PENGAMANAN JAWABAN UJIAN *COMPUTER BASED TEST* (CBT) DENGAN MENGGUNAKAN ALGORITMA RSA DAN FUNGSI HMAC BERBASIS ALGORITMA SHA-1

SKRIPSI

DEO VALIANDRO. M

H13116306

PROGRAM STUDI ILMU KOMPUTER DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

OKTOBER 2020

PENGAMANAN JAWABAN UJIAN *COMPUTER BASED TEST* (CBT) DENGAN MENGGUNAKAN ALGORITMA RSA DAN FUNGSI HMAC BERBASIS ALGORITMA SHA-1

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains pada Program Studi Ilmu Komputer Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

DEO VALIANDRO. M

H13116306

PROGRAM STUDI ILMU KOMPUTER DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

OKTOBER 2020

LEMBAR PERNYATAAN KEOTENTIKAN

Saya yang bertanda tangan di bawah ini menyatakan dengan sungguh-sungguh bahwa skripsi yang saya buat dengan judul:

PENGAMANAN JAWABAN UJIAN *COMPUTER BASED TEST* (CBT) DENGAN MENGGUNAKAN ALGORITMA RSA DAN FUNGSI HMAC BERBASIS ALGORITMA SHA-1

adalah benar hasil karya sendiri, bukan hasil plagiat dan belum pernah dipublikasikan dalam bentuk apapun.

Makassar, 06 Oktober 2020

Deo Valiandro. M

NIM. H13116306

PENGAMANAN JAWABAN UJIAN *COMPUTER BASED TEST* (CBT) DENGAN MENGGUNAKAN ALGORITMA RSA DAN FUNGSI HMAC BERBASIS ALGORITMA SHA-1

Disetujui oleh:

Pembimbing Utama	**Pembimbing Pertama**
Dr. Eng. Armin Lawi, S.Si., M. Eng.	**Dr. Hendra, S.Si., M.Kom.**
NIP. 19720423 199512 1 001	**NIP. 19760102 200212 1 001**

Pada 06 Oktober 2020

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama	: Deo Valiandro. M
NIM	: H13116306
Program Studi	: Ilmu Komputer
Judul Skripsi	: Pengamanan Jawaban Ujian *Computer Based Test* (CBT) Dengan Menggunakan Algoritma RSA Dan Fungsi HMAC Berbasis Algoritma SHA-1

Telah berhasil mempertahankan di hadapan dewan penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Sains pada Program Studi Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

DEWAN PENGUJI

		Tanda Tangan
1. Ketua	: Dr.Eng. Armin Lawi, S.Si., M.Eng.	(.....)
2. Sekretaris	: Dr. Hendra, S.Si., M.Kom.	(.....)
3. Anggota	: Dr. Muhammad Hasbi, M.Sc.	(.....)
4. Anggota	: Andi Muh. Amil Siddik, S.Si., M.Si.	(.....)

Ditetapkan di : Makassar

: 06 Oktober 2020

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah Tritunggal, Allah Bapa, Anak-Nya yang Tunggal Yesus Kristus dan Roh Kudus atas segala penyertaan dan kasih-Nya sehingga penyusunan skripsi yang berjudul "**Pengamanan Jawaban Ujian *Computer Based Test* (CBT) dengan Menggunakan Algoritma RSA dan Fungsi HMAC Berbasis Algoritma SHA-1**" dapat diselesaikan dengan baik. Penyertaan Tuhan yang tidak pernah berhenti merupakan suatu kasih karunia selama penulis menempuh pendidikan di Program Studi Ilmu Komputer Departemen Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

Rasa hormat dan terima kasih yang tak terhingga juga penulis ucapkan kepada Ayahanda **Morison, S.Th.** dan Ibunda **Ramalia** yang telah merawat, membesarkan penulis dengan penuh kasih sayang dan senantiasa memberi dukungan doa, moril, tenaga, materi dan waktu sehingga penulis dapat menyelesaikan pendidikan di perguruan tinggi. Juga kepada adik penulis, **Prionaray Bram. M** yang senantiasa mendoakan dan memotivasi penulis.

Demikian pula penulis menyampaikan ucapan terima kasih kepada Bapak dan Ibu dosen Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin atas segala ilmu yang telah diberikan selama penulis menempuh pendidikan. Terkhusus pada Almarhum **Dr. Loeky Haryanto, MS. M.Sc, M.Math.** selaku dosen pengajar mata kuliah Kriptografi yang telah banyak memberikan ilmunya kepada penulis. Ucapan terima kasih yang sama penulis ucapkan kepada Bapak **Dr. Eng. Armin Lawi, M. Eng** selaku pembimbing utama dan juga sebagai dosen pembimbing akademik serta Bapak **Dr. Hendra S.Si., M.Kom.** selaku pembimbing pertama yang dengan penuh keikhlasan telah meluangkan waktu dan pikirannya untuk memberikan petunjuk dan berbagi ilmu kepada penulis. Juga kepada Anggota Tim Penguji, Bapak **Dr. Muhammad Hasbi, M.Sc.** dan Bapak **Andi Muh. Amil Siddik, S.Si., M.Si.**, terima kasih telah memberikan kritikan yang membangun dalam penyempurnaan penyusunan tugas serta waktu yang telah diberikan kepada penulis.

Penulisan skripsi ini dapat terselesaikan berkat bantuan dan motivasi dari berbagai pihak. Oleh karena itu, penulis menyampaikan ucapan terima kasih dan penghargaan yang tulus kepada:

1. **Prof. Dr. Dwia Aries Tina Pulubuhu, M.A.,** selaku Rektor Universitas Hasanuddin.
2. **Dr. Eng. Amiruddin, S.Si., M.Si.,** selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, **Dr. Nurdin, S.Si., M.Si.** sebagai Ketua Departemen Matematika FMIPA Unhas, dan almarhum **Dr. Diaraya, M.Ak.** sebagai Ketua Program Studi Ilmu Komputer Unhas.
3. Keluarga besar rumah biru, **GMKI Cabang Makassar Komisariat FMIPA Unhas** dan teman-teman **MIPA Kristen 2016**.
4. Teman-teman seperjuangan **Program Studi Ilmu Komputer 2016** yang telah mendukung dan berjuang bersama-sama dalam suka dan duka.
5. Kakak-kakak dan adik-adik **Program Studi Ilmu Komputer 2014, 2015, 2017, 2018, 2019** dan **2020**.
6. Serta segala pihak yang tidak dapat disebutkan satu persatu, yang telah membantu penulis dalam penyelesaian skripsi ini.

Akhirnya, penulis menyadari bahwa skripsi ini masih memiliki banyak kekurangan. Oleh karena itu, saran dan kritik yang membangun sangat diharapkan dari semua pihak. Harapan penulis skripsi ini dapat bermanfaat bagi setiap orang yang membacanya. Tuhan Yesus memberkati kita semua, Amin.

Makassar, 06 Oktober 2020

Deo Valiandro. M

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama	: Deo Valiandro. M
NIM	: H13116306
Programa Studi	: Ilmu Komputer
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Prediktor Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas tugas akhir saya yang berjudul:

"Pengamanan Jawaban Ujian *Computer Based Test* (CBT) dengan Menggunakan Algoritma RSA dan Fungsi HMAC Berbasis Algoritma SHA-1"

beserta perangkat yang ada (jika diperlukan). Terkait dengan hal diatas, maka pihak universitas berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar pada 06 Oktober 2020

Yang menyatakan,

iandro. M)

ABSTRAK

Computer based test (CBT) sifatnya rentang terhadap serangan, misalnya jawaban ujian yang dimanipulasi atau jawaban peserta yang dapat dilihat oleh orang yang tidak berwenang. Sehingga, diperlukan suatu mekanisme pengamanan untuk mencegah serangan-serangan tersebut. Salah satu cara untuk mengamankan hasil jawaban ujian CBT adalah dengan mengenkripsi jawaban tersebut dengan menggunakan algoritma RSA, dan fungsi HMAC berbasis algoritma SHA-1 untuk membuat suatu sidik jari/*digest* sebagai jaminan keaslian jawaban yang dikirim. Pesan dihitung nilai hashnya lalu dienkripsi dengan kunci publik RSA. Pesan terenkripsi tersebut dikirim ke server pusat pelaksana ujian, pesan yang telah sampai ke penerima akan didekripsi dengan kunci privat dan pesan hasil dekripsi dihitung nilai hashnya. Jawaban yang dienkripsi akan memiliki tingkat keamanan yang lebih tinggi, namun akan membuat ukuran pesan menjadi beberapa kali lebih besar. Keamanan pesan akan bergantung pada panjang kunci yang digunakan pada kunci RSA dan variasi karakter pada kunci HMAC.

Kata kunci: CBT, algoritma RSA, fungsi HMAC, algoritma SHA-1

ABSTRACT

Computer based tests (CBT) are vulnerable to attack, for example manipulated test answers or participant answers that can be seen by unauthorized persons. So, we need a security mechanism to prevent these attacks. One way to secure CBT exam answers is to encrypt the answers using the RSA algorithm, and the HMAC function using the SHA-1 algorithm to create a fingerprint/digest as a guarantee of the authenticity of the answers sent. The message is hashed and encrypted with the RSA public key. The encrypted message is sent to the server of the examiner, the message that has reached the recipient will be decrypted with the private key and the decrypted message will have the hash value calculated. Encrypted answers will have a higher level of security, but will make the message several times larger. Message security will depend on the length of the key used in the RSA key and the variety of characters in the HMAC key.

Keywords: CBT, RSA algorithm, HMAC function, SHA-1 algorithm

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi yang semakin hari semakin berkembang dengan sangat cepat. Perkembangan ini berdampak pada informasi yang dapat diakses oleh siapa saja. Salah satu dari hasil perkembangan teknologi adalah internet. Internet sudah menjadi media penyebaran informasi bagi orang banyak. Pengiriman dan penggunaan informasi (*sharing*) seperti data pribadi, gambar, video dan lainnya yang bersifat pribadi bisa dengan cepat dan mudah untuk diakses oleh orang lain (Zhang & Jin, 2012).

Ada banyak aspek yang memanfaatkan perkembangan internet yang sangat cepat, salah satunya pada dunia pendidikan. Mulai dari pembelajaran berbasis internet, pencarian bahan ajar menggunakan internet hingga ujian tes berbasis komputer, baik itu pada ujian akhir nasional bagi siswa Sekolah Menengah Atas maupun pada ujian masuk perguruan tinggi resmi. Ujian berbasis CBT (*Computer Based Test*) sudah mulai digunakan di Indonesia sejak ujian nasional tahun 2015 yang dilaksanakan oleh Sekolah Menengah Atas dan Seleksi Bersama perguruan Tinggi Negeri atau disingkat SBMPTN sejak tahun 2016.

Ujian berbasis komputer yang menggunakan internet sebagai media memiliki sisi positif maupun negatif. Sisi positifnya pada transparansi, akurasi, dan keandalan serta pelaksanaan ujian yang efektif dan efisiensi karena distribusi soal-soal dan pengiriman hasil jawaban cepat. Namun sisi negatifnya, kerentanan keamanan pada hasil ujian yang bisa di manipulasi oleh penyadap yang ingin mengganti jawaban ujian kemudian. Masalah yang lain adalah jawaban yang dapat dilihat oleh penyadap sebelum jawaban tersebut dikirim atau sementara ujian tersebut dikirim ke pusat pelaksana ...n sehingga memungkinkan terjadinya penyontekan jawaban ujian. Oleh ...ena itu perlu penanganan khusus untuk mengamankan sistem ujian

berbasis komputer, khususnya dalam pengiriman hasil ujian ke pusat server penyelenggara ujian.

Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikan ke dalam bentuk yang tidak dapat dipahami lagi maknanya (Meyer, 1982). Kriptografi bertujuan untuk memberikan keamanan berupa kerahasiaan, integritas data, otentikasi dan anti-penyangkalan (Schneier, 1996). Kriptografi bekerja dengan cara mengubah suatu pesan, baik itu berupa teks, gambar, video, ataupun media lainnya menjadi bentuk yang lain dengan menggunakan algoritma tertentu yang kemudian disebut sebagai proses enkripsi. Untuk membaca pesan ini, diperlukan suatu proses untuk mengubah kembali pesan yang sudah terenkripsi menjadi bentuk awal, proses ini disebut dengan dekripsi. Dalam kriptografi dikenal dua jenis algoritma berdasarkan kunci yang digunakan, algoritma simetris yaitu algoritma yang menggunakan kunci yang sama untuk enkripsi dan dekripsi pesan, dan kriptografi non-simetri atau yang dikenal pula dengan istilah kriptografi kunci publik yaitu algoritma kriptografi yang menggunakan dua jenis kunci, kunci publik untuk mengenkripsi pesan dan kunci privat untuk mendekripsi pesan.

Salah satu fungsi dari kriptografi adalah integritas data yang berfungsi untuk menjamin data keaslian suatu pesan selama pengiriman informasi tersebut melalui media tertentu (Schneier, 1996). Integritas data dalam kriptografi dinyatakan dalam fungsi *hash* satu arah. Fungsi *hash* satu arah bekerja dengan membuat suatu tanda tangan digital yang sifatnya tidak sama untuk setiap pesan (Kromodimoeljo, 2010). Hasil dari fungsi *hash* pada pesan adalah berupa pesan pendek (*message digest*) yang memiliki ukuran tetap. Sehingga, setiap perubahan dalam suatu pesan baik itu pengurangan, penambahan ataupun perubahan akan dapat diketahui oleh pihak penerima . Salah satu algoritma fungsi *hash* adalah *secure hash algorithm* (*SHA*) yang miliki banyak varian dengan parameter berbeda-beda. Algoritma SHA-1 lah salah satu varian dalam keluarga SHA yang paling banyak digunakan ena tingkat keamanannya.

Fungsi *hash* memiliki kelemahan pada bagaimana mendeteksi perubahan pesan sekaligus perubahan pada nilai *hash*. Apabila penyadap memiliki akses terhadap file atau dokumen nilai *hash* dari pesan, maka penyadap dapat mengubah nilai pesan kemudian menghitung ulang nilai *hash* pesan yang sudah diubah tersebut dan mengubahnya pada file nilai *hash* semula. Hal ini dapat diatasi dengan menambahkan sejumlah kunci rahasia pada pesan sebelum nilai *hash*-nya dihitung sehingga penyadap tidak akan bisa menghitung nilai *hash* yang baru ketika tidak mengetahui kunci rahasia yang ditambahkan pada pesan yang ada.

Algoritma RSA merupakan salah satu algoritma kriptografi kunci publik yang populer. Algoritma ini dibuat oleh 3 orang peneliti dari MIT (*Massachussets Institute of Technology*) pada tahun 1976. Algoritma RSA dapat digunakan untuk enkripsi dan dekripsi berbagai jenis data, misalnya teks, gambar, suara dan video. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima (Rivest, dkk., 1977).

Untuk mengatasi masalah keamanan pada sistem ujian nasional berbasis komputer, maka digunakan algoritma SHA-1 untuk memverifikasi keaslian pesan

1.2. Rumusan Masalah

Berdasarkan latar belakang di atas, dapat dirumuskan beberapa masalah berikut:

1. Bagaimana cara implementasi fungsi HMAC dengan menggunakan algoritma SHA-1 pada hasil ujian CBT untuk mencari *message digest*-nya?
2. Bagaimana cara implementasi enkripsi algoritma RSA pada hasil ujian CBT?
3. Bagaimana cara mendekripsi hasil ujian CBT yang telah di enkripsi dan memverifikasi keaslian hasil ujian CBT?
4. Bagaimana analisis performa dari enkripsi dan dekripsi hasil ujian CBT tersebut?

1.3. Tujuan Penelitian

Dengan memperhatikan latar belakang dan rumusan masalah di atas, maka tujuan penelitian ini adalah:

1. Mampu mengimplementasikan fungsi HMAC dengan menggunakan algoritma SHA-1 pada hasil ujian CBT,
2. Mampu mengimplementasikan enkripsi algoritma RSA pada hasil ujian CBT.
3. Mampu mendekripsi dan memverifikasi keaslian hasil ujian CBT.
4. Mampu untuk menganalisis performa dari enkripsi dan dekripsi hasil ujian CBT.

1.4. Manfaat Penelitian

Hasil penelitian ini diharapkan dapat dimanfaatkan dalam pengamanan hasil ujian yang berbasis komputer (*computer based test*) baik dalam pengamanan jawaban, maupun dalam verifikasi keaslian jawaban yang diberikan oleh pihak yang mengikuti ujian.

1.5. Batasan Masalah

Adapun batasan masalah yang dilakukan untuk mencegah pembahan yang terlalu luas, maka masalah yang dibahas dibatasi pada:

1. Pengamanan ujian CBT menitik beratkan pada pengamanan pengiriman hasil jawaban peserta,
2. Sistem ujian yang digunakan adalah ujian pilihan ganda sebanyak 100 soal,
3. Jawaban yang dikirim dikodekan menggunakan sistem bilangan heksadesimal (basis 16),
4. Soal yang digunakan menggunakan lima pilihan ganda yaitu A, B, C, D dan E, di mana salah satu pilihan tersebut adalah pilihan yang benar,
5. Jika peserta ujian tidak memberikan jawaban, maka diberikan tanda tertentu yang mengindikasikan tidak ada jawaban, tanda tersebut adalah huruf F.

BAB II
TINJAUAN PUSTAKA

2.1. Ujian

Ujian adalah cara untuk mengukur kemampuan seseorang. Pelaksanaan ujian bertujuan untuk mengukur pengetahuan seseorang atau dalam hal ini peserta didik, tujuan ini misalnya pada pelaksanaan ujian nasional di jenjang pendidikan SD, SMP/MTs, SMPLB, SMA/MA/SMAK/SMTK, SMALB, maupun di SMK/MAK dan juga sebagai salah satu tolak ukur pencapaian pembelajaran dalam rangka penjaminan dan peningkatan mutu pendidikan (Badan Standar Nasional Pendidikan (BSNP) Dan Badan Penelitian Dan Pengembangan, Kementerian Pendidikan Dan Kebudayaan (Balitbang Kemdikbud), 2018). Selain itu, pelaksanaan ujian juga digunakan untuk menyeleksi para calon untuk memasuki suatu institut tertentu, hal ini terlihat pada ujian masuk perguruan tinggi yang dilaksanakan baik oleh perguruan tinggi negeri maupun oleh perguruan tinggi swasta (Menteri Pendidikan dan Kebudayaan Republik Indonesia, 2018).

Sebagai bentuk dari tes kemampuan seorang, maka setiap orang, kelompok atau lembaga pelaksana ujian menggunakan prinsip-prinsip kejujuran, kerahasiaan, keamanan dan kelancaran (Menteri Pendidikan dan Kebudayaan Republik Indonesia, 2018). Ujian diimplementasikan menjadi berbagai bentuk ujian, misalnya ujian tertulis, ujian fisik, ujian tes psikologi maupun ujian wawancara.

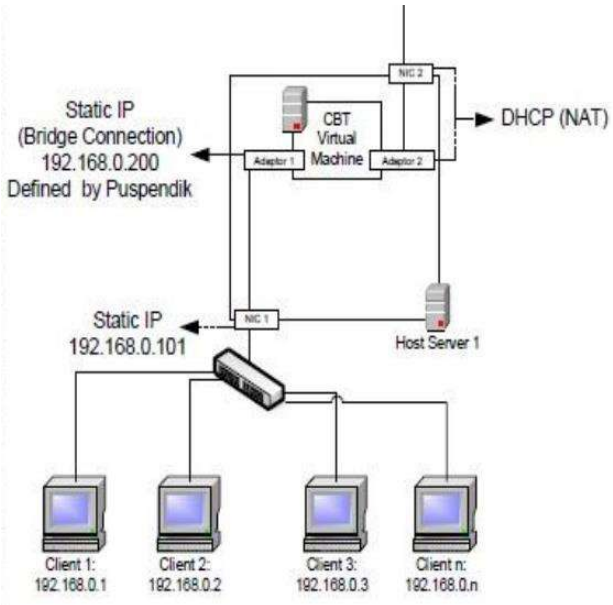
2.2. Ujian Computer Based Test (CBT)

Ujian berbasis komputer (*Computer Based Test*, CBT) adalah sistem ujian yang digunakan dengan menggunakan sistem komputer. Ujian berbasis komputer dilaksanakan untuk mengatasi kelemahan ujian berbasis kertas. Kelemahan ini seperti bentuk soal yang digunakan pada saat ujian sulit uk dibuat bervariasi, tampilan soal terbatas, hanya dua dimensi, erlukan banyak kertas dan biaya penggandaan yang cukup besar;

pengamanan kerahasiaan soal relatif sulit dan memerlukan biaya cukup besar; pengolahan hasil memerlukan waktu yang relatif lama.

Ujian CBT bekerja dengan cara menggunakan klien dan server. Ujian CBT bisa menggunakan 3 jenis sistem yaitu (Kementrian Pendidikan dan Kebudayaan, 2020):

1. Sistem *online* atau terhubung langsung dengan server pusat secara *realtime*, soal langsung dari server pusat dan hasil pengerjaan soal tersebut juga langsung masuk ke server pusat,
2. Sistem semi-*online* yaitu soal dikirim dari server pusat secara online melalui jaringan (sinkronisasi) ke server lokal (sekolah/pusat ujian), kemudian ujian siswa dilayani oleh server lokal (sekolah/pusat ujian) secara *offline*. Selanjutnya hasil ujian dikirim kembali dari server lokal (sekolah/pusat ujian) ke server pusat secara *online* (melalui proses *upload*). Sistem ini seperti ditunjukkan dalam Gambar 2.1.
3. Sistem *offline*, yaitu ujian dilayani oleh server lokal, soal dan hasil pengerjaan soal dikirim dengan menggunakan media penyimpanan.



Gambar 2.1. Skema Ujian CBT semi-offline

Implementasi ujian CBT di Indonesia dapat ditemukan pada ujian nasional (UN) dan ujian seleksi bersama masuk perguruan tinggi negeri (SBMPTN) yang masing-masing menggunakan sistem ujian semi-*online*. Ujian nasional berbasis komputer adalah kegiatan pengukuran dan penilaian pencapaian standar kompetensi lulusan SMP/MTs, SMPLB, SMA/MA/SMAK/SMTK, SMALB, SMK/MAK secara nasional meliputi mata pelajaran tertentu yang menggunakan komputer sebagai media untuk menampilkan soal dan proses menjawabnya (Badan Standar Nasional Pendidikan, 2019).

2.3. Kriptografi

Kriptografi berasal dari Bahasa Yunani, yaitu "*cryptós*" yang berarti rahasia, sedangkan "*gráphein*" artinya tulisan. Jadi, kriptografi dapat diartikan sebagai tulisan rahasia (Munir, 2019). Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, serta otentikasi (Menezes, dkk., 1997).

Kriptografi memegang peranan penting dalam kehidupan sehari-hari pada saat ini. Berbagai informasi yang sifatnya rahasia, sensitif atau bernilai tinggi seperti *password* untuk ATM, data rekam jejak transaksi bank, akses perangkat komputer dan telepon genggam ke internet hingga percakapan melalui sosial media seperti *WhatsApp* dan *Telegram* pun menggunakan kriptografi sebagai pengamanannya. Penggunaan kriptografi pada media-media ini bertujuan untuk mencegah penyadapan pada data-data penting yang ada di dalamnya.

Ada banyak kasus pembobolan informasi sensitif, penyadapan percakapan dan pencurian dokumen-dokumen penting yang terjadi pada masa sekarang ini. Contoh nyata kasus-kasus ini misalnya kasus bocornya 91 juta data pengguna Tokopedia yang baru-baru ini terjadi dan kasus 500 ribu akun gguna Zoom yang dijual di *darkweb*. Dari banyaknya kasus yang terjadi nudian memberikan pesan moral bahwa keamanan informasi adalah salah u kebutuhan utama yang perlu pada saat ini.

Kriptografi memiliki tujuan memberikan keamanan yang terbagi menjadi beberapa aspek sebagai berikut (Schneier, 1996):

1. Kerahasiaan (*confidentiality*), yaitu menjaga keamanan informasi sehingga tidak dapat diketahui oleh siapa pun kecuali pihak yang memiliki otoritas untuk mengetahui pesan tersebut. Di dalam kriptografi, aspek ini direalisasikan dengan enkripsi dan dekripsi informasi yang bersifat rahasia.
2. Integritas data (*data integrity*), yaitu penjaminan informasi masih asli atau tidak diubah oleh pihak lain selama pengiriman informasi tersebut. Di dalam kriptografi, aspek ini direalisasikan dengan menggunakan fungsi *hash* dan tanda tangan digital (*digital signature*).
3. Otentikasi (*authentication*), yaitu identifikasi kebenaran pihak-pihak yang saling berkomunikasi mempertukarkan informasi bahwa pesan yang diterima atau dikirim benar dari pengirim atau penerima yang sesungguhnya. Di dalam kriptografi, aspek ini direalisasikan dengan menggunakan tanda tangan digital (*digital signature*).
4. Anti-penyangkalan (*non-repudiation*), yaitu pencegahan pengirim informasi menyangkal bahwa pengirim yang telah mengirim informasi. Di dalam kriptografi, aspek ini direalisasikan dengan menggunakan tanda tangan digital (*digital signature*).

Dalam kriptografi dikenal beberapa istilah yang digunakan. Berikut ini dijelaskan istilah-istilah tersebut:

1. Pesan, plainteks dan cipherteks. Pesan (*message*) adalah data atau informasi yang dapat dibaca, dipersepsi dan dimengerti artinya, pesan ini dapat berbentuk teks, citra (*image*), suara (*audio*), video atau bentuk-bentuk biner lainnya. Plainteks (*plaintext*) adalah pesan yang berupa teks yang masih asli. Cipherteks (*cypertext*) adalah pesan yang sudah disandikan.

Pengirim dan penerima. Pengirim (*sender*) adalah pihak yang mengirim pesan kepada pihak lainnya sedangkan penerima (*receiver*) adalah pihak yang menerima pesan dari pengirim pesan. Pengirim dan

penerima tidak harus berupa manusia, namun juga dapat berupa mesin, robot atau komputer.

3. Enkripsi dan dekripsi. Enkripsi (*encryption, enchiphering*) adalah proses menyandikan plainteks menjadi cipherteks. Sedangkan proses mengubah cipherteks menjadi plainteks disebut dekripsi (*decryption, deciphering*).
4. Cipher dan kunci. Cipher adalah algoritma kriptografi untuk enkripsi dan dekripsi yang berupa aturan berupa fungsi matematis untuk *enchipering* dan *dechipering*. Sedangkan kunci adalah parameter yang digunakan untuk enkripsi dan dekripsi palinteks.
5. Sistem kriptografi. Sistem kriptografi adalah himpunan yang terdiri dari algoritma enkripsi, algoritma dekripsi, kunci, plainteks dan cipherteks (Schneier, 1996).
6. Penyadap. Penyadap (*eavesdropper*) adalah pihak yang mencoba menangkap secara tidak sah pesan yang dikirim.
7. Kriptanalisis dan kriptologi. Kriptanalisis (*cryptanalysis*) adalah ilmu dan seni untuk memecahkan cipherteks menjadi plainteks tanpa mengetahui kunci yang digunakan selama proses enkripsi maupun dekripsi pesan. Sedangkan kriptologi adalah studi mengenai kriptografi dan kriptanalisis.

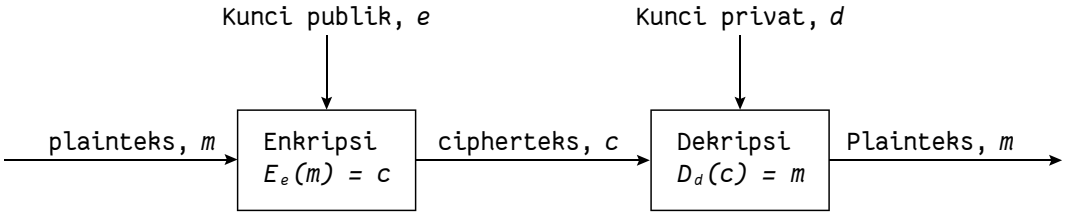
Cipher atau algoritma tidak perlu rahasia, namun pada kriptografi klasis, kekuatan algoritma kriptografi terletak pada kerahasiaan algoritmanya. Ketika sebuah algoritma kriptografi diketahui oleh pihak lain, maka keamanannya sudah tidak dijamin lagi. Sedangkan pada kriptografi modern, keamanan kriptografi terletak pada kunci (*key*) yang digunakan untuk enkripsi dan dekripsi. Algoritma kriptografi sudah tidak lagi rahasia, namun kunci untuk dekripsi pesan akan menjadi rahasia. Hal ini sesuai dengan prinsip Kerckhoff yang mengatakan “hanya kerahasiaan kuncilah yang menjamin keamanan” (Kerchoffs, 1883).

2.4. Kriptografi Kunci Publik/ Nirsimetri

Sampai tahun 1975, kriptografi kunci simetris masih menjadi satu-satunya jenis kriptografi yang ada. Algoritma kriptografi simetris adalah algoritma kriptografi yang menggunakan kunci yang sama antara enkripsi pesan dan dekripsinya. Hal ini membuat algoritma simetris memiliki kelemahan yaitu bagaimana menjaga kerahasiaan kunci (sehingga kunci disebut dengan *secret key*) (Munir, 2019). Masalah lain adalah bagaimana membagikan kunci rahasia ke pihak yang akan melakukan dekripsi pesan. Mengirim kunci bersama dengan pesan jelas tidak aman, sehingga dibutuhkan saluran lain untuk mengirimkan kunci rahasia.

Kekurangan algoritma kunci simetris kemudian dipecahkan oleh Diffie dan Hellman pada tahun 1976 yang mempublikasikan Algoritma Diffie-Hellman yang menjadi peletak dasar kriptografi kunci publik. Konsep kriptografi kunci-nirsimetri atau sering disebut dengan algoritma kriptografi kunci publik menawarkan pendekatan yang berbeda yang menghilangkan kebutuhan untuk pendistribusian kunci (Deffie & Hellman, 1976). Kriptografi kunci publik disebut demikian karena terdapat dua kunci yang digunakan, satu untuk enkripsi dan satu untuk dekripsi (Aumasson, 2018).

Enkripsi dan dekripsi dalam kriptografi kunci publik menggunakan sepasang kunci yaitu kunci publik (*public key*) yang diberi simbol *e* dan kunci privat (*private key*) yang diberi simbol *d*. Pesan yang telah dienkripsi menggunakan kunci privat hanya dapat di dekripsi menggunakan kunci publik dan pesan yang dapat di dekripsi menggunakan kunci publik dapat dipastikan telah di enkripsi menggunakan kunci privat. Sebaliknya, naskah yang telah di enkripsi menggunakan kunci publik hanya dapat di dekripsi menggunakan kunci privat. Mekanisme ini memungkinkan berbagai aplikasi, dua yang terpenting di antaranya adalah tidak perlunya distribusi kunci dan tanda tangan digital (*digital signature*) (Kromodimoeljo, 2010). Karena kunci ripsi dan kunci dekripsi tidak sama, maka kriptografi kunci publik disebut a kriptografi nirsimetris. Skema kriptografi kunci publik dapat dilihat pada mbar 2.2. berikut.



Gambar 2.2. Skema kriptografi kunci-publik

Seperti terlihat dalam Gambar 2.2., jika diberikan fungsi enkripsi *E*, fungsi dekripsi *D* dan (*e*, *d*) adalah sepasang kunci publik dan kunci privat, maka enkripsi pesan *m* dengan menggunakan kunci publik *e* menghasilkan cipherteks *c* dinyatakan sebagai:

$$E_e(m) = c \tag{2.1}$$

dan dekripsi cipherteks *c* dengan menggunakan kunci privat *d* dinyatakan sebagai:

$$D_d(c) = m \tag{2.2}$$

Kedua persamaan ini menyatakan bahwa walaupun dengan mengetahui *e* dan *c*, maka secara komputasi hampir tidak mungkin menemukan *m*. Asumsi lainnya, dengan mengetahui *e*, secara komputasi hampir tidak mungkin menurunkan *d* dari *e*.

2.5. Fungsi Hash-Satu Arah

Fungsi hash adalah fungsi yang menerima masukan *string* yang panjangnya sembarang kemudian memampatkannya menjadi *string* luaran yang panjangnya tetap (*fixed-lenght output*), yang umumnya lebih kecil daripada ukuran *string* semula (Houtven, 2013). Ukuran *string* hasil tidak terlalu besar, antara 128 sampai 512 bit tergantung algoritma yang digunakan, sedangkan panjang naskah atau pesan tidak terbatas. Jika *string M* menyatakan pesan (*message*), maka sembarangan pesan *M* berukuran bebas dikompresi oleh fungsi hash *H* dengan:

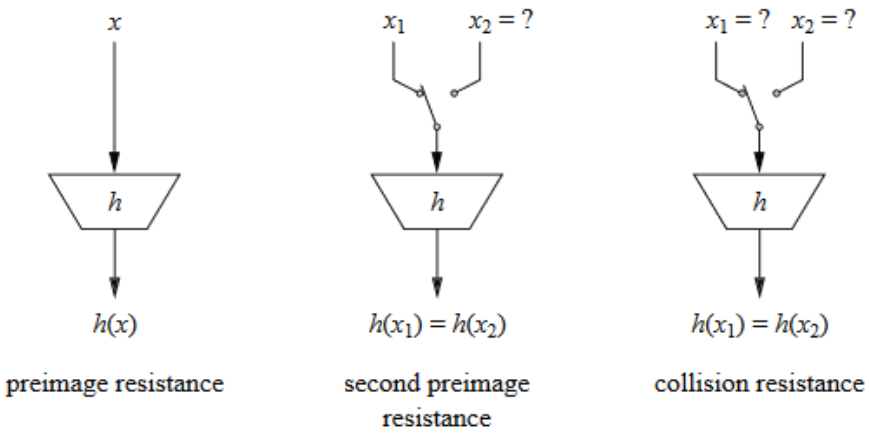
$$h = H(M) \tag{2.3}$$

Proses hash ini adalah proses pembuatan suatu "sidik jari" (*fingerprint* u sering disebut *diggest*) untuk suatu pesan atau naskah. Hasil fungsi *hash*

sering disebut dengan nilai *hash* (*hash value*) atau pesan ringkas (*message digest*). Fungsi hash seperti disamakan dengan *Swiss Army Knife* yang memiliki banyak manfaat: digunakan dalam tanda tangan digital (*digital signature*), kriptografi kunci publik, verifikasi integritas data, otentikasi pesan, perlindungan terhadap password, protokol *key aggrement*.

Fungsi *hash* satu arah (*one-way hash*) adalah fungsi *hash* yang bekerja dalam satu arah. Artinya, pesan dapat dikompresi menjadi *message digest*, tetapi *message digest* tersebut tidak dapat dikembalikan lagi menjadi pesan semula. Nilai hash untuk dua buah pesan akan berbeda, bahkan untuk perbedaan satu bit saja. Ada beberapa kriteria fungsi *hash* satu-arah sebagai standar keamanan fungsi hash, sebagai berikut dan digambarkan dalam Gambar 2.3. (Paar & Pelzl, 2010):

1. *Preimage resistance* atau sifat satu arah, yaitu untuk setiap h yang diberikan, tidak mungkin menemukan sedemikian sehingga $H(x) = h$,
2. *Second preimage resistance*, yaitu untuk setiap x, tidak mungkin mencari $y \neq x$ sedemikian sehingga $H(y) = H(x)$, kriteria ini kerap disebut juga *weak collision resistance*,
3. *Collision resistance*, yaitu tidak mungkin (secara komputasi) mencari pasangan x dan y sedemikian sehingga $H(x) = H(y)$, persyaratan ini kerap disebut juga *strong collision resistance*.



Gambar 2.3. Tiga kriteria utama fungsi hash

2.6. Message Authentication Code (MAC)

Message authentication code (*MAC*) adalah fungsi *hash* satu-arah yang menggunakan kunci rahasia (*secret key*) dalam pembuatan nilai *hash*. Sehingga nilai *hash* suatu adalah pesan itu sendiri dan kunci. *MAC* memiliki sifat yang sama dengan sifat-sifat *hash* kecuali dengan tambahan adanya komponen kunci yang digunakan untuk memverifikasi nilai *hash*. *MAC* disebut juga *keyed hash function* atau *key-dependent one-way hash function* (Munir, 2019).

MAC berguna untuk mengatasi kelemahan dari fungsi *hash*. Kelemahan fungsi *hash* terletak pada bagaimana mendeteksi perubahan pada pesan sekaligus perubahan pada nilai *hash*-nya. Misalnya terdapat sebuah pesan atau dokumen yang sudah dihitung nilai *hash*-nya. Nilai hash dari dokumen tersebut di simpan di dalam sebuah file. Jika penyadap dapat mengakses file tersebut dan kemudian mengubah dokumen lalu mengubah nilai hash yang ada pada file tersebut, maka penerima dokumen atau pesan tersebut tidak dapat memastikan dokumen tersebut masih asli atau sudah diubah. *MAC* dapat dinyatakan sebagai:

$$MAC = H_K(M)$$

dengan *MAC* adalah nilai *hash*, *H* adalah fungsi *hash* dan *K* adalah kunci rahasia. Fungsi *MAC* memampatkan pesan *M* yang telah diberikan kunci *K* sehingga walaupun pesan *M* diketahui oleh pihak penyerang, namun pihak penyerang tidak dapat membuat nilai *hash* dari *M* jika tidak mengetahui kunci *K* yang digunakan. Sehingga keamanan dari *MAC* terletak pada keamanan kunci *K*.

2.7. Keyed-hash Message Authentication Code (HMAC)

Keyed-hash message authentication code atau HMAC adalah salah satu mekanisme *message authentication code* (*MAC*) yang dibuat oleh Mihir lare, Ran Canetti dan Hugo Krawczyk pada tahun 1996. HMAC rupakan salah satu algoritma MAC yang berbasis fungsi *hash* satu arah. IAC bekerja dengan cara melakukan fungsi hash terhadap sebuah pesan

kemudian mengenkripsi pesan tersebut dengan sebuah kunci. Hal ini memungkinkan untuk mencegah serangan terhadap nilai hash dengan menggunakan serangan *dictionary attack*. Algoritma HMAC diberikan sebagai berikut (Bellare, Canetti, & Krawczyk, 1997) :

$$HMAC(K, m) = H((K \oplus opad)||((K \oplus ipad)||m))$$

dengan *K* adalah *key* atau kunci yang diketahui oleh pengirim dan penerima, *H* adalah fungsi *hash*, *m* adalah pesan, sementara *ipad* dan *opad* adalah byte-*byte padding* (agar ukurannya bloknya sama jika kurang dari panjang tertentu). Nilai *opad* adalah 0x5c5c5c5 ... c5 dan *ipad* adalah 0x363636 ... 36 (Bellare, Canetti, & Krawczyk, 1997). Operasi || adalah operasi penggabungan (*concatenation*).

Dari algoritma di atas, jika diasumsikan *K* adalah kunci, *B* adalah ukuran blok (dalam *byte*) yang merupakan input dalam fungsi *hash*, K_0 adalah kunci yang telah diproses sehingga panjangnya sama dengan *B*, *L* adalah ukuran blok (dalam *byte*) dari output fungsi *hash*, *opad* dan *ipad* adalah dua buah *byte padding* dengan nilai *0x5c5c5c5* dan *0x363636*, dan *m* adalah pesan, maka algoritma *HMAC* dapat dijabarkan ke dalam langkah-langkah sebagai berikut :

Step 1. Jika panjang *K* = *B*, set K_0 = *K*. Kemudian langsung ke *step 4*,

Step 2. Jika panjang *K* > *B*, lakukan fungsi *hash* terhadap *K* untuk mendapatkan *L* string byte, kemudian *append* dengan angka 0 dengan jumlah (*B* – L) untuk mendapatkan string K_0 yang panjangnya sama dengan *B*. Kemudian lanjutkan ke *step 4*.

Step 3. Jika panjang *K* < *B*, *append* angka 0 dengan jumlah (*B* – *L*) untuk mendapatkan *string byte* K_0 yang panjangnya sama dengan *B*. Kemudian lanjutkan ke *step* 4.

Step 4. Lakukan *XOR* antara K_0 dengan *ipad*, yang menghasilkan *byte* panjang B.

Step 5. *Append string m* dengan *string* hasil dari *step 4*.

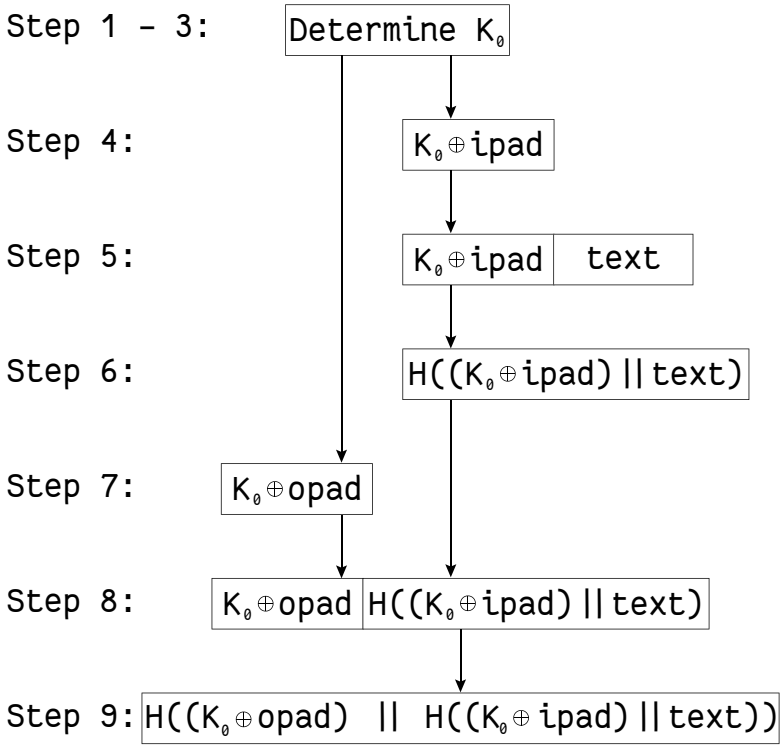
Step 6. Operasikan fungsi *hash* untuk *string* hasil dari *step 5*.

Step 7. Lakukan *XOR* antara K_0 dengan *opad*.

Step 8. Lakukan *append* antara *string* hasil *step 6* dengan *string* hasil dari *step 7*.

Step 9. Lakukan operasi fungsi *hash* terhadap *string* hasil dari *step 8*.

Algoritma di atas dapat digambarkan seperti diagram seperti terlihat dalam Gambar 2.4. berikut ini (Information Technology Laboratory, 2008) :



Gambar 2.4. Konstruksi HMAC

2.8. Secure Hash Algorithm (SHA)

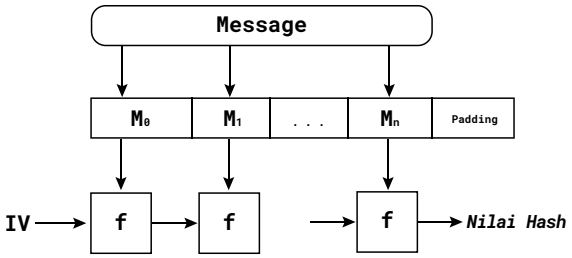
SHA (*Secure Hash Algorithm*) adalah fungsi *hash* satu-arah yang dirancang oleh National Security Agency (NSA) dan dijadikan standar oleh National Institute of Standards and Technology, disingkat NIST (Badan Nasional Standar dan Teknologi Amerika Serikat) (NIST, 2004) dan unakan bersama *DSS* (*Digital Signature Standard*). SHA adalah anjutan dari pendahulunya, MD5 yang telah digunakan secara luas namun emukan kolisinya.

SHA merupakan keluarga fungsi *hash* satu-arah dengan enam varian dengan perbedaan pada parameter yang digunakan. Fungsi SHA-1 merupakan variasi yang paling banyak digunakan dan diimplementasikan ke dalam berbagai aplikasi dan protokol keamanan seperti *TLS*, SSL, PGP, SSH, S/MIME dan Ipsec. Variasi pertama yang dipublikasikan adalah SHA-0 pada 1993 dan sudah ditemukan kolisinya. SHA-1 dipublikasikan pada tahun 1995. Empat variasi lain kemudian dipublikasikan yang dikenal dengan SHA-2 yaitu SHA-224, SHA-256, SHA-384 dan SHA-512. NIST kemudian mengumumkan fungsi *hash Keccak* sebagai SHA-3 pada tahun 2012 (Paar & Pelzl, 2010).

Algoritma SHA bekerja dengan membagi naskah atau pesan menjadi beberapa blok, setiap blok biasanya 512 atau 1024 bit. Naskah atau pesan diberi *padding* agar panjang pesan merupakan kelipatan dari besarnya blok dan *padding* diberi akhiran berupa panjang dari pesan. Algoritma biasanya terdiri dari dua tahap, yaitu: 1) *preprocessing data*, yaitu terdiri dari *padding* dan *parameter setup*, 2) *hashing*, yaitu tahap membuat *message digest* dengan mengompresi data. Kompresi dilakukan dengan berurutan tiap blok dan hasil blok sebelumnya dijadikan *feedback* untuk blok berikutnya. Cara kerja ini bekerja secara iteratif. Konstruksi seperti ini dinamakan konstruksi Merkle–Damgård (Merkle, 1979). Misalnya untuk fungsi *hash* adalah blok pesan (*M*) dan luaran dari *hasing* blok sebelumnya (h_{i-1}), maka dapat dinyatakan sebagai berikut:

$$h_i = H(M_i, h_{i-1}) \tag{2.4}$$

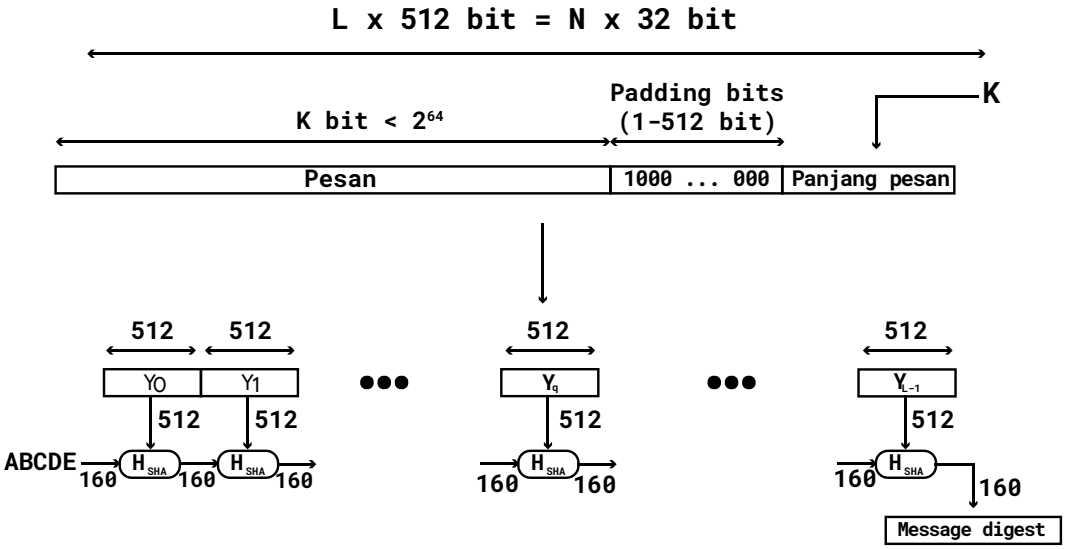
Konstruksi Merkle–Damgård diperlihatkan dalam Gambar 2.5. sebagai berikut ini. Terlihat bahwa operasi dari M_0 menggunakan *initialization vector* (IV) sebagai nilai awal. M_0 dioperasikan pada fungsi *f* dan hasilnya digunakan sebagai parameter pada operasi M_{0+1}.



Gambar 2.5. Konstruksi Merkle-Damgard

2.9. SHA-1

SHA-1 merupakan salah satu varian dari keluarga *Secure Hash Algorithm* (SHA) yang dipublikasikan pada tahun 1995. SHA-1 menerima masukan berupa pesan dengan ukuran maksimum 2^{64} bit (2.147.483.648 *gigabyte*) dan menghasilkan *message digest* sebesar 160 bit. SHA-1 menggunakan konstruksi *Merkle–Damgård* seperti algoritma *hash* lainnya dalam keluarga SHA. Algoritma SHA-1 bekerja menggunakan fungsi kompresi seperti pada blok cipher, di mana input untuk blok berikutnya adalah hasil dari H_{i-1}. Operasi setiap ronde dalam algoritma SHA-1 mirip dengan cara kerja skema Feistel dalam block cipher Gambaran mengenai algoritma fungsi *hash* SHA-1 diberikan dalam Gambar 2.6. berikut.



Gambar 2.6. Pembuatan message digest algoritma SHA-1

Langkah-langkah pembuatan *message digest* dengan SHA-1 sebagai ikut (Munir, 2019):

Penambahan bit-bit pengganjal (*padding bits*),

2. Penambahan nilai panjang pesan semula,
3. Inisiasi penyangga (*buffer*) *MD*,
4. Pengolahan pesan dalam blok berukuran 512 bit.

Penjelasan secara rinci dari keempat langkah di atas dijelaskan di bawah ini.

Penambahan bit-bit pengganjal (*padding bits*)

Diasumsikan misalnya terdapat pesan dengan panjang *l*, maka ditambahkan sejumlah pengganjal sehingga panjang pesan merupakan kelipatan 512. Pesan ditambah dengan "1" diikuti sejumlah *k* angka "0" lalu 64 bit terakhir adalah panjang pesan yang direpresentasikan dengan *l*. Jumlah "0" diberikan oleh:

$$k \equiv 512 - 64 - 1 - l \qquad (2.5)$$

$$= 448 - (l + 1)\ mod\ 512$$

Angka 512 muncul karena SHA-1 memproses pesan dalam blok-blok yang berukuran 512. Jika panjang pesan adalah 448 bit, maka pesan tersebut ditambah dengan 512 bit sehingga menjadi 960 bit. Panjang bit-bit pengganjal dari 1-512. Misalnya diberikan pesan adalah "abc" yang mana terdiri dari 3 karakter 8-bit ASCII dengan panjang pesan *l* = 24 bits:

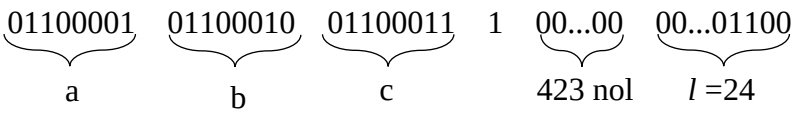
$$\underbrace{01100001}_{a} \quad \underbrace{01100010}_{b} \quad \underbrace{01100011}_{c}$$

Ditambahkan "1" diikuti dengan *k* = 423 angka "0", nilai *k* diberikan oleh persamaan:

$$k \equiv 448 - (l + 1) = 448 - 25 = 423\ mod\ 512$$

Penambahan nilai panjang semula

Kemudian ditambahkan 64-bit nilai yang merupakan representasi dari panjang pesan $l = 24_{10} = 11000_2$. Setelah ditambahkan dengan 64-bit, panjang pesan sekarang menjadi kelipatan dari 512 bit. Sehingga pesan yang telah di berikan *padding* adalah:



Inisiasi penyanggah *MD*

SHA-1 membutuhkan 5 buah penyanggah (*buffer*) dengan panjang masing-masing penyanggah adalah 32-bit, sehingga total penyanggah adalah $5x32 = 160\ bit$. Kelima penyangga ini berfungsi untuk inisiasi nilai iterasi pertama, menampung hasil sementara selama operasi tiap blok dan merupakan tempat akhir nilai *hash*. Kelima penyangga diberi nama A, B, C, D dan E. Setiap penyangga diinisiasi dengan nilai-nilai sebagai berikut (dalam notasi *HEX*):

A = 67452301_{16}

B = $EFCDAB89_{16}$

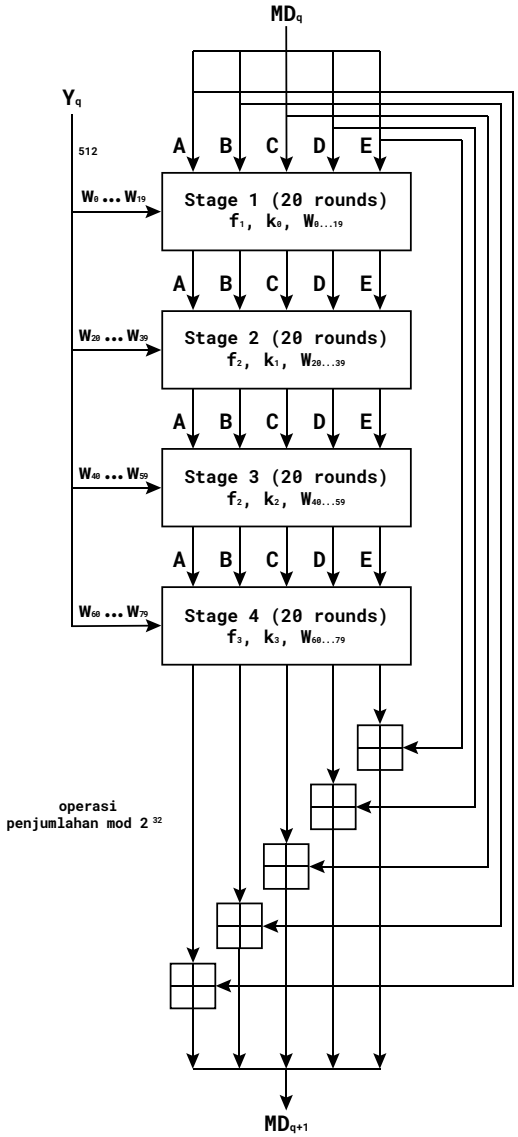
C = $98BADCFE_{16}$

D = 10325476_{16}

E = $C3D2E1F0_{16}$

Pengolahan pesan dalam blok berukuran 512 bit

Pesan dibagi menjadi *L* blok dengan ukuran masing-masing blok adalah 512 bit (Y_0 – Y_{L-1}). Setiap blok kemudian diproses dengan menggunakan penyangga *MD* sehingga menghasilkan nilai 128-bit, yang disebut dengan proses H_{SHA-1}. Pada blok pertama (Y_0) penyangga *MD* menggunakan inisiasi penyangga *MD* yang telah didefinisikan sebelumnya. Blok berikutnya menggunakan *MD* yang adalah hasil dari proses H_{SHA} sebelumnya. Proses H_{SHA} ditunjukkan pada Gambar 2.7. berikut.



Gambar 2.7. Pengolahan blok 15 bit (Proses H_{SHA})

Pada Gambar 2.7 ditunjukkan proses H_{SHA} yang mana terdiri dari 80 putaran yang dibagi menjadi 4 stage. Tiap stage terdiri dari 20 putaran dengan perbedaan pada operasi yang dijalankan dan bilangan penambahnya. Pada Gambar 6, Y_q menyatakan blok 512-bit ke-q dari pesan yang telah diberikan bit pengganjal dan tambahan 64-bit panjang pesan. Hasil dari proses H_{SHA} adalah MD_q (*message digest*) dengan panjang 160-bit. Pada Y_0 nilai MD_0 adalah nilai dari inisiasi penyangga *MD*.

Setiap putaran menggunakan operasi dasar yang sama (dinyatakan agai fungsi *f*). Operasi dasar *SHA-1* diberikan sebagai berikut:

$$a, b, c, d, e \leftarrow (CLS_5(a) + f_t(b, c, d) + e + W_t + K_t), a, CLS_{30}(b), c, d \quad (2.6)$$

Dengan keterangan:

a, *b*, *c*, *d*, *e* = lima buah penyanggah 32-bit (berisi nilai penyanggah *A*, *B*, *C*, *D*, *E*)

t = putaran, $0 \leq t \leq 79$

f_t = fungsi logika

CLS_s = *circular left shift* sebanyak *s*-bit

W_t = *word* 32-bit yang diturunkan dari blok 512-bit yang sedang diproses

K_t = konstanta penambah

\+ = operasi penjumlahan dalam modulo 2^{32}

Pada setiap putaran pada 80 putaran, masing-masing menggunakan bilangan penambah K_t yang diberikan sebagai berikut:

Untuk putaran $0 \leq t \leq 19$ $K_t = 5A827999_{16}$

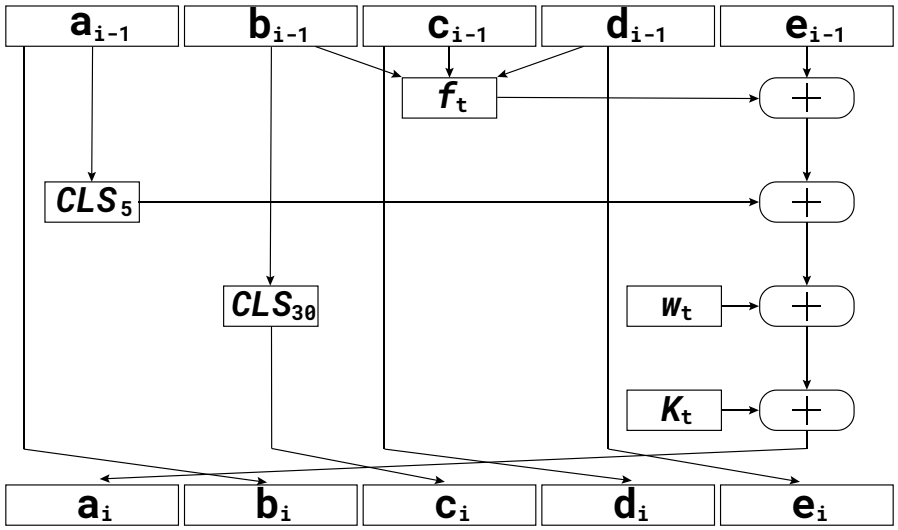
Untuk putaran $20 \leq t \leq 39$ $K_t = 6ED9EBA_{16}$

Untuk putaran $40 \leq t \leq 59$ $K_t = 8F1BBCDC_{16}$

Untuk putaran $6 \leq t \leq 79$ $K_t = CA62C1D6_{16}$

Untuk lebih jelasnya, ditunjukkan pada Gambar 2.8. yang terlihat bahwa dari 5 buah penyanggah a, b, c, d dan e masing-masing dioperasikan. Penyanggah a kemudian menjadi penyanggah b, dan nilai penyanggah a dilakukan operasi *circular left shift* sebanyak 5 kali, kemudian nilainya digunakan pada operasi penyanggah e. Penyanggah b dilakukan operasi *circular left shift* sebanyak 30 kali dan kemudian menjadi nilai penyanggah c. Nilai penyanggah b, c dan kemudian operasikan dengan fungsi *f* untuk kemudian digunakan pada rasi penyanggah e, nilai penyanggah d sendiri kemudian menjadi nilai yanggah e yang baru. Penyanggah e dioperasikan dengan menggunakan

hasil operasi penyanggah a, b, c dan d dan menggunakan parameter W_t dan K_t.



Gambar 2.8. Operasi SHA1 dalam satu putaran

Fungsi f_t adalah fungsi logika yang melakukan operasi *bitwise*. Operasi *bitwise* untuk setiap putaran diberikan sebagai berikut:

Putaran	$f_t(b,c,d)$
0 ... 19	$(b \wedge c) \vee (\sim b \wedge d)$
20 ... 39	$b \oplus c \oplus d$
40 ... 59	$(b \wedge c) \vee (b \wedge d) \vee (c \wedge d)$
60 ... 79	$b \oplus c \oplus d$

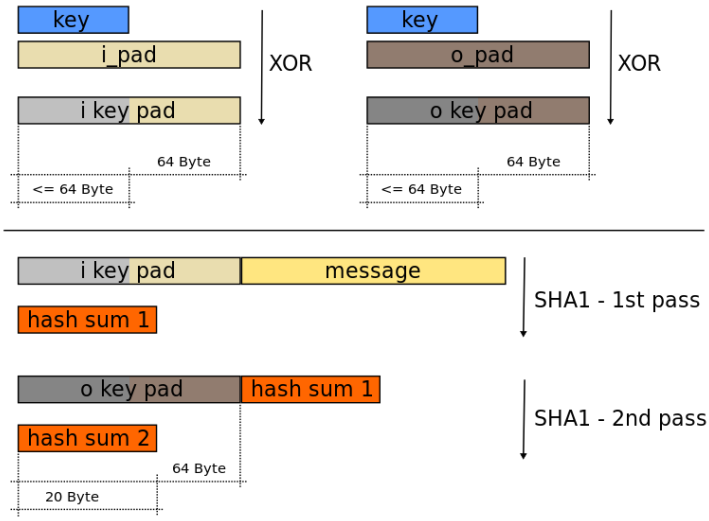
Nilai W_t diberikan sebagai berikut, untuk W_0 sampai W_{15} berasal dari 16 *word* pada blok yang sedang di proses, sedangkan untuk nilai W_t berikutnya diberikan oleh persamaan berikut ini:

$$W_t = W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3} \qquad (2.7)$$

Setelah putaran ke-79, *a*, *b*, *c*, *d* dan *e* dijumlahkan ke *A*, *B*, *C*, *D* dan *E* dan selanjutnya algoritma memproses blok berikutnya (Y_{q+1}). Hasil dari *A*, *B*, *C*, *D* dan *E* kemudian disambungkan yang kemudian menjadi hasil akhir dari algoritma *SHA-1*.

2.10. Algoritma *HMAC* Berbasis SHA-1

Salah satu contoh implementasi HMAC adalah pada algoritma fungsi *hash* SHA-1. Kunci yang digunakan adalah suatu string tertentu. Pada SHA-1, implementasi HMAC diberikan dalam diagram yang ditunjukkan dalam Gambar 2.9.. Implementasi HMAC mengikuti algoritma HMAC hanya pada algoritma fungsi *hash* menggunakan algoritma SHA-1 sebagai algoritmanya.



Gambar 2.9. HMAC pada SHA-1

2.11. Algoritma RSA

Algoritma RSA adalah salah satu algoritma kunci publik yang dibuat oleh 3 orang peneliti dari *MIT* (*Massachussets Institute of Technology*) pada tahun 1976, yaitu Ron Rivest, Adi Shamir dan Leonard Adleman. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima. Pemfaktoran ini dilakukan untuk mencari kunci privat.

Skema enkripsi RSA terdiri dari tiga bagian proses, yaitu proses pembangkitan sepasang kunci, proses enkripsi dan proses dekripsi. Skema ini menggunakan analogi Alice dan Bob.

1. **Pembangkitan Kunci**

Alice memilih dua bilangan prima *p* dan *q*. Sebaiknya nilai $p \neq q$ untuk menghindari nilai *p* diperoleh dengan mencari nilai $\sqrt{n^2}$. Kemudian Alice menghitung:

$$n = p.q \tag{2.8}$$

Kemudian Alice menghitung kembali:

$$\phi(n) = (p-1)(q-1) \tag{2.9}$$

Alice kemudian memilih kunci publik $e \in \{1, 2, \dots, \phi(n) - 1\}$ yang memenuhi persamaan:

$$\gcd(e, \phi(n)) = 1 \tag{2.10}$$

Alice kemudian menghitung kunci privat dengan menggunakan persamaan:

$$d \cdot e \equiv 1 (mod\ \phi(n)) \tag{2.11}$$

Hal yang sama juga dilakukan oleh Bob untuk membangkitkan pasangan bilangan kunci publik dan kunci privat.

Kunci publik yang digunakan adalah pasangan bilangan (*e*, *n*), dan kunci privat adalah pasangan bilangan (*d*, *n*). Nilai *n* tidak rahasia karena diperlukan pada perhitungan enkripsi dan dekripsi.

2. **Enkripsi**

Alur enkripsi bekerja sebagai berikut:

Misalnya Bob mengirim pesan *m* ke Alice, nilai *m* haruslah terletak di dalam selang [0, *n* -1], jika panjang pesan *m* melebihi nilai *n* – 1, maka pesan dibagi menjadi blok-blok dengan ukuran lebih kecil. Kemudian dengan menggunakan kunci publik Alice, yaitu *e* dan *n*, Bob menghitung:

$$c = m^e\ mod\ n \tag{2.12}$$

Nilai *c* kemudian dikirim kepada Alice sebagai pesan terenkripsi.

Dekripsi

Alur dekripsi bekerja sebagai berikut:

Misalnya pesan *m* diterima Alice. Alice kemudian menggunakan kunci privat *d* dan *n* untuk menghitung:

$$m = c^d \ mod \ n \tag{2.13}$$

2.12. Pesan Yang Digunakan

Konsep dari pesan yang digunakan adalah deretan huruf yang merupakan representasi dari hasil jawaban yang dianggap benar oleh peserta ujian CBT. Ujian yang diberikan adalah tipe ujian pilihan ganda, di mana peserta memilih satu dari beberapa pilihan jawaban. Jawaban yang diberikan pada setiap soal adalah merupakan salah satu dari huruf A, B, C, D dan E. Dengan jumlah soal sebanyak 100 soal, maka akan menghasilkan 100 buah huruf jawaban.

Untuk mengantisipasi peserta ujian tidak memberikan jawaban, maka setiap soal yang tidak dijawab oleh peserta ujian, maka jawabannya menjadi huruf F. Huruf F dipilih menjadi huruf *dummy* dikarenakan kedekatannya dengan deretan jawaban A, B, C, D dan E. Hal ini mengakibatkan nilai dari huruf F ketika direpresentasikan ke dalam sistem desimal untuk dioperasikan, memiliki nilai yang tidak terlalu jauh dari representasi desimal jawaban yang lain. Setelah semua soal dijawab oleh peserta ujian, maka jawaban digabungkan pada penyimpanan sementara untuk dioperasikan.

Jawaban yang diberikan akan dioperasikan dalam beberapa representasi bilangan yang berdasarkan pada *enkoding* ASCII (*American Standard Code for Information Interchange*) seperti ditunjukkan dalam Tabel 4.1.. ASCII adalah sebuah standar representasi setiap karakter yang ditetapkan oleh badan standarisasi Amerika Serikat. Representasi bilangan yang digunakan yaitu:

Tabel 2.1. Enkoding ASCII pada huruf kunci jawaban

No.	Huruf	Desimal	Heksadesimal
1	A	65	0041
2	B	66	0042
3	C	67	0043
4	D	68	0044
5	E	69	0045
6	F	70	0046