Analisis Penggunaan HMAC - SHA256 pada Keamanan Aplikasi Chatting

Dewa Ayu Mutiara Kirana Praba Dewi (18220084)
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
mutiarakirana21@gmail.com

***Abstrak*—Perkembangan digital yang pesat memberikan kemudahan dalam bidang komunikasi. Komunikasi digital membuat manusia dapat bertukar informasi tanpa batasan ruang dan waktu melalui aplikasi *chatting.* Terkadang, informasi atau pesan yang dikirimkan bersifat rahasia, sehingga meningkatkan kebutuhan penerapan protokol keamanan yang baik untuk menjaga privasi dan keamanan pesan. Oleh karena itu, umumnya diterapkan mekanisme *end-to-end encryption* yang menggunakan HMAC-SHA256 untuk menjaga integritas dan otentikasi pengguna pada aplikasi *chatting*. Penggunaan HMAC-SHA256 memberikan tingkat keamanan yang baik dan efisien untuk mendukung pertukaran pesan secara instan dan aman. Makalah ini akan menganalisis penggunaan HMAC-SHA256 yang lebih dipilih dibandingkan dengan penerapan HMAC yang menggunakan fungsi *hash* lainnya dalam mengatasi masalah keamanan pada aplikasi *chatting*.**

Keywords—Aplikasi chatting, Fungsi hash, HMAC-SHA256

I. PENDAHULUAN

Perkembangan digital yang sangat pesat menyebabkan pertukaran informasi dan interaksi dapat terjadi melalui internet tanpa ada batasan ruang dan waktu. Per Januari 2022, pengguna Internet di Indonesia telah mencapai angka 204,7 juta pengguna, yang menunjukkan peningkatan pesat dari tahun 2018 dengan pengguna internet hanya sebanyak 132,7 juta pengguna. Peningkatan data pengguna internet di Indonesia membuktikan bahwa penggunaan internet menjadi semakin intensif dalam kehidupan manusia. Salah satu hal yang populer dalam penggunaan internet saat ini, yaitu adanya komunikasi secara digital.

Peran teknologi internet telah memberikan kemudahan dalam bidang komunikasi. Adanya komunikasi secara digital memungkinkan manusia untuk berinteraksi secara instan, kapanpun dan dimanapun. Melalui platform *messaging*, pesan yang dikirim dapat diterima dengan cepat oleh pihak yang dituju. Hal ini memungkinkan pertukaran informasi dilakukan secara *real-time* dan membuat komunikasi menjadi lebih efisien. Terkadang, komunikasi yang dilakukan antarpihak melibatkan pertukaran informasi yang bersifat rahasia. Informasi atau pesan yang bersifat rahasia hanya boleh diketahui antarpihak yang berkomunikasi. Dalam kasus ini, apabila menggunakan platform *messaging* atau aplikasi *chatting,* seperti Whatsapp, Telegram, LINE, ataupun Signal, terdapat pihak lainnya berupa server yang dapat mengetahui pesan yang bersifat rahasia tersebut. Masalah privasi ini kemudian menjadi perhatian untuk pengembangan proteksi keamanan berkomunikasi melalui aplikasi *chatting*. Sebagai bentuk implementasinya, dilakukan proses enkripsi pesan yang disimpan dalam database server, sehingga pesan asli hanya diketahui oleh pihak yang terlibat dalam komunikasi.

Proses enkripsi pesan yang disimpan pada database server tidaklah cukup untuk mengatasi masalah keamanan dalam berkomunikasi melalui platform *messaging*. Selama perjalanan, pesan yang dikirimkan dari *client* ke *server* bisa saja diserang oleh pihak tertentu. Hal ini melanggar aspek *confidentiality.* Untuk itu, diperlukan pengembangan metode proteksi keamanan yang lebih kuat dalam berkomunikasi melalui platform *messaging*.

Dalam memaksimalkan penggunaan dan kemudahan komunikasi yang diberikan oleh platform *messaging*, diaplikasikan konsep ilmu kriptografi lainnya, selain proses enkripsi dan dekripsi, sebagai solusi atas proteksi keamanan yang lebih *advanced* agar pesan-pesan yang dikirimkan dalam berkomunikasi melalui platform *messaging* dapat terjamin kerahasiaannya. Salah satu cara yang dapat dilakukan, yaitu berupa penerapan mekanisme *Message Authentication Code* (MAC). Teknik MAC merupakan teknik yang menggunakan fungsi hash satu-arah dan kunci rahasia (*secret key*) untuk membangkitkan kode berupa nilai *hash* yang dihasilkan. Teknik ini dapat menjamin keamanan pesan yang dikirimkan dari segi integritas dan otentikasi pengguna, sehingga pesan tidak dapat diserang oleh pihak ketiga yang tidak bersangkutan dengan komunikasi yang dilakukan. Dalam kasus menjaga keamanan pada platform *messaging* atau aplikasi *chatting,* MAC yang biasanya digunakan berupa *keyed-Hash Message Authentication Code* (HMAC) dengan menggunakan fungsi hash SHA-256 atau biasa disebut dengan HMAC-SHA256.

Pada konsep HMAC, terdapat pilihan fungsi hash lainnya selain SHA-256, seperti MD5, SHA-1, ataupun SHA-512. Namun, berbagai kelemahan dan kompleksitas yang terdapat pada fungsi hash lainnya membuat HMAC-SHA256 lebih umum digunakan pada aplikasi *chatting*. Dalam makalah ini, penulis melakukan analisis terkait penggunaan HMAC-SHA256 yang lebih dipilih dibandingkan dengan HMAC yang menggunakan fungsi hash lainnya dalam konteks penerapannya untuk meningkatkan keamanan pada aplikasi

chatting. Melalui analisis ini, diharapkan dapat memberikan pemahaman yang lebih baik terkait pentingnya menggunakan HMAC-SHA256 dibandingkan dengan penerapan HMAC yang menggunakan fungsi hash lainnya dalam mengamankan proses integritas dan otentikasi pengguna, serta mencegah serangan yang mungkin terjadi pada aplikasi *chatting*.

II. Metodologi Penelitian

A. Metode Penelitian

Pembuatan makalah ini dilakukan dengan melakukan studi literatur, yaitu pencarian informasi melalui internet, berupa materi perkuliahan, jurnal, *paper*, web, ataupun sumber lainnya, yang memberikan informasi terkait penerapan HMAC-SHA256 dalam mengamankan pesan pada aplikasi *chatting*, serta informasi terkait HMAC yang menggunakan fungsi hash lainnya. Hasil yang diperoleh dari berbagai studi literatur yang dilakukan akan dibandingkan untuk memberikan kesimpulan terkait tingkat keamanan yang diberikan oleh HMAC-SHA256 dibandingkan dengan HMAC yang menggunakan fungsi *hash* lainnya dalam konteks aplikasi *chatting*.

B. Batasan Penulisan

Pada makalah ini, penelitian dibatasi dengan pemanfaatan HMAC sebagai mekanisme keamanan yang diterapkan pada aplikasi *chatting*, khususnya HMAC-SHA256 yang lebih umum digunakan dibandingkan dengan penggunaan HMAC yang menggunakan fungsi *hash* lainnya, yaitu HMAC-SHAMD5, HMAC-SHA1, dan HMAC-SHA512, dalam mengamankan pesan pada aplikasi *chatting*.

III. Dasar Teori

A. Fungsi Hash

Fungsi *hash* merupakan fungsi yang melakukan komporesi suatu pesan dengan ukuran sembarang menjadi *string* yang berukuran tetap (*fixed*). Keluaran atau hasil dari fungsi *hash* ini dikenal dengan pesan ringkas (*message-digest*) atau nilai *hash* (*hash value*). Suatu pesan yang telah diproses dengan fungsi *hash* tidak dapat dikembalikan lagi menjadi pesan semula karena fungsi *hash* bersifat *irreversible*.

Berikut merupakan ilustrasi dari implementasi algoritma fungsi *hash*.

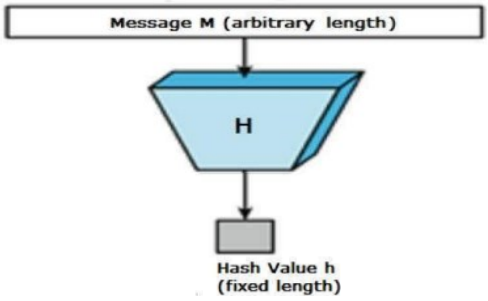


Fig. 1. Ilustrasi Implementasi Algoritma Fungsi *Hash* (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/17%20-%20Fungsi-hash-2021.pdf)

Keterangan:
- Pesan = M
- H = fungsi *hash*
- h = *hash value* atau *message digest*

Ilustrasi di atas memenuhi persamaan berikut.

$$h = H(M)$$

Fungsi *hash* memiliki beberapa sifat yang dijabarkan sebagai berikut.

1. *Collision Resistance*
   Sifat ini menyatakan bahwa sangat sulit untuk menemukan dua masukan a dan b yang menyebabkan terjadinya H(a) = H(b).
2. *Preimage Resistance*
   Pada fungsi *hash,* untuk sembarang keluaran (y), sulit untuk menemukan suatu input (a) yang menyebabkan H(a) = y.
3. *Second Preimage Resistance*
   Apabila terdapat masukan (a) dan keluaran (y) yang membuat terjadinya y = H(a), sulit untuk mendapatkan masukan kedua (b) yang menyebabkan H(b) = y.

Terdapat beberapa fungsi *hash* satu-arah yang ada dalam kriptografi. Berikut beberapa fungsi *hash* tersebut.

1. MD5
   MD5 merupakan fungsi *hash* satu-arah yang dibuat sebagai perbaikan dari MD4 oleh Ron Rivest. Algoritma MD5 menghasilkan *message digest* berukuran 128 bit dari masukan pesan berukuran sembarang.

   Berikut merupakan ilustrasi dari implementasi algoritma fungsi *hash* MD5.

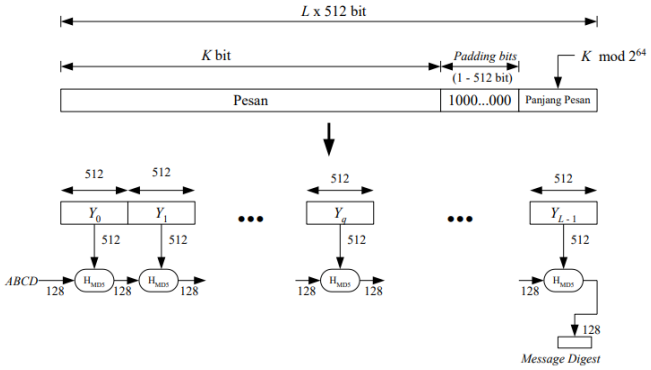


Fig. 2. Ilustrasi Implementasi Algoritma Fungsi *Hash* MD5 (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/18%20-%20Fungsi-hash-MD5-2021.pdf)

Message digest dihasilkan melalui penambahan bit-bit pengganjal pada pesan, yang kemudian ditambah dengan nilai panjang pesan semula. Lalu, dilakukan inisasi penyangga (MD) sebanyak 4 buah yang akan diinisialisasi dengan nilai-nilai dalam notasi heksadesimal. Setelah itu, pesan diolah dalam blok berukuran 512 bit yang akan diproses bersama penyangga MD dengan proses yang terdiri atas 4 buah

putaran, yang kemudian menghasilkan keluaran 128 bit.

2. SHA

Secure hash algorithm (SHA) merupakan fungsi *hash* satu-arah yang dibuat oleh NIST. Algoritma ini menghasilkan *message digest* 160 untuk SHA-1, 256/512 untuk SHA-2, dan berukuran *arbitrary* untuk SHA-3.

Berikut merupakan ilustrasi dari implementasi algoritma salah satu fungsi *hash* SHA, yaitu SHA-1.

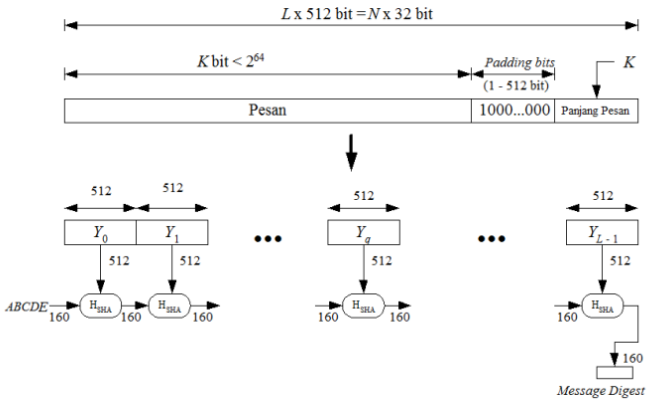


Fig. 3. Ilustrasi Implementasi Algoritma Fungsi *Hash* SHA-1 (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/19%20-%20Fungsi-hash-SHA-2021.pdf)

Message digest dihasilkan melalui cara yang mirip dengan MD5. Hanya saja terdapat perbedaan jumlah penyangga dan putaran yang dilakukan, yaitu pada SHA-1 membutuhkan 5 buah penyangga dengan eksekusi proses yang terdiri atas 80 buah putaran. Pada SHA-1 dihasilkan keluaran berukuran 160 bit.

B. *Message Authentication Code*

Message Authentication Code merupakan kode yang diperoleh melalui fungsi *hash* satu-arah dengan menggunakan kunci rahasia atau *secret key* untuk menghasilkan nilai *hash*. Algoritma MAC sebenarnya memiliki mekanisme yang mirip dengan fungsi *hash* biasa, hanya saja perbedaannya terletak pada komponen kunci rahasia yang digunakan oleh MAC sebagai masukan.

Berikut merupakan ilustrasi dari implementasi algoritma MAC secara umum.

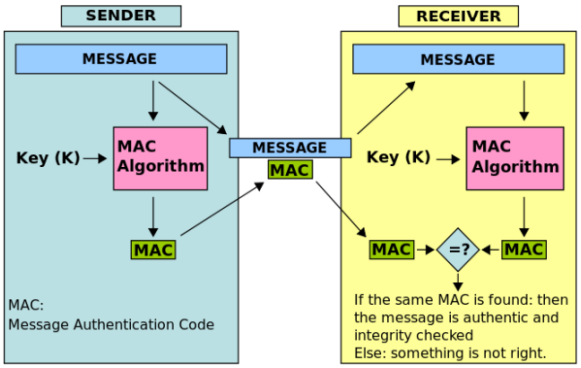


Fig. 4. Ilustrasi Implementasi Algoritma *Message Authentication Code* (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/22%20-%20MAC-2021.pdf)

Pesan yang dikirimkan dari pihak pengirim mula-mula akan dihitung MAC-nya menggunakan algoritma MAC dan kunci rahasia (k). MAC yang dihasilkan kemudian diletakkan (embed) pada pesan. Pesan dan MAC yang diletakkan (embed) pada pesan akan dikirimkan kepada penerima. Pada pihak penerima, pesan dan MAC akan dipisahkan. Pesan akan dihitung nilai MAC-nya menggunakan algoritma MAC dan kunci rahasia (k) yang sama dengan yang digunakan oleh pengirim. MAC yang dihasilkan pada pihak penerima kemudian dibandingkan dengan MAC yang diletakkan pada pesan. Apabila hasilnya sama, maka pesan yang diterima masih asli. Apabila hasilnya tidak sama, maka pesan yang diterima telah mengalami perubahan. Terdapat 2 algoritma MAC yang umum digunakan, yaitu algoritma MAC berbasis *block cipher* dan algoritma MAC berbasis fungsi *hash* satu-arah (HMAC).

Algoritma MAC berbasis *block cipher* mengimplementasikan mekanisme berupa pembangkitan MAC berdasarkan *block cipher*. *Block chiper* yang digunakan, yaitu mode CBC atau CFB. Hasil enkripsi blok terakhir dari *block cipher* dengan mode CBC atau CFB ini akan dijadikan sebagai nilai *hash* atau nilai yang akan menjadi MAC.

Berikut merupakan ilustrasi dari implementasi algoritma MAC berbasis *block cipher.*

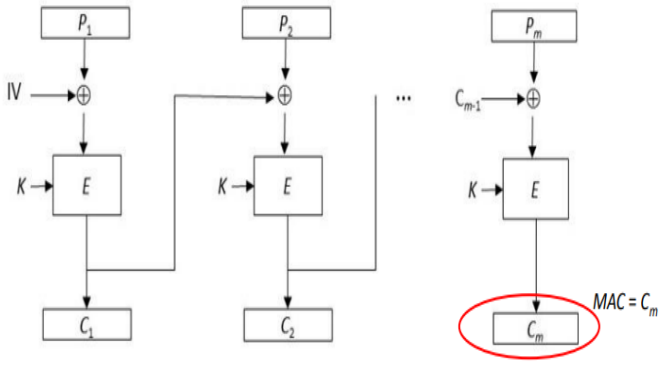


Fig. 5. Ilustrasi Implementasi Algoritma *Message Authentication Code* berbasis *block cipher* (sumber:

https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/22%20-%20MAC-2021.pdf)

MAC yang dihasilkan menggunakan algoritma MAC berbasis *block cipher* ini akan memiliki ukuran yang sama dengan ukuran bloknya.

Algoritma MAC berbasis fungsi *hash* satu-arah (HMAC) mengimplementasikan mekanisme yang menggunakan fungsi *hash* seperti MD5 ataupun SHA dalam menghasilkan nilai MAC.

Berikut merupakan ilustrasi dari implementasi algoritma MAC berbasis fungsi *hash* satu-arah (HMAC).

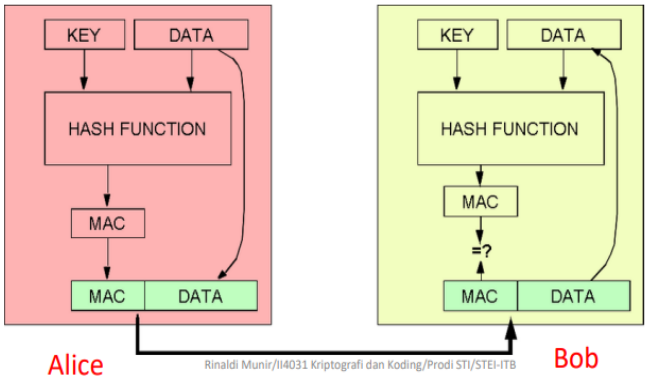


Fig. 6. Ilustrasi Implementasi Algoritma *Message Authentication Code* berbasis fungsi *hash* satu-arah (HMAC) (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/22%20-%20MAC-2021.pdf)

MAC yang dihasilkan menggunakan algoritma MAC berbasis fungsi *hash* satu-arah (HMAC) ini akan memiliki ukuran yang sama dengan ukuran *message digest* yang dihasilkan pada fungsi *hash* yang digunakan.

C. *HMAC-SHA256*

HMAC-SHA256 merupakan algoritma MAC berbasis fungsi *hash* satu-arah (HMAC) yang menggunakan fungsi *hash* SHA-256 untuk menghasilkan nilai MAC.

SHA-256 menghasilkan *message digest* dengan cara yang mirip dengan SHA-1, terdapat perbedaan yang terletak pada jumlah putaran yang dilakukan, yaitu pada SHA-256 memiliki eksekusi proses yang terdiri atas 64 buah putaran. SHA-256 ini menghasilkan keluaran berukuran 256 bit.

Berikut merupakan ilustrasi dari implementasi algoritma SHA-256.

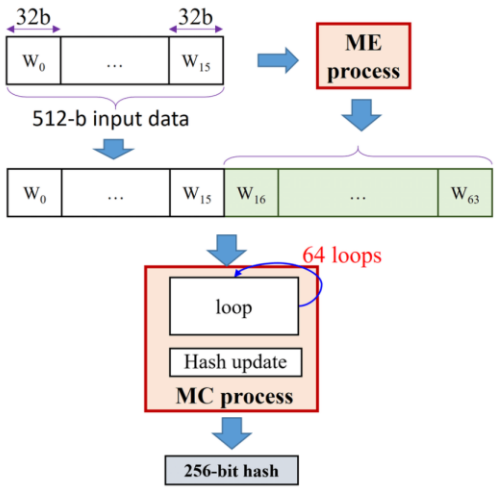


Fig. 7. Ilustrasi implementasi algoritma SHA-256 (sumber: https://www.researchgate.net/figure/The-overview-operation-of-the-SHA-256-algorithm_fig2_349744176)

Dalam HMAC-SHA256, pesan dan kunci rahasia yang telah ditentukan akan diolah menggunakan fungsi *hash* SHA256. *Message digest* yang dihasilkan oleh fungsi *hash* SHA256 akan menjadi nilai MAC yang akan diletakkan (embed) pada pesan. Kemudian, pesan dan nilai MAC yang telah di-*embed* tersebut akan dikirimkan ke penerima. Lalu, akan dilakukan validasi di sisi penerima menggunakan fungsi *hash* SHA256 dan kunci rahasia yang sama.

D. *Aplikasi Chatting*

Aplikasi *chatting* merupakan platform yang memfasilitasi komunikasi melalui jaringan internet. Aplikasi *chatting* memungkinkan pengirim dan penerima untuk dapat bertukar pesan secara instan. Melalui aplikasi *chatting*, pengirim dan penerima pesan dapat terhubung secara *real time*, mengatasi hambatan jarak dan waktu. Berbagai fitur yang tersedia pada aplikasi *chatting* memungkinkan pengguna untuk berbagi informasi secara efektif.

Terkadang, pesan yang dikirimkan melalui aplikasi *chatting* bersifat rahasia. Untuk menjaga keamanan pesan, aplikasi *chatting* menerapkan protokol *end-to-end encryption*. Dalam mekanisme *end-to-end encryption*, terdapat penerapan HMAC-SHA256 yang digunakan secara luas untuk menjaga integritas dan otentikasi pengguna, agar pesan tetap rahasia dan tidak dimanipulasi. Beberapa aplikasi *chatting* yang menggunakan HMAC-SHA256, yaitu Whatsapp, Signal, Telegram, dan iMessage.

IV. PEMBAHASAN

Pada pembahasan makalah ini, akan dijabarkan mengenai cara kerja HMAC-SHA256 pada aplikasi *chatting*. Selain itu, akan dibandingkan pula HMAC-SHA256 dengan HMAC yang menggunakan fungsi *hash* MD5, SHA-1, dan SHA-512 untuk menunjukkan mengapa HMAC-SHA256 lebih umum diterapkan dalam memberikan proteksi keamanan pesan pada aplikasi *chatting*.

Pada aplikasi *chatting,* dibentuk sesi untuk dapat melakukan pertukaran pesan. Lalu, setelah sesi terbentuk, akan dilakukan pertukaran pesan yang dilindungi dengan *message key*. *Message key* ini akan selalu berbeda pada setiap paket yang dikirimkan. Penggunaan HMAC-SHA256 diaplikasikan pada pembentukan *message key* ini sebagai bentuk keamanan integritas data, bersamaan dengan penggunaan ilmu kriptografi lainnya untuk menambah proteksi keamanan, seperti enkripsi AES.

Berikut merupakan diagram alir dari implementasi algoritma HMAC secara mendetail.

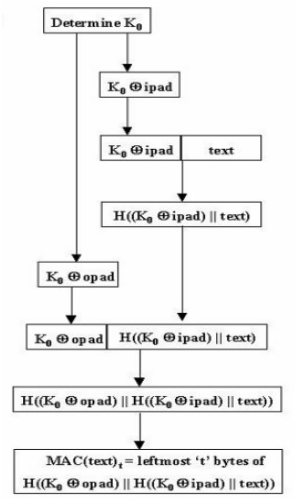


Fig. 8. Diagram alir HMAC
(sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2006-2007/Makalah2/Makalah-075.pdf)

Cara kerja HMAC pada diagram alir di atas ditunjukkan melalui persamaan berikut.

$$HMAC_k(text) = H((k \oplus opad) || H((k \oplus ipad) || text))$$

Dalam melakukan *hash* pada pesan, digunakan kunci (k) yang memiliki batas maksimal 64 karakter. Kunci *k* telah ditambahkan bit 0 agar ukurannya sebesar 512 bit. Text adalah pesan yang dikirimkan. H adalah fungsi *hash* yang digunakan, serta *ipad (inner padding)* dan *opad* (*outer padding*) adalah bit penambah yang berukuran 512 bit.

Algoritma fungsi *hash* SHA-256 yang digunakan pada HMAC di atas adalah sebagai berikut.

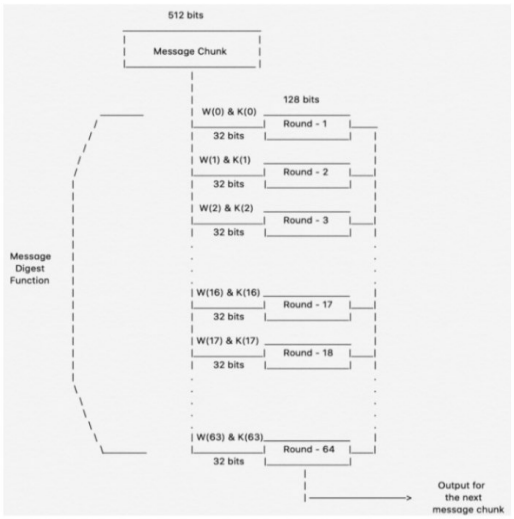


Fig. 9. Ilustrasi *compression function* dari SHA-256 (sumber: https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm)

Mekanisme kerja HMAC dengan proses fungsi hash SHA-256 tersebut yang akan diaplikasikan dalam pembentukan *message key* untuk mengamankan integritas data pada sesi pertukaran pesan dalam aplikasi *chatting*.

HMAC dengan menggunakan fungsi *hash* SHA-256 lebih aman dan efektif penggunaannya untuk mengamankan pesan dalam aplikasi *chatting* dibandingkan dengan HMAC yang menggunakan fungsi *hash* lainnya. Sebagai pembanding, berikut merupakan algoritma fungsi *hash* MD5.

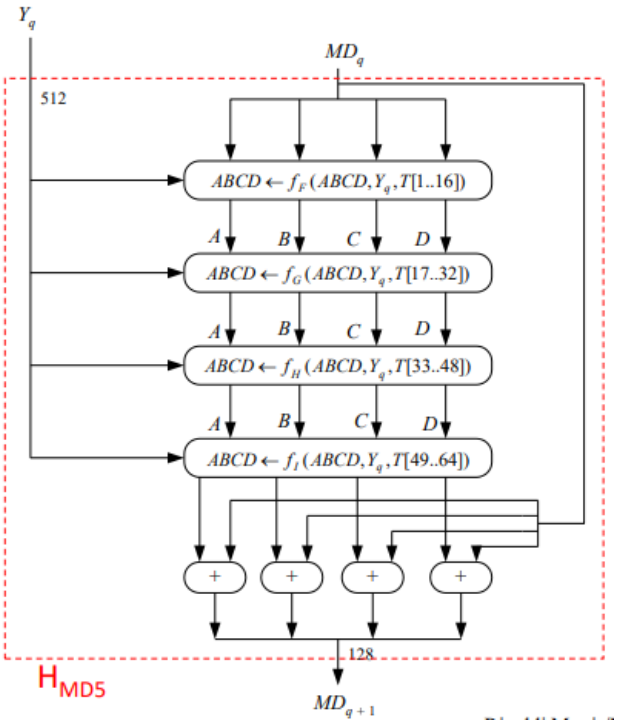


Fig. 10. Ilustrasi proses yang terjadi pada fungsi *hash* MD5 (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/18%20-%20Fungsi-hash-MD5-2021.pdf)

Dapat dilihat pada gambar di atas bahwa pada fungsi *hash* MD5, prosesnya hanya terdiri atas 4 putaran, sementara pada fungsi *hash* SHA-256 prosesnya terdiri atas 64 putaran. Meskipun jumlah putaran bukan satu-satunya faktor yang menentukan tingkat keamanan, tetapi secara teori, semakin banyak putaran yang dilakukan, maka semakin sulit bagi penyerang untuk menemukan celah keamanan pada algoritma terkait. Terlebih lagi, nilai *hash* yang dihasilkan pada fungsi *hash* MD5 berukuran 128 bit, sedangkan nilai *hash* yang dihasilkan pada fungsi *hash* SHA-256 berukuran 256 bit. Ukuran *hash* yang lebih besar memberikan ruang yang lebih besar untuk menghasilkan nilai *hash* yang unik, sehingga mengurangi kemungkinan terjadinya kolisi. Hal ini dibuktikan dengan kolisi yang pernah ditemukan pada algoritma MD5. Oleh karena itu, HMAC dengan menggunakan fungsi *hash* SHA-256 menjadi lebih aman dibandingkan dengan menggunakan fungsi *hash* MD5 dalam mengamankan pesan pada aplikasi *chatting*.

Sebagai pembanding lainnya, berikut merupakan algoritma fungsi *hash* SHA-1.

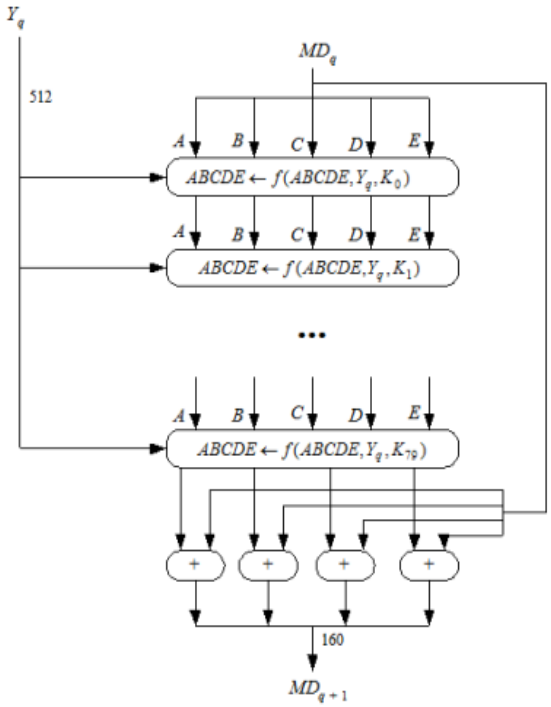


Fig. 11. Ilustrasi proses yang terjadi pada fungsi *hash* SHA-1 (sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/19%20-%20Fungsi-hash-SHA-2021.pdf)

Dapat dilihat pada gambar di atas, nilai *hash* yang dihasilkan pada fungsi *hash* SHA-1 berukuran 160 bit, sedangkan nilai *hash* yang dihasilkan pada fungsi *hash* SHA-256 berukuran 256 bit. Meskipun jumlah putaran yang dimiliki SHA-1 lebih banyak, yaitu sebanyak 80 putaran, nilai hash yang lebih besar dari SHA-256 membuat SHA-256 lebih aman dan tahan terhadap serangan, serta lebih kecil kemungkinan untuk terjadinya kolisi dibandingkan dengan SHA-1. Hal ini dibuktikan dengan kolisi yang pernah ditemukan pada algoritma SHA-1. Oleh karena itu, HMAC dengan menggunakan fungsi *hash* SHA-256 menjadi lebih aman dibandingkan dengan menggunakan fungsi *hash* SHA-1 dalam mengamankan pesan pada aplikasi *chatting*.

Sebagai pembanding lainnya, terdapat fungsi *hash* SHA-512 yang memiliki mekanisme yang mirip dengan SHA-256 dengan tingkat keamanan yang baik. Namun, fungsi *hash* SHA-512 tidak digunakan pada HMAC sebagai metode keamanan pada aplikasi *chatting* karena memerlukan waktu dan sumber daya komputasi yang lebih besar dibandingkan dengan SHA-256. Hal ini sejalan dengan ukuran nilai *hash* yang dihasilkan pada fungsi *hash* SHA-512 yang lebih besar dibandingkan dengan SHA-256. Dalam konteks aplikasi *chatting*, kecepatan dan efisiensi komputasi menjadi hal yang penting agar pertukaran pesan dapat terjadi secara *real time*, dengan tetap mempertimbangkan aspek keamanan. Penggunaan HMAC-SHA256 sudah cukup aman dan efisien untuk diterapkan pada aplikasi *chatting* karena sejauh ini belum ditemukan kolisi pada fungsi *hash* SHA256. Hal tersebut menyebabkan penggunaan HMAC-SHA512 belum diperlukan akibat performansi yang lebih baik yang dapat dilakukan oleh HMAC-SHA256 dengan tingkat keamanan yang sudah cukup baik.

V. KESIMPULAN

Berdasarkan hasil pembahasan, dapat disimpulkan bahwa HMAC-SHA256 merupakan algoritma yang cukup baik untuk menjaga keamanan dari segi integritas dan otentikasi pengguna pada aplikasi *chatting*. Penggunaan HMAC-SHA256 mampu memberikan tingkat keamanan yang baik dengan performansi yang efisien untuk mendukung pertukaran pesan secara instan. Berikut beberapa poin perbandingan yang menunjukkan bahwa HMAC-SHA256 lebih baik dibandingkan dengan HMAC menggunakan fungsi *hash* lainnya.

1. Fungsi *hash* SHA-256 memiliki jumlah *rounds* yang lebih banyak dibandingkan MD5, sehingga fungsi *hash* SHA-256 yang digunakan pada HMAC menjadi lebih sulit diserang celah keamanannya,
2. Nilai *hash* yang dihasilkan oleh fungsi *hash* SHA-256 lebih besar dibandingkan MD5 dan SHA-1, sehingga lebih kecil kemungkinan terjadinya kolisi pada SHA-256.
3. Kolisi sudah pernah ditemukan pada fungsi *hash* MD5 dan SHA-1, sedangkan hingga saat ini belum pernah ditemukan kolisi pada fungsi *hash* SHA-256.
4. HMAC-SHA256 memberikan tingkat keamanan yang sudah cukup baik dengan performansi yang lebih efisien dibandingkan dengan HMAC-SHA512 yang memerlukan waktu dan sumber daya yang lebih banyak.

VIDEO LINK AT YOUTUBE

https://youtu.be/llOMiVqKuak

UCAPAN TERIMA KASIH

Puji syukur saya ucapkan sebesar-besarnya kepada Tuhan Yang Maha Esa karena atas berkat dan rahmat-Nya saya diberikan kesehatan dan kelancaran untuk menyelesaikan tugas makalah ini. Tak lupa juga ucapan terima kasih saya sampaikan kepada keluarga atas dukungan yang diberikan kepada saya. Saya juga ingin mengucapkan terima kasih kepada Bapak Rinaldi Munir selaku dosen yang telah membimbing dan mengajarkan saya terkait Kriptografi dan Koding, sehingga saya dapat memahami bidang kriptografi dengan lebih baik.

Saya juga mengucapkan terima kasih atas bantuan pihak-pihak lainnya yang telah membantu saya dalam menyelesaikan makalah ini, seperti teman-teman seperjuangan pada mata kuliah II4031 Kriptografi dan Koding, kakak tingkat yang memberikan referensi pembuatan makalah, dan penulis-penulis yang mempublikasikan hasil penelitian yang sejalan dengan makalah ini di internet.

REFERENSI

[1] Adrisatria, Yogie (2007). "STUDI PENCARIAN KOLISI PADA SHA-1 OLEH XIAOYUN WANG dkk". [Online]. Available : https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2006-2007/Makalah2/Makalah-079.pdf [Diakses pada 21 Mei 2022].

[2] Ammiruddin, Muhammad Faqih Rohmani (2019). "Perancangan Spesifikasi Keamanan untuk Pengembangan Aplikasi Secure Chat Berdasarkan Common Criteria For It Security Evaluation". [Online]. Available : https://scholar.archive.org/work/5oscplkponhzhlcelmii5d4gei/access/wayback/https://jtiik.ub.ac.id/index.php/jtiik/article/download/3637/pdf [Diakses pada 20 Mei 2022].

[3] Annur, Cindy Mutia (2022). "Ada 204,7 Juta Pengguna Internet di Indonesia Awal 2022". [Online]. Available : https://databoks.katadata.co.id/datapublish/2022/03/23/ada-2047-juta-pengguna-internet-di-indonesia-awal-2022 [Diakses pada 20 Mei 2022].

[4] Anonim. "The overview operation of the SHA-256 algorithm". [Online]. Available: https://www.researchgate.net/figure/The-overview-operation-of-the-SHA-256-algorithm_fig2_349744176 [Diakses pada 21 Mei 2022].

[5] Frosch, Tilman, Christian Mainka, Christoph Bader, Florian Bergsma, Jorg Schwenk, Thorstenn Holz (2016). "How Secure is TextSecure?". [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7467371 [Diakses pada 20 Mei 2022].

[6] Jena, Baivab Kumar (2023). "A Definitive Guide to Learn The SHA-256 (Secure Hash Algorithms)" [Online]. Available : https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm [Diakses pada 20 Mei 2022].

[7] Munir, Rinaldi (2023). "MAC (Message Authentication Code)". [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/22%20-%20MAC-2021.pdf [Diakses pada 20 Mei 2022].

[8] Munir, Rinaldi (2023). "Fungsi Hash". [Online]. Available : https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/17%20-%20Fungsi-hash-2021.pdf [Diakses pada 20 Mei 2022].

[9] Munir, Rinaldi (2023). "Algortitma MD5". [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/18%20-%20Fungsi-hash-MD5-2021.pdf [Diakses pada 20 Mei 2022].

[10] Munir, Rinaldi (2023). "Secure Hash Algorithm (SHA)". [Online]. Available : https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/2021-2022/19%20-%20Fungsi-hash-SHA-2021.pdf [Diakses pada 20 Mei 2022].

[11] Nugroho, Prasetyo (2007). "Implementasi keyed-Hash Message Authentication Code-MD5 pada Aplikasi Instant Messenger Pidgin 2.2.0" [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2007-2008/Makalah2/MakalahIF5054-2007-B-062.pdf [Diakses pada 21 Mei 2022].

[12] Ramadhany, Taufik (2007). "Keyed-hash Message Authentication Code(HMAC)". [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2006-2007/Makalah2/Makalah-075.pdf [Diakses pada 21 Mei 2022].

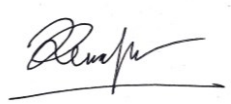
[13] Sanjaya, Fadly. "Analisis Fitur Keamanan Enkripsi End-to-end pada Aplikasi Chating Whatsapp". [Online]. Available: https://budi.rahardjo.id/files/courses/2016/EL6115-2016-23214353-Report.pdf [Diakses pada 20 Mei 2022].

[14] Xie, Tao, Dengguo Feng (2009). "How To Find Weak Input Differences For MD5 Collision Attacks". [Online]. Available: https://eprint.iacr.org/2009/223.pdf [Diakses pada 21 Mei 2022].

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.

Bandung, 22 Mei 2023



Dewa Ayu Mutiara K P D
18220084