arsip dengan arsip asli. Di sini salinan dokumen berada di tempat yang jauh dari basis data arsip asli. Dari pada mengirim salinan arsip tersebut secara keseluruhan ke komputer pusat (yang membutuhkan waktu transmisi lama), lebih baik mengirimkan *message digest*-nya. Jika *message digest* salinan arsip sama dengan *message digest* arsip asli, berarti salinan arsip tersebut sama dengan arsip master.

3. Menormalkan Panjang Data Yang Beraneka Ragam

   Misalnya *password* yang panjangnya bebas (minimal & karakter). Kemudian password disimpan di komputer host (*server*) untuk keperluan otentikasi pemakai komputer. Untuk menyeragamkan panjang *field password* di dalam basis data, *password* disimpan dalam bentuk nilai *hash* (panjang nilai *hash* tetap).

Berdasarkan pengertian diatas maka dapat disimpulkan bahwa fungsi *hash* juga merupakan fungsi yang secara efisien memiliki sifat keamanan tambahan

sehingga dapat dipakai untuk mengamankan data teks beserta dokumen-dokumen lainnya.

2.9 Kriteria Keamanan Fungsi *Hash*

Fungsi *hash* h yang dipakai pada sistem kriptografi harus memnuhi beberapa syarat sehingga dapat dianggap aman, yaitu ketahanan terhadap serangan *preimage*, ketahanan terhadap seranganan *second preimage* dan ketahanan terhadap serangan *collision*. Makna ketahanan disini adalah jika diberikan persoalan *preimage*, *second preimage* dan *collision* pada fungsi *hash* h, maka persoalan itu semuannya susah untuk diselesaikan.

Preimage fungsi *hash* h diharuskan bersifat satu arah, yaitu jika diberikan pesan M, maka dapat dihitung dengan mudah y = h (M). Namun jika diberikan y dan fungsi h, maka sulit bagi penyerang untuk menemukan M.

Second preimage permasalahan *second preimage* adalah jika sebuah pesan M dan fungsi *hash* h, maka ditemukan M sehingga h(M)= h (M). sebuah fungsi *hash* untuk sistem kriptografi harus tahan terhadap serangan *second preimage* atau dalam kata lain sulit bagi penyerang untuk menyelesaikan persoalan *second preimage.*

Collicion persoalan *collicion* hamper mirip dengan persoalan *second preimage* hanya saja persoalan *collicion* penyerang hanya diberikan fungsi *hash* lalu penyerang berusaha menemukan sepasang pesan M dan M" sehingga h(M) = h(M"). sebuah fungsi *hash* untuk sistem kriptografi harus tahan terhadap penyerangan *collicion* atau dalam kata lain *collicion* merupakan persoalan yang sangat sulit.

2.1.6 Keamanan Data

Secara umum data dibagi menjadi dua, yaitu: data yang bersifat rahasia dan tidak bersifat rahasia, dalam hal ini, pesan yang diperhatikan dan perlu diamankan adalah pesan yang bersifat rahasia.

Menurut Agustius Widyartono (2011), beberapa ancaman dan serangan yang terjadi saat data tidak lagi dipertukarkan dengan menggunakan media penyimpanan yang bersifat *mobile*, dan saat data melalui jalur telekomunikasi. Disini banyak yang akan terjadi dalam keamanan data sehingga menimbulkan beberapa ancaman yaitu:

1. *Interruption*

   Mengancam ketersediaan data dan informasi yang ada di dalam sistem komputer dan komunikasi secara fisik, sehingga saat data dan informasi dibutuhkan mengalami kesulitan dalam mengaksesnya.

2. *Interception*

   Mengancam kerahsiaan sebuah data, merupakan penyadapan informasi oleh pihak-pihak yang tidak berhak atas sebuah informasi.

3. *Modification*

   Mengancam validitas isi sebuah data, selain berhasil melakukan penyadapan juga dilakukan perubahan atas data sehingga informasi yang dihasilkan menjadi bias.

4. *Fabrication*

   Mengancam integritas sumber pengiriman data, pihak yang tidak berhak berhasil melakukan peniruan sehingga dianggap sebagai pihak yang benar-benar dikehendaki.

   Menurut Stalling, ada beberapa hal yang terpenting dalam issue keamanan data yaitu:

1. *Confidentialy*

   Menjamin bahawa data-data tersebut hanya bisa diakses oleh pihak tertentu saja.

2. *Autehntification*

   Pada saat mengirim atau menerima informasi, kedua belah pihak perlu menegtahui bahwa pengirim dari pesan tersebut adalah orang yang sebenarnya seperti yang diklaim.

3. *Integrity*

   Tuntunan ini berhubungan dengan jaminan setiap pesan yang dikirim pasti sampai pada penerimanya tanpa ada bagian dari pesan tersebut yang diganti diduplikasikan, dirusak, diubah urutannya, dan ditambahkan.

4. *Non-Repudiation*

   Mencegah pengiriman maupun penerima mengingkari bahwa mereka telah mengirimkan atau menerima suatu pesan atau informasi, jika sebuah pesan dikirim, penerima dapat membuktikan bahwa pesan

diterima, pengirim dapat membuktikan bahwa pesannya telah diterrima oleh pihak yang ditujuhnya.

5. *Acces Control*

   Membatasi sumber-sumber data hanya kepada orang-orang tertentu.

6. *Availability*

   Diperlukan setiap saat semua informasi pada sistem komputer harus tersedia bagi semua pihak yang berhak atau informasi tersebut.

2.1.7 UML (*Unifled Modelling Language*)

1. Pengertian UML

UML digunakan untuk menggambarkan perancangan awal dari sistem yang akan dibangun UML memiliki banyak jenis pemodelan. Tetapi hanya beberapa yang digunakan saja yang akan dibahas. UML merupakan suatu bahasa, suatu bahasa terdiri dari kata-kata, dan memiliki aturan untuk menggabungkan kata-kata tersebut, sehingga tercipta komunikasi. Sebuah pemodelan bahasa adalah suatu bahasa dimana kata-kata dan aturannya berfokus pada penggambaran sistem secara konseptual dan fisik. Sebuah pemodelan bahasa seperti UML telah menjadi bahasa standar untuk merencanakan suatu aplikasi.

Hasil dari pemodelan tadi adalah penegertian dari suatu sistem. Satu model saja tidak cukup untuk menggambarkan sistem secara keseluruhan, maka dibutuhkan banyak model yang berhubung satu dengan yang lainnya untuk memberikan pengertian pada dasar dari sistem.

Menurut Windu Gata, Grace (2013), *Unifled Modelling Language* (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak UML merupakan metodolgi dalam penegmbangan sisten beriontasi objek dan juga merupakan alat untuk mendukung pengembangan sistem.

Keuntungan UML adalah:

1. Sebagai bahasa pemodelan yang general-purpose, difokuskan pada pokok himpunan konsep yang dapat dipakai bersama dan menggunakan pengetahuan bersama dengan mekanisme perluasan.
2. Sebagai bahasa pemodelan yang mudah diaplikasikan dapat diapalikasikan untuk bermacam tipe sistem (software dan non-software), domain dan metode atau proses.
3. Sebagai bahasa pemodelan standart industry, bukan merupakan bahasa yang tertutup atau satu-satunya, tapi bersifat terbuka dan sepenuhnya dapat di perluas.

2. Jenis-Jenis Diagram UML

Berikut adalah jenis-jenis diagram UML yang sering digunakan untuk mengambarkan objek sistem dan pengguna sebelum merancangan suatu sistem dan gambaran alur kerja sistem:

a. *Use Case* Diagram

Use case diagram secara grafis mengambarkan interaksi antara sistem, sistem ekternal dan penggunan. Dengan kata lain *use case* diagram secara grafis mendeskripsikan siapa yang akan digunakan sistem dan dalam cara

apa pengguna (*user*) mengharapkan interaksi dengan sistem itu. *Use case* secara naratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi.

b. *Class* Diagram

Mengambarkan struktur object sistem. Diagram ini menunjukan *class object* yang menyusun sistem dan juga hubungan antara *class object* tersebut.

c. *Activity* Diagram

Secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *use case, activity* diagram dapat juga digunakan untuk memodelkan *action* yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari *action* tersebut.

d. *State Chart* Diagram

Digunakan untuk modelkan *behaviour* objek khusus yang dinamis diagaram ini mengilustrasikan siklus hidup objek berbagai keadaan yang dapat diasumsikan oleh objek dan *event-event* (kejadian) yang menyebabkan objek beralih dari satu *state* ke *state* yang lainnya.

e. *Sequence* Diagram

Secara grafis menggambarkan bagaimana objek berinteraksi dengan satu sama yang lain melalui pesan pada sekuensi sebuah *use case* atau operasi.

6. *Package* Diagram

   Package diagram utamanya digunakan untuk mengelompokan elemen diagram UML yang berlainan secara bersama-sama ke dalam tingkat pembangunan yang lebih tinggi yaitu berupa sebuah paket. Diagram paket pada dasarnya adalah diagram kelas yang hanya menampilkan paket, disamping kelas, dan hubungan ketergantungan, disamping hubungan khas yang ditampilkan pada diagram kelas.

7. *State Machines* Diagram

   Statechart diagram digunakan untuk memodelkan perilaku dinamis satu kelas atau objek. Statechart diagram memperlihatkan urutan keadaan sesaat (*state*) yang dilalui sebuah objek, kejadian yang menyebabkan sebuah transisi dari suatu state atau aktivitas kepada yang lain.

8. *Communication* Diagram

   Collaboration diagram menggambarkan interaksi antara objek seperti *sequence* diagram, tetapi lebih menekankan pada peran masing-masing objek. Setiap *message* memiliki *sequence number*, dimana *message* dari level tertinggi memiliki nomor 1. Diagram membawa informasi yang sama dengan diagram *sequence,* tetapi lebih memusatkan atau memfokuskan pada kegiatan obyek dari waktu pesan dikirim.

9. *Composite Structure* Diagram

   Diagram struktur komposit adalah diagram yang menunjukan struktur internal *classifier*, termasuk poin interaksinya ke bagian lain dari *system*. Hal ini menunjukan konfigurasi dan hubungan bagian, yang bersama-

sama melakukan perilaku *classifier*. Diagram struktur komposit merupakan jenis diagram struktur yang statis dalam UML, yang menggambarkan struktur internal kelas dan kaloborasi.

10. *Object* Diagram

    Object diagram merupakan sebuah gambaran tentang objek-objek dalam sebuah *system* pada satu titik waktu. Karena lebih menonjolkan perintah-perintah dari pada *class,* object diagram lebih sering disebut sebagai sebuah diagram perintah.

11. *Timing* Diagram

    Timing diagram adalah bentuk lain dari *interaction* diagram, dimana focus utamanya lebih ke waktu. *Timing* diagram sangat berdaya guna dalam menunjukan faktor pembatas waktu diantara perubahan *state* pada objek yang berbeda.

12. *Component* Diagram

    Diagram ini bila dikombinasikan dengan diagram penyebaran dapat digunakan untuk menggambarkan distribusi fisik dari modul perangkat lunak melalui jaringan. Misalnya, ketika merancang sistem *client server* hal ini berguna untuk menunjukan mana kelas atau paket kelas akan berada pada node klien dan mana yang akan berada di *server*.

13. *Deployment* Diagram

    Deployment diagram mengambarkan detail bagaimana komponen di deploy dalam infranstruktur *system,* dimana komponen akan terletak (pada mesin, server atau piranti keras), bagaimana kemampuan jaringan

pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Hubungan antara node (misalnya TCP/IP dan requirement dapat juga didefinisikan dalam diagram ini.

14. *Interaction Overview* Diagram

    Interaction overview diagram adalah pecangkolan secara bersama antara *activity* diagram dengan *sequence* diagram. Interaction overview diagram dianggap sebagai activity diagram dimana semua aktivitas diganti dengan sedikit *sequence* diagram, atau bisa juga dianggap sebagai *sequence* diagram yang dirincikan dengan notasi *activity* diagram yang digunakan untuk menunjukkan aliran pengawasan.

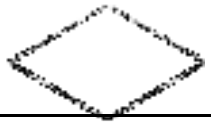
2.1.8 *Flowchart*

Menurut Rini Nuraini dalam jurnal yang berjudul Desain Algoritma Operasi Perkalian Matriks Menggunakan Metode *Flowchart* (2015). *Flowchart* dapat artikan sebagai suatu alat atau dilaksanakan dalam menyelesaikan suatu permaslahan untuk serangkaian simbol-simbol grafis khusus.

Flowchart atau yang lebih dikenal dengan sebutan diagram alir merupakan kumpulan skema yang menunjukan atau menggambarkan rangkaian kegiatan-kegiatan program dari awal sampai akhir. *Flowchart* ini merupakan penggambaran dari urutan langkah-langkah pekerjaan dari suatu algoritma. Tujuan utama pembuatan *flowchart* ini adalah untuk menggambarkan suatu tahapan penyelesaian masalah sederhana, terurai, rapi dan jelas.

Adapun simbol-simbol *flowchart* lihat pada table sebagai berikut:

Tabel 2.1 Simbol- Simbol Flowachart

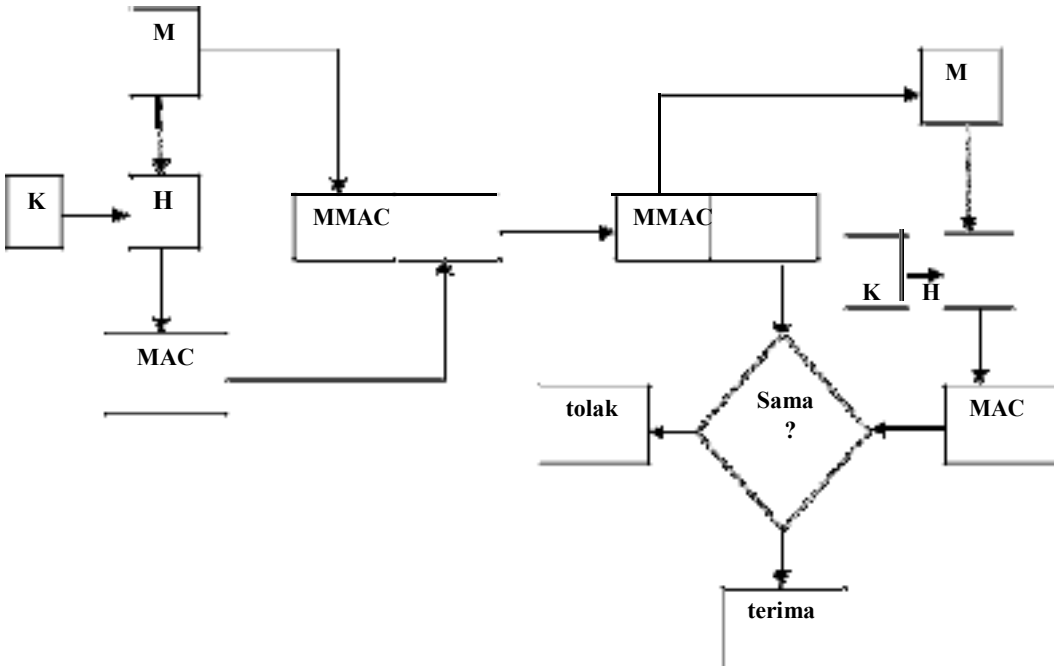
NO	SIMBOL	Nama Simbol	FUNGSI
1		Terminal	Untuk memulai atau mengakhiri suatu program
2		Proses	Suatu simbol yang menunjukan setiap pengolahan yang dilakukan
3		Input-output	Untuk memasukan menunjukan hasil dari suatu proses
4		Decision	Suatu kondisi yang akan menghasilkan beberapa kemungkinan jawaban atau pilihan
5		Preparation	Suatu simbol yang menyediakan tempat pengolahan
6		Connector	Suatu prosedur penghubungan yang akan dimasukan atau keluar melalui simbol ini dalam lembar yang sama
7		Off-Page Connector	Merupakan simbol masuk atau keluarnya suatu prosedur pada lembaran kertas lainnya
8		Predefined Process	Untuk menyatakan sekumpulan langkah proses yang ditulis sebagai prosedur
9		Display	Simbol untuk *output* yang ditunjukan ke suatu *device* sperti *printer* Dan sebagainya
10		Storage Data	Penyimpanan *file* secara sementara
11		Megnetic Dask	Menunjukan input/output hardisk (media penyimpanan)

(**Sumber :** Rini Nuraini 2015)

2.1.9 Kode Otentikasi Pesan (*Message Authentication Code*)

Fungsi *hash* dapat dipakai untuk mewujudkan layanan otentikasi pesan dengan memberikan keluaran kode otentikasi pesan atau dikenal dengan MAC**.** Fungsi *hash* yang dipakai pada otentikasi pesan dapat memiliki masukan sebuah kunci sehingga disebut juga fungsi *hash* dengan kunci. Misalnya A dan B telah berbagi kunci rahasia K dan sebuah fungsi h. Ketika A mengirim sebuah pesan M, maka A juga menyertakan nilai $y = h_K(M)$. B menerima M dan y dan memverifikasi apakah $y = h_K(M)$ jika benar, maka B memastikan bahwa pesan tidak berubah dan memang berasal dari si A. dan contoh gambar layanan MAC dengan fungsi *hash* bisa dilihat dibawah.

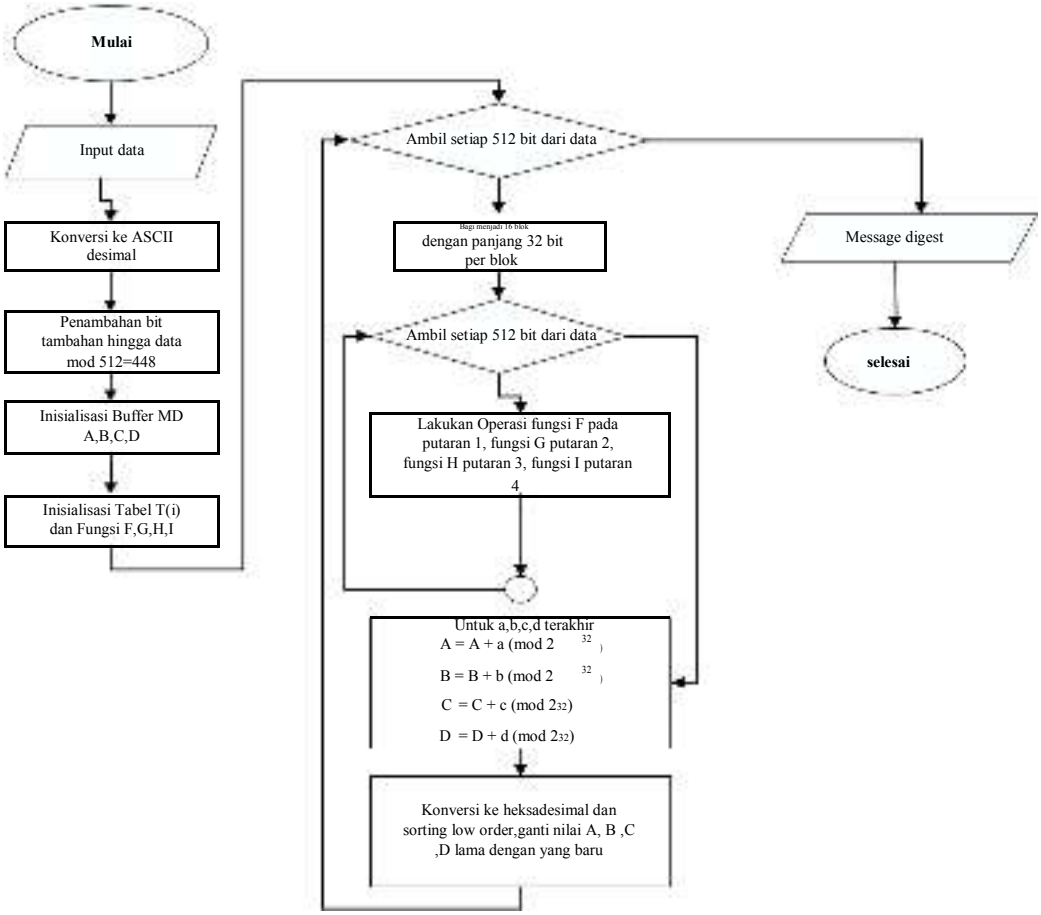
Message authentication code (MAC) adalah suatu kode yang dibangkitkan dari pesan dengan mengunakan kunci tertentu dan fungsi tertentu fungsi tersebut dapat berupa fungsi hash ataupun fungsi enkripsi yang diopersikan dengan kode CBC, jika fungsi yang digunakan adalah fungsi hash, maka MAC tersebut dinamankan *keyed-Hash Message authentication (KHMAC),* pada KHMAC, kunci digunakan oleh penerima pesan untuk memverifikasi nilai *hash*. Mekanisme dari KHMAC secara sederhana adalah, penggunan menggabungkan kunci pesan dengan pesan yang ingin diotentifikasi, lalu menghitung nilai *hash*-nya dengan menggunakan fungsi *hash* tertentu.



Gambar 2.4. *Message Authentication Code*
(Sumber: Ainul Wardah, 2017)

2.1.10 Pengertian *Message Digest*

Menurut Muhammad Husni Rifqo dalam Jurnal Pseudocode yang berjudul Aplikasi Keamanan Lembar Hasil Studi Menggunakan Algoritma *Message Digest* (2015), message digest adalah fungsi hash (prosedur terdefinisi atau fungsi matematika yang mengubah variable dari suatu data yang berukuran besar menjadi lebih sederhana) kriptografi yang digunakan secara luas dengan hash value 128-bit. Message digest dimanfaartkan dalam berbagai aplikasi keamanan dan umumnya untuk menguji intergritas file. Contoh pembangkitan *message digest* :



Gambar 2.5. Pembentukan *Message Digest*
(Muhammad Husni Rifqo 2015)

2.1.11 Pengertian Algoritma *Adler*32

Menurut Andysah Putera Utama Siahaan dalam *Internal Journal of Computer Science And Software Engineering* (IJCSSE) yang berjudul *A Three Layer – Visual Hash Function Using Adler32* (2016), algoritma *Adler*32 merupakan algoritma checksum yang di temukan oleh *Mark Adler* pada tahun 1995. Algoritma ini mampu menjalankan tesredun dan sisiklik sepanjang 32 bit untuk menjaga dari modifikasi mendadak terhadap data, seperti distorsi yang terjadi pada saat transmisi data. Algoritma ini memegang peran penting dalam perhitungan dalam perangkat lunak karena kecepatannya.

Adler32 juga lemah untuk pesan pendek karena jumlah A tidak terbungkus. Jumlah maksimum pesan 128-byte adalah 32640, yang berada dibawah nilai 65521 yang digunakan oleh operasi modulo, yang berarti bahwa sekitar setengah dari ruang output tidak digunakan, dan distribusi dalam bagian yang digunakan adalah tidak seragam. Penjelasan lebih lanjut dapat ditemukan di RFC 3309 yang mengamanatkan penggunan CRC32C bukan *adler32* untuk SCTP, protokol transmisi control aliran *adler32* juga terbukti untuk perubahan incremental kecil dan juga lemah untuk string yang dihasilkan dari awalan umum dan angka berurutan (seperti nama label yang dibuat secara otomatis oleh generator kode khas).

Adler32 adalah untuk *hash* cepat, memiliki sedikit ruang kecil, dan algoritma sederhana. Tingkat tabrakannya rendah, tetapi tidak cukup rendah untuk aman. MD5, SHA dan *hash* kriptografi/ aman lainnya (atau intisari pesan) memiliki ruang bit jauh lebih besar dan algoritma yang lebih kompleks, sehingga memiliki tabrakan. Bandingkan dengan SHA2-256, misalnya; 256 bit dibandingkan dengan 32 bit *adler32*. Adler tidak memiliki tujuannya, dalam table *hash* misalnya, atau pemeriksaan integritas data yang cepat. Namun, itu tidak dirancang dengan tujuan yang sama seperti MD5 atau mencerna aman lainnya.

Namun penggunaan utama *Adler32*'s adalah Zlib debug implementasi *Adler32* dan CRC-32 tidak mencukupi untuk tujuan apapun yang memerlukan tingkat akurasi yang tinggi. Pihak-pihak yang tidak bertanggung jawab dapat mencoba untuk mencuri data yang sedang dikirim. *Adler32* dapat

memverifikasi data untuk menjamin intergritas informasi *Adler32 checksum* yang diperoleh dengan menghitung *checksum* 16-bit dua A dan B dan menggabungkan hasil 16-bit ke 32-bit integer.

A adalah jumlah dari semua *byte* plus satu dan B adalah jumlah individu a nilai-nilai dari setiap langkah pada awalnya A diinisialisasi ke-1 B ke-0, nomor diperoleh dengan modulus 65521, nomor perdana paling signifikan lebih kecil dar 216. *Byte* disimpan dalam urutan dan B menempati dua *byte* paling signifikan fungsi ini dapat dinyatakan sebagai nilai algoritma *Adler*32 yang berjumlah 32 bit di dapatkan dengan menggabungkan 2 buah 16 bit *checksum*.

A dan B dengan aturan sebagai berikut:

1. Merupakan jumlah dari semua *bytes* dalam *string*, di inisialisasi Dengan 1.
2. Merupakan jumlah individual dari masing-masing angkah, nilai B diinisalisasi dengan 0.

Algoritma *checksum* lebih cepat digunakan dalam zlib untuk memverifikasi hasil dekompresi. Ini terdiri dari 2 jumlah modulo 65521. Mulai dengan s1 = 1 dan s2 = 0, kemudian untuk setiap byte x, s1 = s1 + x, s2 + s1. Dua jumlah digabungkan menjadi nilai 32-bit dengan s1 dan s2 dalam 16 bit tinggi.

Jumlah *adler* dihitung lebih dari 8-bit dari pada kata-kata 16 bit, menghasilkan dua kali jumlah iterasi loop. Ini menghasilkan checksum *adler32* yang mengambil antara satu setengah hingga dua kali lipat selama checksum

fletcher untuk data byte- *aligned*, *adler32* lebih cepat dari pada checksum *fletcher* yang di implementasikan dengan benar (misalnya, yang ditemukan dalam format data hirarki).

Sebuah checksum *adler32* diperoleh dengan menghitung dua checksum 16-bit A dan B dan menggabungkan bit-bit mereka kedalam *integer* 32-bit. A adalah jumlah sebuah byte dalam aliran plus satu, dan B adalah jumlah nilai individu A dari setiap langkah. Pada awalnya menjalankan *adler32* ,A diinisialisasi ke 1, B ke 0. Jumlahnya dilakukan modulo 65521 (bilangan prima terbesar lebih kecil dari 2^{16}). Byte disimpan dalam urutan jaringan (*big edian*), B menempati dua byte paling signifikan.

Fungsi *adler-32* ini dapat diekspresikan sebagai berikut:

$A = 1 + D_1 + D_2 + \ldots\ldots + D_n \pmod{65521}$

$B = (1 + D_1) + (1 + D_1 + D_2) + \ldots\ldots + (1 + D_1 + D_2 + \ldots.. + D_n) \pmod{65521} = n^x D_1 + (n-1) \times D_2 + (n-2) \times D_3 + \ldots\ldots..+ D_n + n \pmod{65521}$

$\text{Adler-32 (D)} = B \times 65536 + A$

Dimana D adalah *string byte* masukan, dan n adalah panjang dar D.

CRC-32 adalah salah satu jenis algoritma Cyclic Redundancy Check (CRC). CRC adalah salah satu fungsi hash yang berguna untuk mendeteksi kerusakan data dalam proses transmisi ataupun penyimpanan. CRC menghasilkan suatu checksum yaitu suatu nilai dihasilkan dari fungsi hash-nya, dimana nilai ini akan digunakan untuk mendeteksi error pada transmisi ataupun penyimpanan data. Nilai CRC dihitung dan digabungkan sebelum dilakukan transmisi data atau penyimpanan, dan kemudian penerima akan melakukan verifikasi apakah data yang diterima tidak mengalami perubahan ataupun kerusakan. CRC cukup terkenal karena mudah diterapkan dalam hardware, dan mudah dilakukan analisis secara matematika. Prinsip utama yang digunakan adalah dengan melakukan pembagian polinomial dengan mengabaikan bit-bit carry. Cara yang biasa digunakan adalah dengan menggunakan tabel CRC yang nilainya telah dihitung sebelumnya, sehingga dapat menghemat waktu dan meminimalisir kesalahan di tengah perhitungan. CRC adalah menganggap file sebagai suatu string yang besar, yang terdiri dari bit-bit, dan kita operasikan sebagai suatu bilangan polynomial yang sangat besar. Untuk menghitung nilai CRC, kita membagi bilangan polynomial, sebagai representasi dari file, dengan suatu bilangan polynomial kecil yang sudah terdefinisi untuk jenis varian CRC tertentu. Nilai CRC adalah sisa hasil bagi tersebut, yang biasa disebut dengan checksum.

BAB III

ANALISA DAN PERANCANGAN

Bab analisa dan perancangan sistem berisi tentang pembahasan Analisa dan perancangan sistem yang akan dibangun. Pembahasan ditunjukan untuk menguraikan kebutuhan-kebutuhan dalam pengembangan sistem yang dibuat.

3.1 Analisa sistem

Analisa sistem (*system analyst*) dapat didefinisikan sebagai pengurain dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya dengan dengan maksud mengidentifikasi dan mengevaluasi permasalahan-permaslahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya.

Analisa sistem merupakan penguraian dari suatu sistem yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permaslahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya. Hal-hal yang akan dianalisa pada tahap Analisa sistem ini adalah Analisa prosedur sistem yang sedang berjalan, Analisa aliran informasi, Analisa pengkodean, analisi basis data dan Analisa kebutuhan fungsional.

Berdasarkan analisa yang dilakukan oleh penulis terhadap sistem yang sedang berjalan. Sistem hanya melakukan perhitungan dan menghasilkan nilai *adler32* desimal dan nilai *adler32* heksadesimal.

3.2 Analisa Kebutuhan Fungsional

Kebutuhan fungsional adalah Analisa yang dibutuhkan yang dibuthkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi ini juga meliputi elemen-elemen atau komponen-komponen apa saja yang dibutuhkan untuk sistem yang akan dibangun sampai dengan sistem tersebut diimplemetasikan. Analisa kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan sistem dan proses yang akan dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan.

1. Mengimplemtasikan pengguna *Visual Basic . Net* 2012 dalam membuat "Perancangan Apalikasi *Digital Signature* dengan *Adler-32*"
2. Aplikasi harus dapat melakukan perhitungan pada *plaintext*.
3. Aplikasi harus bisa menghasilkan nilai *adler-32* desimal dan heksadesimal.

3.3 Analisa Kebutuhan NonFungsional

Kebutuhan ini adalah berupa tipe kebutuhan yang berisikan properti perilaku yang memiliki oleh sistem. Berikut adalah kebutuhan nonfungsional yang dimiliki sistem:

1. Operasional
    a. Dapat digunakan pada sistem operasi *Microsoft Windows* 7/8/10 secara *stand alone.*
    b. Sistem dibangun dengan menggunakan komponen *IDE Visual Basic* 2012.

c. Spedifikasi komputer standard untuk menjalankan *Visual Studio* 2012 yaitu *Processor Pentium* IV 2,6 GHz, Memori 512 MB Kartu Grafik 128 MB.

2. Kinerja

   Waktu yang diperlukan dalam mengeksekusi sistem perancangan aplikasi *digital signature* dengan *adler-32* dibangun dengan konsep serderhana dan cukup ringan. Sehingga eksekusi dan proses plaintext, proses, sangat cepat dan sangat mudah digunakan.

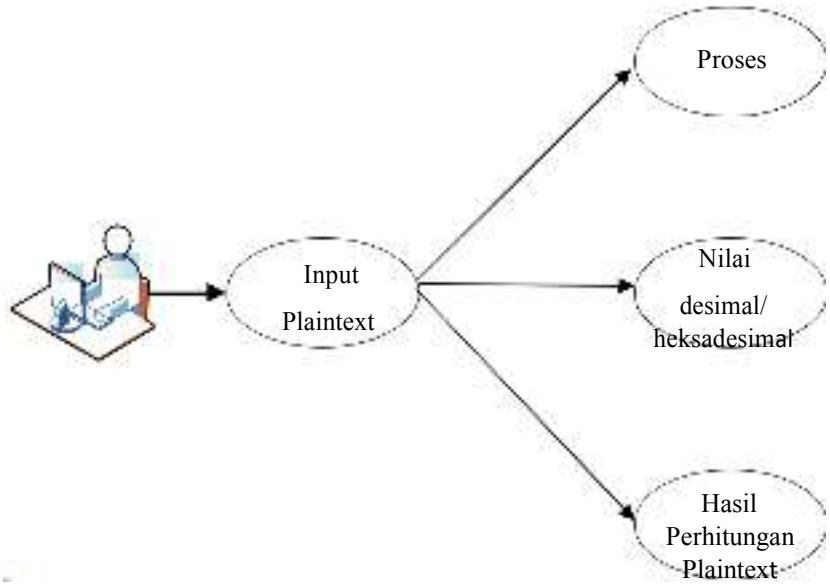
3.4 Rancangan Sistem

Sub bab ini berisikan tentang perancangan sistem yang akan dibangun, dalam hal ini penulis menjabarkan bahwa rancangan sistem tersebut seperti gambar yang ada dibawah :

3.4.1 *Use Case* Diagram

Use Case diagram merupakan pemodelan untuk menggambar kelakuan dari sistem yang dibuat dan medeskripsikan sebuah interaksi antara satu dan lebih aktor dengan sistem yang dibuat serta digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

berikut ini merupakan *use case* diagram dari "perancangan aplikasi *digital signature* dengan *adler32* terlihat pada gambar dibawah ini:



Gambar 3.1. *Use Case* Diagram Sistem

pada gambar diagram *use case*` ini, penulis menjelaskan bahwa *user* adalah individu yang menjalankan sistem. Ketika sistem dijalankan *user* akan menampilkan *interface* kemudian *user* akan menginputkan *plaintext* dan selanjutnya *user* mengklik tombol proses dan secara otomatis *plaintext* yang diproses akan menampilkan hasil hitungan dan nilai desimal dan heksadesimal.

3.4.2 *Activity* Diagram

Activity diagram merupakan bentuk visual dari alur kerja yang berisi aktivitas dan tindakan, yang juga dapat berisi pilihan, atau pengulangan. Dalam *Unified Modeling Language* (UML), diagram aktivitas dibuat untuk menjelaskan aktivitas komputer maupun alur aktivitas dalam organisasi. Selain itu diagram aktivitas juga menggambarkan alur kontrol secara garis besar.

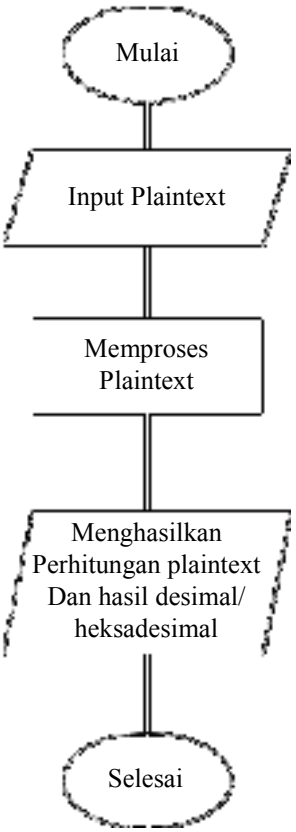
Berikut ini merupakann *activity* diagram dari sistem " perancangan aplikasi *digital signature* dengan *adler-32*" terlihat pada gambar dibawah ini :

Gambar 3.2. *Activity* Diagram

3.4.3 Perancangan *Flowchart*

Rancangan ini digunakan untuk mendesain dan mempresentasikan program. Sebelum pembuatan program, *flowchart* digunakan untuk mempermudah progremer dalam penyusunan alur program yang akan dibuatnya. Flowchart adalah bagan yang menunjukan aliran yang di dalam suatu program atau prosedur sistem secara logika. Flowchart mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah, dan merupakan salah satu cara penyajian algoritma.

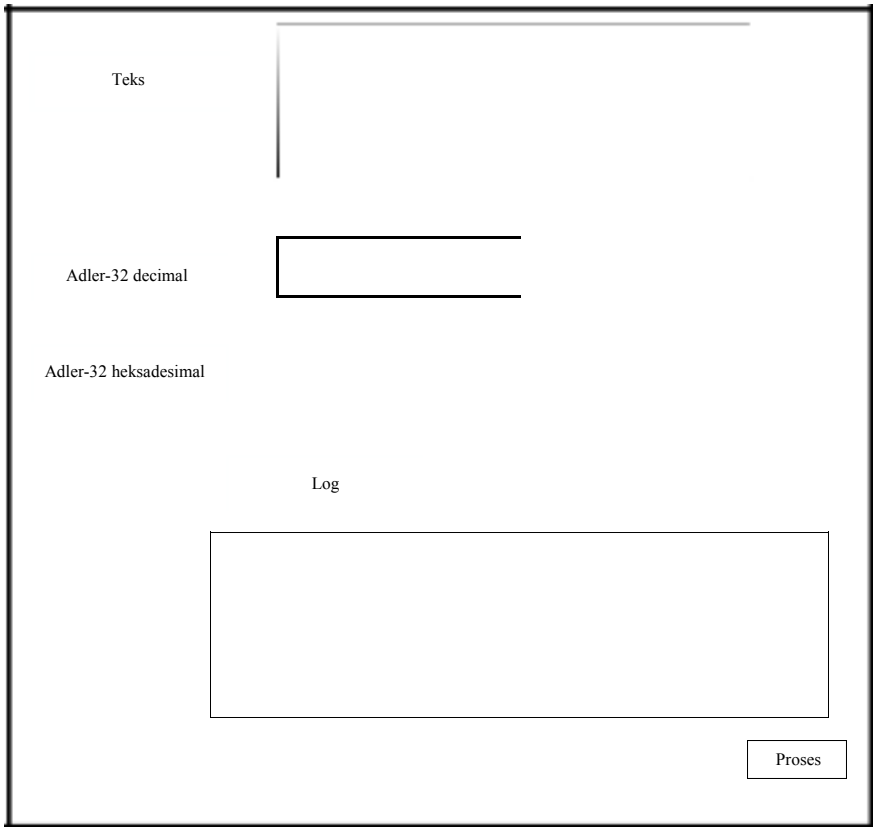
Flowchart untuk perancangan sistem dan alur kerja sistem dapat dilihat pada gambar 3.3 dibawah :



Gambar 3.3. Diagram Flowchart

3.4.4 *User Interface*

User Interface merupakan tampilan awal dari program yang merupakan mekanisme komunikasi antara pengguna dan user dengan sistem. Antar muka interface dapat menerima informasi dari pengguna user dan memberikan informasi kepada pengguna user untuk membantu mengarahkan alur penelusuran masalah sampai ditemukan solusi.



Gambar 3.4. Rancangan *User Interface*

Keterangan Gambar 3.4 :

1. *Plaintext*

   Fungsi dari kolom ini yaitu sebagai inputan text yang nantinya akan diproses dan akan mendapat hasil perhitungannya.

2. *Adler32* desimal

   Fungsi dari kolom ini akan menampilkan hasil perhitungan yang sudah proses sebelumnya yaitu berupa angka decimal.

3. *Adler32* heksadesimal

   Fungsi dari kolom ini akan menampilkan hasil perhitungan yang sudah diproses sebelumnya yaitu berupa angka heksadesimal

4. Log

   Fungsi log yaitu akan menampilkan hasil perhitungan dari *plaintext*.

BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1 Implementasi Sistem

Sistem merupakan sekelompok komponen dan elemen yang digabungkan menjadi satu untuk mencapai tujuan tertentu. Dalam implemtasi sistem yang sudah dirancang, maka dibutuhkan beberapa perangkat yang sangat penting agar sistem yang sudah dirancang dapat berjalan pada saat diimplementasikan, berikut ini adalah beberapa perangkat yang dibutuhkan:

1. Perangkat Keras

Perangkat keras merupakan bagian dari sistem komputer yang merupakan perangkat yang dapat dirabah dan dilihat secara fisisk serta prangkat yang menjalankan intruksi dari perangkat lunak (*software*). Perangkat dapat disediakan adalah berupa sistem komputer yang lengkap. pada saat sistem ini diuji, digunakan spesifikasi komputer sebagai berikut:

Processor : Intel® celeron ® CPU 1007U @ 1.50GHz 1.50GHz

RAM : 2.00 GB (1.80 GB usable)

HD : 500 GB

Monitor : 19”

2. Perangkat Lunak Komputer

Perangkat lunak merupakan kumpulan beberapa perintah yang dieksekusi oleh mesin komputer dalam menjalankan pekerjaannya.

Perangkat lunak ini merupakan catatan bagi mesin komputer untuk menyimpan perintah, maupun dokumen serta arsip lainnya.

Untuk cara mengimplementasikan sistem ini, harus dibutuh beberapa perangkat lunak yang sudah ada di komputer yang digunakan. Berikut perangkat yang digunakan.

3. Sistem Operasi

Sistem Operasi merupakan perangkat lunak komputer atau *software* yang bertugas untuk melakukan kontrol dan manajemen perangkat keras dan juga operasi-operasi dasar sistem, termasuk menjalankan *software* aplikasi seperti program-program pengolah data yang bisa digunakan untuk mempermudah kegiatan manusia.

Sistem Operasi yang digunakan dalam pengujian ini adalah *windows* 10, dan aplikasi ini masih bisa berjalan dengan baik pada sistem operasi *windows xp*, *windows* 7, *windows* 8, dan *windows* 8,1.

Adapun perangkat minimum agar system ini tetap berjalan secara optimal adalah sebagai berikut:

RAM : 512 MB

HD :1GB

Monitor : 19”

4.2 Hasil Perhitungan *Adler32*

Plaintext yang akan dihitung: PANCABUDI

Tabel 4.1 Perhitungan *plaintext*

HURUF	Perhitungan = a 1	Perhitungan = b 0
P=80	a = a + P = 81	b = b + a = 81
A=65	a = a + A = 146	b = 81 + 146 = 227
N=78	a = 146 + 78 = 224	b = 227 + 224 = 451
C=67	a = 224 + 67 = 291	b = 451 + 291 = 742
A=65	a = 291 + 65 = 356	b = 742 + 356 = 1098
B=66	a = 356 + 66 = 422	b = 1098 + 422 = 1520
U=85	a = 422 + 85 = 507	b = 1520 + 507 = 2027
D=68	a = 507 + 68 = 575	b = 2027 + 575 = 2602
I=73	a = 575 + 73 = 648	b = 2602 + 648 = 3250
Hasil	(b * 65536) + a 3250 x 65536) + 648	Hasil akhir = 212992648

4.3 Pengujian Sistem

1. Tampilan awal program

Pada *from* ini, *user* bisa langsung masukkan teks yang akan diproses dengan mengklik *start* di aplikasi *visual studio* lalu akan muncul *from* seperti diatas dan *user* langsung dapat memasukan teks yang akan diproses setelah itu untuk mengetahui hasil perhitungan dari teks yang sudah dimasukan akan ditampilkan di*from* selanjutnya.

Gambar 4.1 Tampilan Awal

Di tampilan ini ada beberapa kolom yaitu:

a. Teks : Untuk menampilkan teks yang akan diproses.

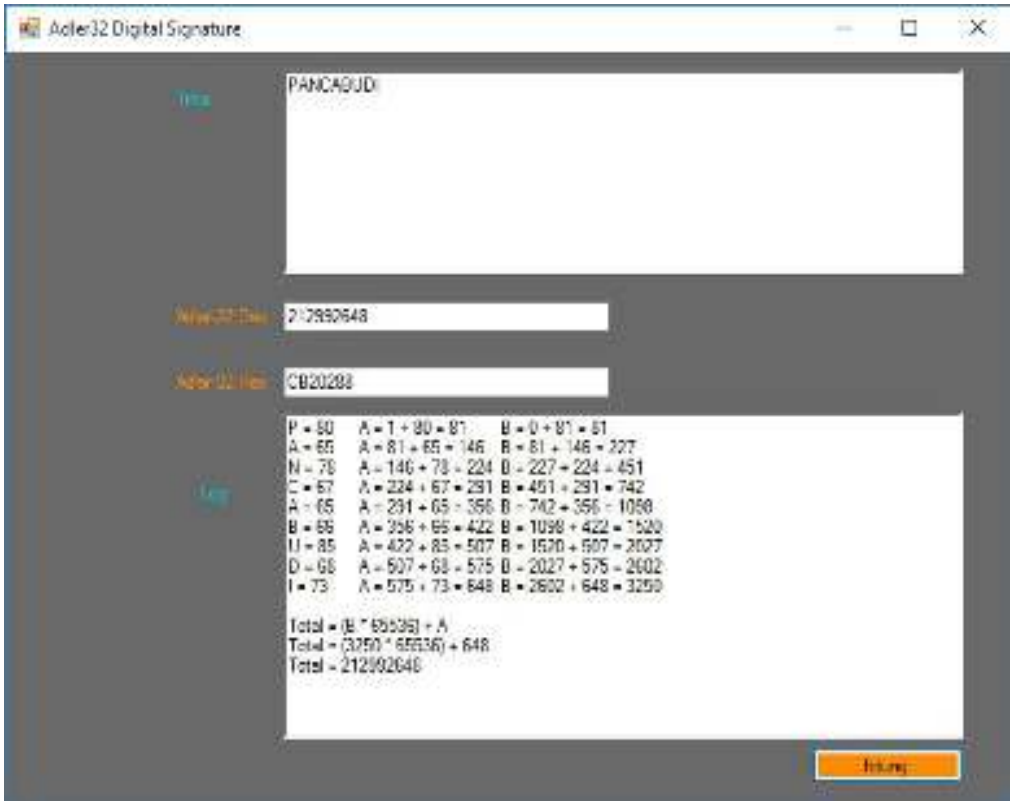
b. *Adler-32* desimal : Untuk menampil nilai desimal yang didapatkan.

c. *Adler-32* heksadesimal : Untuk menampilkan nilai heksadesimal yang didapatkan.

d. *Log* : untuk menampilkan hasil perhitungan teks.

e. Hitung : untuk memproses hasil perhitungan.

2. Tampilan from hasil hitung



Gambar 4.2 Tampilan hasil perhitungan

Pada *from* ini, *user* dapat melihat hasil hitungan teks yang sudah di*input*kan pada *from* sebelumnya dan mengklik tombol hitung maka keluarlah hasil hitungannya dan pada *from* ini bukan hanya menampilkan hasil perhitungan teksnya saja, akan tetapi juga menampil hasil *adler-32* desimal dan menampilkan hasil *adler-32* heksadesimalnya.

4.4 Kelebihan dan Kekurangan Sistem

Adapun beberapa kelebihan dan kekurangan dari sistem ini dapat dilihat sebagai berikut :

Kelebihan

1. Proses perhitungan berjalan dengan baik sehingga dapat menghemat daya komputer dan waktu.
2. Sistem ini dapat menerima semua jenis bentuk format teks sebagai inputan.
3. Kapasitas teks yang dapat ditampung cukup besar, yaitu berkisar 100 MB.

1. Sistem masih belum mengetahui bagian mana yang sudah dimodifikasi dan sudah sejauh mana yang telah dimodifikasi.
2. Sistem ini masih belum lengkap untuk mengenai tampilannya.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan Analisa dan pengujian yang diatas maka peneliti dapat memberika beberapa kesimpulan sebagai berikut:

1. Sistem yang sudah berjalan saat ini dapat dibilang sederhana, cukup ringan dan sangat mudah digunakan.
2. *Output* yang ditampilkan dalam aplikasi ini berupa bilangan decimal dan hexadecimal.
3. Bilangan decimal dan hexadecimal itu merupakan sebuah kunci untuk verifikasi sebuah text yang dimasukkan.
4. Aplikasi ini dapat dikembangkan dengan adanya pengirim dan penerima.

5.2 Saran

Ada beberapa saran yang dapat disampaikan oleh penulis untuk pengembangkan lebih lanjut untuk penelitian ini adalah sebagai berikut:

1. Diharapkan tidak hanya data text saja yang bisa diamankan dengan aplikasi ini, kedepannya aplikasi ini bisa dikembangkan agar bisa menambahkan *file*, *image*, maupun video.
2. Penulis menyadari bahwa di dalam sistem ini masih banyak kekurangan, maka diharapkan kedepannya dapat dilakukan pengembangan secara intensif supaya sistem ini dapat dapat lebih baik lagi.

3. Dengan demikian dapat disimpulkan bahwa *digital signature* sangat bermanfaat sebagai salah satu teknologi pengaman jaringan. Fungsinya untuk memastikan keaslian data sangat berguna disaat pembajakan dan pemalsuan data semangkin marak.

DAFTAR PUSTAKA

Alfry, Aristo Sainlae Jansen. 2012. Keamanan Data; *Digital Signature* SHA-512 Dan RSA. Diponegoro: Library@Adm.uksw.edu.

Aryza, S., Irwanto, M., Lubis, Z., Siahaan, A. P. U., Rahim, R., & Furqan, M. (2018). A Novelty Design Of Minimization Of Electrical Losses In A Vector Controlled Induction Machine Drive. In IOP Conference Series: Materials Science and Engineering (Vol. 300, No. 1, p. 012067). IOP Publishing.

Bahri, S. (2018). Metodologi Penelitian Bisnis Lengkap Dengan Teknik Pengolahan Data SPSS. Penerbit Andi (Anggota Ikapi). Percetakan Andi Ofsset. Yogyakarta.

Fitriani, W., Rahim, R., Oktaviana, B., & Siahaan, A. P. U. (2017). Vernam Encypted Text in End of File Hiding Steganography Technique. Int. J. Recent Trends Eng. Res, 3(7), 214-219.

Gata, Windu dan Gata, Grace. 2013. Sukses Membangun Aplikasi Penjualan dengan Java. Jakarta: Elex Media Komputindo.

Ihwani, Mohamad. 2016. Model Keamanan Informasi Berbasis *Digital signature* Dengan Algoritma RSA. Medan: mohamadihwani@unimed.ac.id.

Kurniawan, H. (2018). Pengenalan Struktur Baru untuk Web Mining dan Personalisasi Halaman Web. Jurnal Teknik dan Informatika, 5(2), 13-19.

Maryanto, Budi. 2008. Fungsi *Hash* Satu Arah Untuk Enkripsi. Bandung: Media Informatika, budimaryanto@gmail.ac.id.

Mayasari, Nova. "Comparison of Support Vector Machine and Decision Tree in Predicting On-Time Graduation (Case Study: Universitas Pembangunan Panca Budi)." Int. J. Recent Trends Eng. Res 2.12 (2016): 140-151.

Munir Renaldi. 2006. Kriptografi, Jakarta: Informatika.

Maulana Gun Gun. 2017. Pembelajaran Dasar Algoritma Dan Pemrograman menggunakan EL-Goritma Berbasis Web. Bandung: Teknik Elektromekanik, gungun@poiman-bandung.ac.id.

Nuraini Rini. 2015. Desain Algoritma Operasi Perkalian Matriks Menggunkan Metode *Flowchart*. Jakarta Selatan: rini_rna@bsi.ac.id.

Rahim, R., Supiyandi, S., Siahaan, A. P. U., Listyorini, T., Utomo, A. P., Triyanto, W. A., ... & Khairunnisa, K. (2018, June). TOPSIS Method Application for Decision Support System in Internal Control for Selecting Best Employees. In Journal of Physics: Conference Series (Vol. 1028, No. 1, p. 012052). IOP Publishing.

Rahim, R. (2018, October). A Novelty Once Methode Power System Policies Based On SCS (Solar Cell System). In International Conference of ASEAN Prespective and Policy (ICAP) (Vol. 1, No. 1, pp. 195-198).

Rakhmadi, A,& Nugroho. 2016. *Web-Based Application for Single Elimination Tournament Using Linear Congruential Generator, ISETH 2016 (The 2nd International Conference on Science, Technology, and Humanity,* 273-285.

Ramadhan, Z., Zarlis, M., Efendi, S., & Siahaan, A. P. U. (2018). Perbandingan Algoritma Prim dengan Algoritma Floyd-Warshall dalam Menentukan Rute Terpendek (Shortest Path Problem). JURIKOM (Jurnal Riset Komputer), 5(2), 135-139.

RifqoHusni Muhammad. 2015. Aplikasi Keamanan Lembar Hasil Studi Menggunakan Algoritma *Message Digest.* Bengkulu: yoviapriandisyah@gmail.com.

Sari, R. D., Supiyandi, A. P. U., Siahaan, M. M., & Ginting, R. B. (2017). A Review of IP and MAC Address Filtering in Wireless Network Security. Int. J. Sci. Res. Sci. Technol, 3(6), 470-473.

Siahaan, A. P. U. 2016. *.A Three Layer - Visual HASH Function Using Adler-32.* Medan: www.IJCSSE.org, andiesiahaan@gmail.com.

Suherman, S., & Khairul, K. (2018). Seleksi Pegawai Kontrak Menjadi Pegawai Tetap Dengan Metode Profile Matching. IT Journal Research and Development, 2(2), 68-77.

Tarigan, A. D., & Pulungan, R. (2018). Pengaruh Pemakaian Beban Tidak Seimbang Terhadap Umur Peralatan Listrik. RELE (Rekayasa Elektrikal dan Energi): Jurnal Teknik Elektro, 1(1), 10-15.

Tarigan, A. D. (2018, October). A Novelty Method Subjectif of Electrical Power Cable Retirement Policy. In International Conference of ASEAN Prespective and Policy (ICAP) (Vol. 1, No. 1, pp. 183-186).

Tasril, V. (2018). Sistem Pendukung Keputusan Pemilihan Penerimaan Beasiswa Berprestasi Menggunakan Metode Elimination Et Choix Traduisant La Realite. INTECOMS: Journal of Information Technology and Computer Science, 1(1), 100-109.

Wahyuni, S., Lubis, A., Batubara, S., & Siregar, I. K. (2018, September). IMPLEMENTASI ALGORITMA CRC 32 DALAM MENGIDENTIFIKASI KEASLIAN FILE. In Seminar Nasional Royal (SENAR) (Vol. 1, No. 1, pp. 1-6).

Wardah Ainul. 2017. Perancangan Aplikasi Pembelajaran Kriptografi Pada Algoritma Fungsi *Hash* Menggunakan Metode *Computer Based Learning*. Medan: Teknik Informatika, ainulwarda91@yahoo.com.

Wibowo, P., Lubis, S. A., & Hamdani, Z. T. (2017). Smart Home Security System Design Sensor Based on Pir and Microcontroller. International Journal of Global Sustainability, 1(1), 67-73.

Widyartono Agustinus. 2011.” Algoritma Elgamal Untuk Enkripsi Data Menggunakan GNPUG”, Vol.1.