Instituto Tecnológico de Estudios Superiores de Monterrey, Campus Guadalajara

MA2006B.401

Actividad Reporte SHA-1

Uso de álgebras modernas para seguridad y criptografía

Equipo:
Alberto Cortés
Diego Pérez
Luis Ramírez
Mariana Rizo

19 de Abril del 2022

1

¿Qué son las funciones de Hash?

Las funciones Hash convierten un conjunto de datos a un string corto de determinada longitud y cumplen con tres propiedades primarias: el cómputo de un hash debe ser rápido y simple, invertir la función debería ser difícil (tardado), y finalmente debe ser resistente a colisiones, es decir que debería ser difícil encontrar un documento D y otro documento B que compartan el mismo hash.

2

Explica el algoritmo de SHA-1.

El Secure Hash Algorithm 1 (SHA-1) se desarrolló como una función hash criptográfica estándar después de la creación de los estándares SSLv3 y TLSv1. SHA-1 se utiliza para verificar la integridad de los datos mediante la creación de un compendio de mensajes, que es una huella digital de los datos dados. El algoritmo es capaz de tomar un mensaje de cualquier tamaño y generar un compendio de mensajes de 160 bits. Aunque se ha descubierto que el algoritmo es vulnerable a los ataques de colisión, se sigue utilizando en algunas aplicaciones debido a su compatibilidad con versiones anteriores.

El algoritmo funciona tomando el mensaje de entrada y haciéndolo pasar por una serie de ciclos, cada uno de los cuales produce un valor hash. El mensaje se rellena con ceros para que sea un múltiplo de 512 bits. Una cadena de longitud L se denomina M(L), y el mensaje rellenado se denomina M*. A continuación, el mensaje se divide en bloques de 512 bits, denominados Mj. Cada bloque es procesado por una función que produce un valor hash. La función para el bloque i se denomina H(i). La salida de la función H(i) es un valor de 160 bits que se añade a un valor hash inicializado con una cadena de ceros. El valor hash final se emite como compendio del mensaje.

Este fue desarrollado por la NSA. La función toma un mensaje de longitud arbitraria y produce un compendio de mensajes de 160 bits. El resumen del mensaje puede utilizarse para verificar la integridad del mensaje.

El algoritmo está diseñado para que sea difícil encontrar dos mensajes diferentes que produzcan el mismo compendio de mensajes. Sin embargo, se ha demostrado que es posible encontrar colisiones para SHA-1. Esto significa que es posible crear dos mensajes diferentes que produzcan el mismo compendio de mensajes. Esto puede ser aprovechado por un atacante para crear un mensaje falsificado que parezca provenir de una fuente de confianza.

Existen diversas aplicaciones para el algoritmo como los certificados SSL/TLS, las firmas PGP/GPG y los commits de git. A pesar de que se ha descubierto que es vulnerable a los ataques de colisión, se sigue utilizando en algunas aplicaciones debido a su compatibilidad con versiones anteriores.

El Instituto Nacional de Estándares y Tecnología (NIST) ha recomendado que no se utilice para las firmas digitales, la inclusión en bases de datos de firmas a largo plazo o la generación de huellas dactilares de certificados, excepto como parte de un plan de retirada progresiva.

3

Antes del algoritmo SHA-1, ¿qué algoritmo se utilizaba (explique)?

Antes del algoritmo SHA-1 se utilizaban los algoritmos MD2, MD4 y MD5. Estos algoritmos fueron diseñados por Ronald Rivest en los últimos años de los 80's y los principios de los 90's. Sin embargo, se descubrió que el algoritmo MD5, el cual se diseño para reemplazar el algoritmo MD4, tenía muchas vulnerabilidades. Una de las vulnerabilidades del algoritmo es el tamaño del valor del hash (128 bits) es lo suficientemente pequeño para ejecutar un ataque de cumpleaños. Esto quiere decir que es fácil encontrar colisiones.

4

Realiza un ejemplo donde se aplique el algoritmo SHA-1.

Supongamos que deseamos hashear el string "abc", tenemos que en binario es:

$$01100001\ 01100010\ 01100011$$

y en hex:

$$616263$$

1. Inicializar 5 hex aleatorios.

$$H_0 = 67DE2A01$$
$$H_1 = BB03E28C$$
$$H_2 = 011EF1DC$$
$$H_3 = 9293E9E2$$
$$H_4 = CDEF23A9$$

2. Se rellena el mensaje seguido de un 1 y una cantidad de 0's suficiente hasta que se tengan 448 bits, al final se agregan 64 bits representando la longitud del mensaje en cuestión, teniendo un mensaje de 512 bits como resultado.

$$01100001\ 01100010\ 01100011\ 1\ 00...0\ 00...01100$$

3. El mensaje obtenido del pase anterior se divide en bloques de 512 bits y cada uno de ellos es a su vez dividido en 16 palabras de 32 bits $W_0...W_{15}$, para nuestro ejemplo tenemos exactamente 512 bits, por lo tanto solo se realiza una vez el siguiente proceso.

4. Para cada bloque (en este caso 1) se realizan iteraciones del siguiente proceso, terminando con 80 palabras en total, generando las 64 restantes a partir de las anteriores.

$$W(i) = S^1\left(W(i-3) \oplus W(i-8) \oplus W(i-14) \oplus W(i-16)\right) \text{ con } 16 \leq i \leq 79$$

Donde S^n está definida por:

$$S^n(X) = (X << n)\ OR\ (X >> 32 - n)$$

5. Se guardan los valores de hash definidos en el paso 1.

$$A = H_0$$
$$B = H_1$$
$$C = H_2$$
$$D = H_3$$
$$E = H_4$$

6. Se calcula en 80 iteraciones:

$$TEMP = S^5 * (A) + f(i; B, C, D) + E + W(i) + K(i) \text{ con } 0 \leq i \leq 79$$

Donde las definiciones de $f(i; B, C, D)$ y $K(i)$ están dadas por:
Para $0 \leq i \leq 19$

$$f(i; B, C, D) = (B \wedge C) \vee ((\neg B) \wedge D)$$
$$K(i) = 5A827999$$

Para $20 \leq i \leq 39$

$$f(i; B, C, D) = B \oplus C \oplus D$$
$$K(i) = 6ED9EBA1$$

Para $40 \leq i \leq 59$

$$f(i; B, C, D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$$
$$K(i) = 8F1BBCDC$$

Para $60 \leq i \leq 79$

$$f(i; B, C, D) = B \oplus C \oplus D$$
$$K(i) = CA62C1D6$$

Y en cada iteración se hacen las reasignaciones:

$$E = D$$
$$D = C$$
$$C = S^{30}(B)$$
$$B = A$$
$$A = TEMP$$

7. Se añade el hash obtenido para cada bloque al hash general de todo el mensaje original como se muestra a continuación y después se procesa el siguiente bloque (en este caso no hay más paquetes).

$$H_0 = H_0 + A$$
$$H_1 = H_1 + B$$
$$H_2 = H_2 + C$$
$$H_3 = H_3 + D$$
$$H_4 = H_4 + E$$

8. Para finalizar, una vez que se han procesado todos los bloques, el hash del mensaje es representado por un string de 160 bits utilizando el operador OR en los 5 valores hasheados en todo el proceso.

$$HH = S^{128}(H_0) \vee S^{96}(H_1) \vee S^{64}(H_2) \vee S^{32}(H_3) \vee H_4$$

Tenemos entonces que el hash asociado al string "abc" es **a9993e364706816aba3e25717850c26c9cd0d89d**.

References

[Bri,] *https://brilliant.org/wiki/secure-hashing-algorithms/.*

[MDs,] *https://en.wikipedia.org/wiki/MD5.*

[Hoffstein and Silverman, 2014] Hoffstein, P. and Silverman (2014). *8.1 Hash Functions.* Springer.