Aplikasi hash chain untuk kunci kamar hotel

Jon Felix Germinian - 13518025
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13518025@std.stei.itb.ac.id

***Abstract*— Keamanan hotel sering kali menjadi prioritas belakang dari hotel. Standar yang sering dipakai untuk hotel sekarang adalah kunci RFID. Kunci RFID menyimpan sebuah nilai yang akan dipakai untuk autentikasi. Proses ini memiliki banyak kelemahan, salah satunya disebabkan oleh mudahnya informasi dalam kartu RFID untuk disadap. Algoritma Hash Chain dapat digunakan untuk pembangkitan kunci dalam kartu RFID. Dengan Hash chain, pembangkitan kunci dapat dilakukan dengan cepat dan kunci lama akan dapat digunakan lagi.**

Keywords—Hotel, Hash Chain, Cryptography, SHA-256, Keamanan

I. PENDAHULUAN

Pariwisata adalah sektor ekonomi yang terus berkembang pesat. Saat melakukan sebuah wisata, wisatawan biasa memilih untuk menyewa sebuah kamar hotel. Tetapi keamanan bukanlah prioritas utama untuk sebuah hotel. Kebanyakan hotel masih menggunakan teknologi-teknologi masa lampau seperti Punch Card dan Kunci fisik. Kunci-kunci seperti itu memiliki kelemahan berbentuk fisik dan susah diduplikasi jika hilang. Kunci hotel yang lebih modern menggunakan Radio-frequency identification (RFID) dan magnetic stripe (magstripe). Dalam kunci tersebut, kunci RFID dan magstripe menyimpan kunci yang nanti akan dicocokkan dengan pembaca untuk membuka kamar. Kebanyakan sistem RFID masih menggunakan algoritma enkripsi yang simetris. Kunci RFID juga mudah diduplikasi, penyadap dapat mendekati pengujung dan mengambil informasi yang disimpan dalam kunci RFID.

Dalam makalah ini akan dibahas aplikasi hash chain untuk digunakan dalam pembangkitan kunci pintu kamar hotel. Dengan hash chain, kunci baru dapat dibangkitkan dengan mudah dan cepat. Pembangkitan dengan hash chain juga *irreversible* sehingga kunci sebelumnya tidak akan dapat lagi digunakan untuk membuka kamar hotel setelah kunci baru digunakan.

II. LANDASAN TEORI

A. *Kriptografi*

Kriptografi berasal kata Yunani “Kriptos” yang artinya “yang bersembunyi” dan “graphein” yang berarti tulisan. Secara etimologi, kriptografi dapat diartikan sebagai bidang ilmu tentang penyembunyian pesan. Dengan berkembangnya teknologi, lingkup yang masuk ke dalam bidang ini bertambah didasari ilmu matematika. Kriptografi modern mengunakan banyak teknik matematika untuk menyelesaikan persoalan keamanan.

Kriptografi mempelajari berbagai teknik enkripsi. Dalam enkripsi sebuah *plaintext* diproses melalui sebuah algoritma enkripsi menghasilkan sebuah *ciphertext*. Melalui proses ini, seseorang yang tidak memiliki kunci dekripsi tidak dapat membaca data tersebut.

Dalam kriptografi ada beberapa kosakata yang sering digunakan yaitu :

1. Plaintext

   Informasi yang dapat dibaca dan dimengerti maknanya. Pesan juga sering disebut pesan.

2. Pengirim

   Subjek yang mengirimkan sebuah pesan. Subjek pengirim dalam kriptografi dapat berupa manusia, komputer atau mesin

3. Penerima

   Subjek yang menerima sebuah pesan. Subjek penerima dalam kriptografi dapat berupa manusia, komputer atau mesin.

4. Ciphertexts

   Ciphertexts adalah pesan yang telah melewati proses enkripsi sehingga menjadi sulit untuk dibaca. Tujuan utama membuat sebuah *ciphertext* adalah untuk menjaga kerahasian sebuah pesan.

5. Enkripsi

   Enkripsi adalah proses mengubah sebuah plaintext menjadi ciphertext.

6. Dekripsi

   Dekripsi adalah kebalikan dari proses enkripsi. Dekripsi adalah proses mengubah sebuah ciphertext menjadi plaintext.

7. Kunci

   Kunci adalah parameter algoritma yang digunakan dalam proses enkripsi dan dekripsi

8. Penyadap

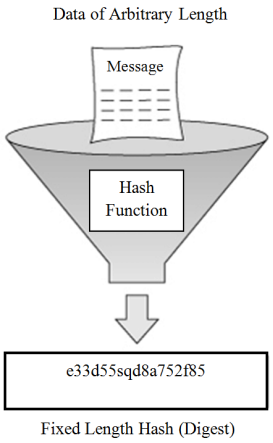
Penyadap adalah subjek ketiga yang mencoba menangkap pesan yang dikirim

9. Kriptanalisis

   Kriptanalisis adalah ilmu untuk memecahkan ciphertext kembali menjadi sebuah plaintext tanpa mengetahui kunci yang digunakan.

B. *Fungsi Hash*

Fungsi Hash adalah fungsi matematika yang dapat digunakan untuk memetakan data berukuran sembarang menjadi satu nilai yang memiliki ukuran tertentu atau *message-digest*. Fungsi hash bersifat satu arah atau *irreversible*, sebuah hash tidak dapat dikembalikan menjadi nilai masukkannya.



Gambar 1 Ilustrasi Fungsi Hash

Fungsi hash yang baik sebaiknya memenuhi kriteria sebagai berikut :

1. *Collision resistance*.

   Fungsi hash sulit untuk mendapatkan dua buah pesan yang memiliki message-digest yang bernilai sama. Fungsi hash harus susah menemukan dua sembarang input a dan b yang akan menghasilkan Hash(a) = Hash(b)

2. *Preimage resistance*.

   Fungsi hash sulit untuk menemukan pesan yang akan menghasilkan message-digest yang diinginkan. Untuk sembarang output x, fungsi hash harus sulit untuk menemukan input a yang akan menghasilkan Hash(a) = x

3. *Second preimage resistance*

   Fungsi hash susah untuk menemukan pesan yang memiliki nilai hash sama dengan sebuah pesan lain. Jika Hash(a) = x, fungsi hash harus sulit menemukan input lain sehingga Hash(b) = x

Dari ketiga kriteria yang harus dipenuhi sebuah fungsi hash, fungsi hash berguna untuk berbagai aplikasi seperti :

1. Menjaga integritas dari sebuah pesan

   Fungsi hash dapat menjaga integritas dari sebuah pesan karena perubahan kecil dari masukan akan berdampak besar terhadap message-digest.

2. Menormalkan panjang data yang beragam

   Fungsi hash akan selalu menghasilkan panjang message-digest yang seragam.

3. Mempercepat proses verifikasi

   Dengan fungsi-hash, jika verifikasi perlu dilakukan terhadap sebuah pesan dapat dipersingkat. Verifikasi dapat dilakukan terhadap nilai hash dari pesan dibandingkan memverifikasi seluruh pesan.

Ada beberapa implementasi fungsi hash sebagai berikut:

- MD2 (Message-Digest 2)
- MD4 (Message-Digest 3)
- MD5 (Message-Digest 5)
- RIPEMD
- RIPEMD-128/256
- RIPEMD-160/320
- SHA-0
- SHA-1
- SHA-256/224
- SHA-512/384

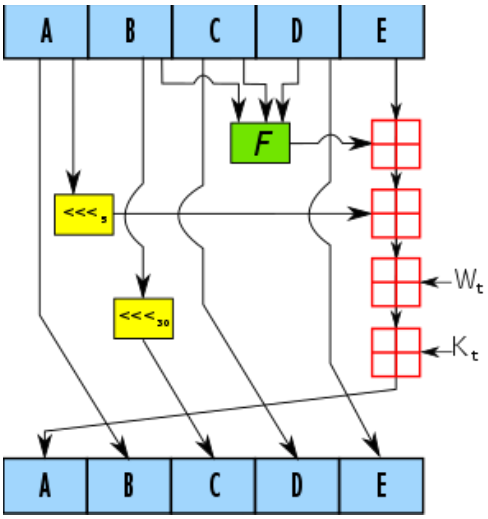
C. *Secure Hash Algoritm (SHA)*

SHA adalah algoritma hash yang dikembangkan oleh US National Security Agency (NSA) dan dipublikasi oleh National Institute of Standards and Technology (NIST) pada tahun 1993. Algoritma yang digunakan SHA dikembangkan dari algoritma fungsi hash MD4 yang sebelumnya telah dikembangkan oleh Ronald L. Rivest.

1) SHA-1

SHA-1 pertama kali keluar pada tahun 1995. Panjang keluaran yang dihasilkan SHA-1 berukuran 160-bit. Algoritma SHA-1 sudah tidak aman digunakan karena telah ditemukan *collision*. Cara kerja SHA-1 secara umum adalah sebagai berikut:

1. *Padding bits* dengan menambahkan bit-bit pengganjal ke dalam pesan
2. Penambahan nilai panjang pesan
3. Inisialisasi buffer message digest
4. Pengolahan pesan dalam blok 512-bit



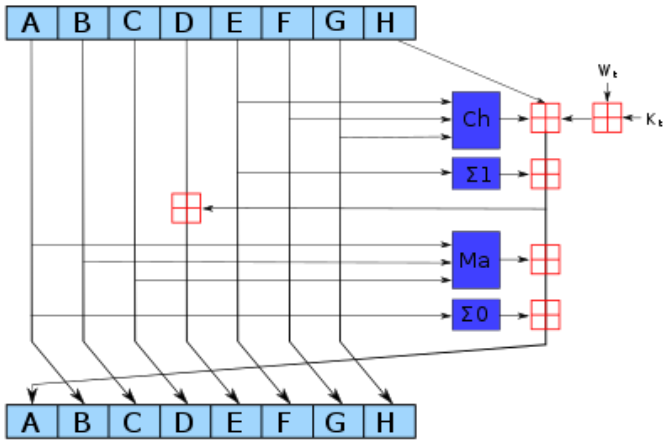
Gambar 2 Ilustrasi fungsi kompresi SHA-1

Sumber:

https://upload.wikimedia.org/wikipedia/commons/e/e2/SHA-1.svg

2) SHA-2

SHA-2 pertama kali dipublikasi pada tahun 2001. SHA-2 dikembangkan karena SHA-1 sudah tidak aman digunakan akibat collision. SHA-2 memiliki 2 variasi yaitu SHA-256 dan SHA-512. SHA-256 menerima masukan sembarang dan mengeluarkan sebuah hash berukuran 256-bit dan dapat dipotong menjadi 224-bit dengan variasi SHA-224. SHA-512 menerima masukan sembarang dan mengeluarkan sebuah hash berukuran 512-bit tetapi dapat dipotong menjadi 384-bit dengan variasi SHA-384. Sejauh ini (2021) SHA-2 masih aman digunakan karena belum ditemukan collision. Cara kerja SHA-2 mirip dengan SHA-1 tetapi terdapat perubahan pada fungsi kompresi yang dilakukan setiap putaran.



Gambar 3 Ilustrasi Fungsi Kompresi SHA-2

Sumber:

https://upload.wikimedia.org/wikipedia/commons/7/7d/SHA-2.svg

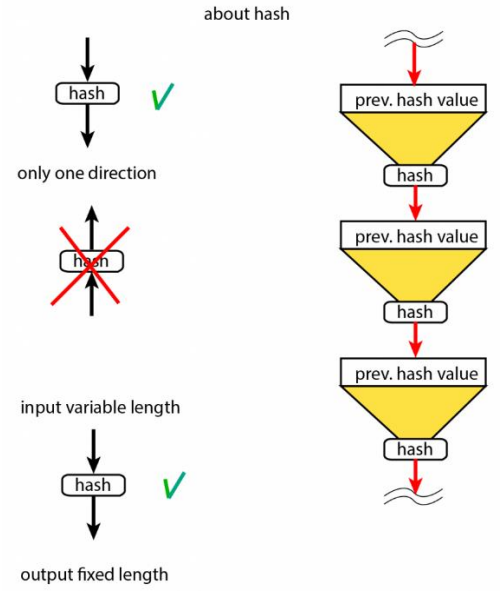
3) SHA-3

SHA-3 pertama kali dipublikasi pada tahun 2015. Lain cerita dari SHA-1 ke SHA-2. SHA-3 dikembangkan untuk menjadi alternatif lain dari SHA-2. SHA-3 pertama muncul dari kompetisi publik menciptakan algoritma hash paru pada tahun 2012. Pemenang dari kompetisi tersebut adalah algoritma Keccak yang dikembangkan oleh Lars Knudsem, Christian Rechberger, Martin Schlaffer, Soren S, Florian Mendel, Praveen Gauravaram dan Krystian Matusiewicz. SHA-3 memiliki beberapa varian yaitu SHA-3-224, SHA-3-256, SHA-3-284 dan SHA-3-512. Setiap varian dari SHA-3 memiliki proses berbeda pada algoritmanya dan memiliki nilai keluaran yang sama dengan nama variannya. Sejauh ini (2021) SHA-3 masih aman digunakan karena belum ditemukan collision.

D. Hash Chain

Hash chain adalah aplikasi fungsi hash untuk menciptakan banyak one-time keys. Secara definisi h(h(h(h(h(x))))) adalah hash chain dengan panjang 5 dan sering dinotasikan sebagai h5(x).

Hash Chain pertama kali disarankan oleh Lesie Lamport pada tahun 1981 untuk menjadi skema pembangkitan banyak password sekali pakai di lingkungan tidak aman. Autentikasi akan dilakukan terhadap sebuah hash chain dibandingkan sebuah password biasa untuk mencegah penyadap menggunakan password yang telah dicuri untuk keperluan yang tidak baik.



Gambar 4 Ilustrasi Hash Chain

Sebagai contoh, sebuah server membangkitkan sebuah hash-chain meyimpan h^{100}(password) dalam server dan memberikan sandi h^{99}(password) kepada pengguna. Saat pengguna masuk, pengguna akan memberikan h^{99}(password) kepada server dan server akan membandingkan h(h^{99}(password)) = h^{100}(password). Setelah autentikasi, server akan memberikan h^{98}(password) sebagai password sandi baru pengguna dan menyimpan nilai h^{99}(password). Penyadap yang mendapatkan h^{99}(password) tidak dapat masuk mengautentikasi dirinya sebagai pengguna

diakibatkan hash chain yang disimpan server sudah maju satu langkah.

III. Rancangan Solusi

Solusi yang diusulkan adalah untuk mengimplementasi sebuah hash chain untuk memverifikasi kunci pengguna. Solusi ini dapat dirancang dari sifat fungsi hash yang bergerak satu arah.

Resepsi hotel perlu menyimpan secret dari pintu kamar hotel dan sebuah nilai n. Pintu kamar hotel perlu menyimpan Hash ke-n+1 dari secret. Setiap pengunjung yang check-in ke dalam hotel, pengunjung diberikan key yang berupa hash ke-n dari secret, dan nilai n dari resepsi hotel dikurangi 1. Untuk membuka kamar hotel, pengecekan akan dilakukan terhadap kunci. Jika hash dari key pengujung sama dengan nilai nilai yang disimpan pintu, pintu akan terbuka. Jika hash ke-2 dari key pengguna sama dengan nilai yang disimpan pintu, pintu akan menyimpan nilai baru yaitu hash dari key dan pintu akan terbuka. Untuk kasus lain, pintu akan tetap terkunci.

Algoritma fungsi hash yang digunakan adalah SHA-256 karena aman dan sudah terdapat kakas yang dapat langsung digunakan untuk bahasa pemrograman python.

IV. Implementasi

Untuk implementasi, program ditulis dalam bahasa python dengan menggunakan kakas hashlib untuk algortima SHA-256.

A. Pintu Kamar Hotel

Berikut adalah objek pintu kamar hotel. Pintu kamar hotel memiliki 1 fungsi untuk membuka pintu dengan mengecek kesamaan nilai yang disimpannya dan hash dari kunci.

```
class hotel_door_lock:
    def __init__ (self, secret, n):
        hash = sha256(secret.encode())

        for i in range(n+1):
            hash = sha256(hash.digest())

        self.__x = hash

    def open_lock(self, k):

        h1k = sha256(k).digest()
        h2k = sha256(h1k).digest()

        if h1k == self.__x.digest():
            print('Door Open')
            return

        if h2k == self.__x.digest():
            print('Door Open (New Guest)')
            self.__x = sha256(k)
            return

        print('Door Access Denied')
```

B. Kunci Pengunjung

Berikut adalah fungsi untuk membangkitkan kunci dari pengunjung. Pengunjung akan mendapatkan kunci bernilai hash ke-n dari secret yang disimpan oleh resepsi hotel.

```
def make_key(guestname, s, n):
    f = open(guestname + '_key.txt', 'w')
    key = generate_hotel_key(s,n)

    f.write(key.hexdigest())
    f.close()

def open_hotel_door(hotel_door_lock, keyfile):
    f = open(keyfile, 'r')
    key = f.read()
    key_digest = bytes.fromhex(key)
    f.close()

    hotel_door_lock.open_lock(key_digest)

def generate_hotel_key (s, n):
    hash = sha256(s.encode())

    for i in range(n):
        hash = sha256(hash.digest())

    return hash
```

V. Pengujian

Pengujian dilakukan dengan 3 skema. Pengujian dilakukan terhadap secret resepsi hotel bernilai ‘O-ya? O-ya-ya-ya?”. Berikut adalah 3 skema yang telah disiapkan untuk pengujuan:

Alice (pengujung pertama) memiliki Kunci yang berupa hash ke-10000 dari secret sebagai berikut :

Kunci Alice
9f16b76ef3fede09c9379400f5a3fda7b0a9287d2f84a1a763332722bced354b

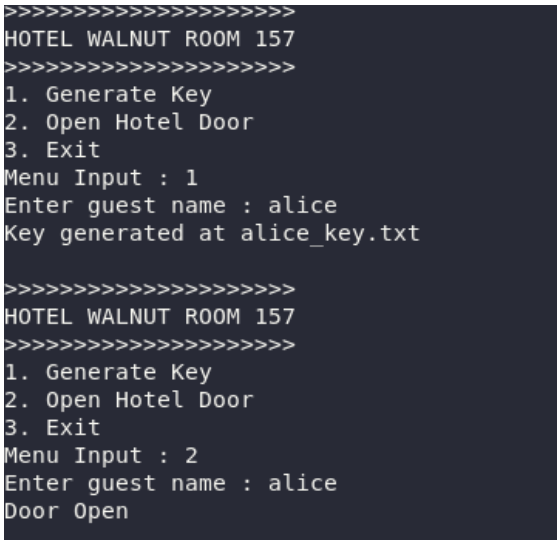
Bob (pengunjung kedua) memiliki kunci yang berupa hash ke-9999 dari secret sebagai berikut :

Kunci Bob
041baf7b14f8ea3607cd4fa9fd590c9b9fa89f154f2d2fda9e62aa144aa9bc08

Charlie (pengunjung ketiga) memiliki kunci yang berupa hash ke-9998 dari secret sebagai berikut :

Kunci Charlie
c0fb7354d0ac20d0a327a500a1af6e62063f7a1818850590f0396f257debc091

1. Pengujian untuk membuka kamar hotel dengan kunci yang baru dibangkitkan.



Gambar 5 Pengujian 1 Alice masuk ke kamar setelah check-in

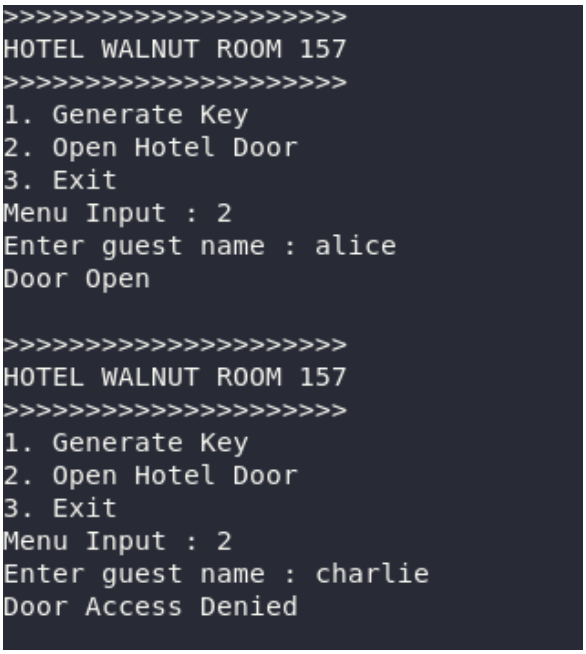
Dapat dilihat dari pengujian ini bahwa Alice (pengunjung ke-1) berhasil masuk ke kamar hotel setelah melakukan check-in.

2. Pengujian untuk membuka pintu kamar hotel oleh pengujung berikutnya.

Gambar 6 Pengujian 2 Bob sudah masuk ke kamar dan Alice ingin membuka pintu

Dapat dilihat dalam pengujian 2 bahwa Alice (Pengunjung ke-1) tidak dapat masuk kembali ke kamar setelah Bob (pengunjung ke-2) sudah masuk ke kamar hotel.

3. Pengujian ketiga adalah pengunjung pertama telah membuka pintu kamar dan pengujung ke-3 ingin mencoba membuka pintu hotel sebelum pengunjung ke-2 membuka pintu



Gambar 7 Pengujian 3 Ada pengunjung yang check-in tetapi tidak masuk ke dalam kamar

Dapat dilihat bahwa dalam pengujian 3, Charlie (pengunjung ke-3) tidak dapat masuk ke kamar hotel jika Bob (pengunjung ke-2) belum masuk ke kamar hotel.

VI. Simpulan dan Saran

Berdasarkan hasil percobaan, algortima hash chain dapat digunakan untuk autentikasi kunci dari pengunjung hotel. Secara aspek keamanan, metode ini menjamin bahwa pengunjung sebelumnya tidak akan bisa membuka setelah pengunjung berikutnya telah check-in dan masuk ke dalam kamar. Metode ini juga baik untuk skala lebih besar karena tidak dibutuhkan koneksi antara meja resepsi dan pintu kamar hotel setelah sinkronisasi awal. Ada satu masalah yang dapat dihadapi jika ada pengunjung yang check in tetapi tidak masuk ke dalam kamar. Dalam kasus tersebut, staf hotel harus masuk ke kamar hotel untuk membuka kamar tersebut.

Acknowledgement

Puji syukur kepada tuhan yang maha esa telah selesainya rangkaian kuliah IF4020 Kriptografi Semester 1 Tahun 2021/2022. Dalam proses penulisan makalah yang berjudul "Aplikasi Hash Chain untuk Kunci Kamar Hotel", penulis menerima banyak bantuan dari sekitar baik materil maupun moral. Penulis ingin mengucapkan terima kasih kepada dosen pengampu kuliah Pak Rinaldi Munir, S.T, M.T. yang telah memberikan wawasan dan ilmu terkait bidang kriptografi.

Referensi

[1] Munir, Rinaldi "Pengantar Kriptografi". https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2021-2022/Pengantar-Kriptografi-(2021).pdf. Diakses pada 19 Desember 2021.

[2] Munir, Rinaldi “Secure Hash Algorithm (SHA)”. https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Fungsi-hash-SHA.pdf. Diakses pada 19 Desember 2021.

[3] Munir, Rinaldi “Fungsi Hash”. https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Fungsi-hash-SHA.pdf. Diakses pada 19 Desember 2021

[4] Munir, Rinaldi “Peran Hash di dalam Blockchain”. https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Fungsi-hash-SHA.pdf. Diakses pada 19 Desember 2021

[5] Fisher, Tim “What is SHA-1 and How is it Used For Data Verification”. https://www.lifewire.com/what-is-sha-1-2626011. Diakses pada 20 Desember 2021

[6] Defu Liu, Guowu Yang, Yong Huang, Jinzhao Wu, "Inductive Method for Evaluating RFID Security Protocols", *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 2138468, 8 pages, 2019. https://doi.org/10.1155/2019/2138468

VII. PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Jakarta , 20 Desember 2021

Jon Felix Germinian
13518025