Manuel Camacho Padilla
A01423135
1/12/2020

Realizar en forma individual una investigación y reflexión de la importancia y eficiencia del uso de las tablas hash para tales fines.

Tabla HASH

Una tabla hash, matriz asociativa, hashing, mapa hash, tabla de dispersión o tabla fragmentada es una estructura de datos que asocia *llaves* o *claves* con *valores*. La operación principal que soporta de manera eficiente es la *búsqueda*: permite el acceso a los elementos almacenados a partir de una clave generada. Funciona transformando la clave con una función hash en un *hash*, un número que identifica la posición (*casilla* o *cubeta*) donde la tabla hash localiza el valor deseado.

Las tablas hash se suelen implementar sobre *vectores* de una dimensión, aunque se pueden hacer implementaciones multi-dimensionales basadas en varias claves. Como en el caso de los arrays, las tablas hash proveen tiempo constante de búsqueda promedio $O(1)^2$ sin importar el número de elementos en la tabla. Sin embargo, en casos particularmente malos el tiempo de búsqueda puede llegar a $O(n)$, es decir, en función del número de elementos.

Comparada con otras estructuras de arrays asociadas, las tablas hash son más útiles cuando se almacenan grandes cantidades de información.

Las tablas hash almacenan la información en posiciones pseudo-aleatorias, así que el acceso ordenado a su contenido es bastante lento. Otras estructuras como árboles binarios auto-balanceables tienen un tiempo promedio de búsqueda mayor (tiempo de búsqueda$(\log n)$), pero la información está ordenada en todo momento.

Ventajas e incovenientes

Una tabla *hash* tiene como principal ventaja que el acceso a los datos suele ser muy rápido si se cumplen las siguientes condiciones:

- Una razón de ocupación no muy elevada (a partir del 75% de ocupación se producen demasiadas colisiones y la tabla se vuelve ineficiente).
- Una función resumen que distribuya uniformemente las claves. Si la función está mal diseñada, se producirán muchas colisiones.

Los inconvenientes de las tablas *hash* son:

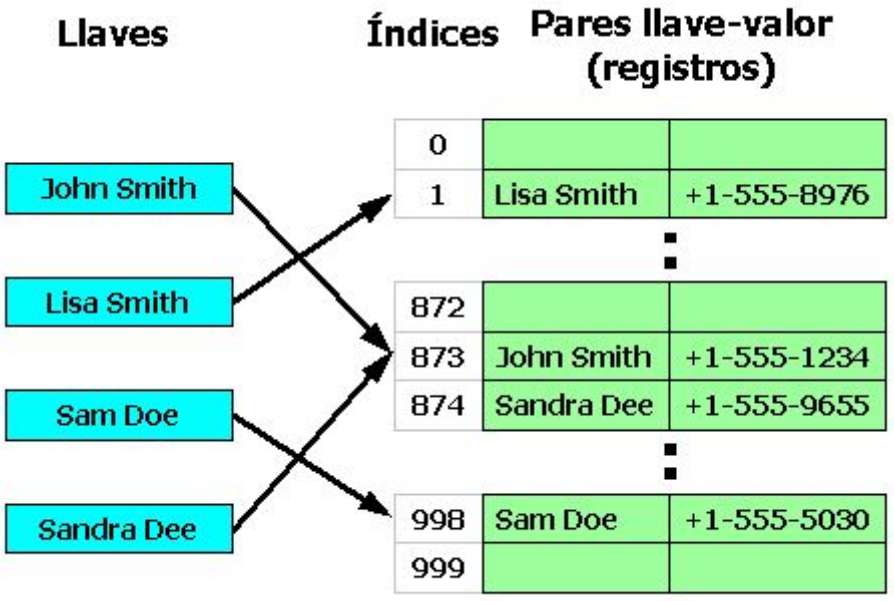
- Necesidad de ampliar el espacio de la tabla si el volumen de datos almacenados crece. Se trata de una operación costosa.

- Dificultad para recorrer todos los elementos. Se suelen emplear listas para procesar la totalidad de los elementos.
- Desaprovechamiento de la memoria. Si se reserva espacio para todos los posibles elementos, se consume más memoria de la necesaria; se suele resolver reservando espacio únicamente para punteros a los elementos.

Reflexión sobre las tablas Hash

A pesar de haber trabajado poco con las tablas Hash pude entender la teoría de lo que son. Incluso investigué por cuenta que existen diferentes tipos de tablas Hash creadas por computologos a través de los años. Y admiro el hecho de que se puedan crear estructuras de datos con este tipo de método. Mi parte favorita fue la creación de la llave para mis vectores, ya que es cuestión de hacer operaciones matemáticas eficaces para evitar el mayor número de colisiones. Creo que fue en donde más me quedé pensando en cómo convertir un número en una clave para su registro.

Quizás me falta reforzar más estas prácticas de estructuras de datos, ya que aún hay tanto que desconozco, pero estoy seguro que serán conceptos muy recurrentes y útiles para el desarrollo de software en el futuro.



El documento de reflexión deberá incluir de manera específica información sobre el uso de SHA-256, sus ventajas, desventajas y otros tipos de uso del SHA-256, así como su complejidad computacional.

¿Qué es?

Es un conjunto de funciones hash criptográficas (**SHA-224, SHA-256, SHA-384, SHA-512**) diseñadas por la Agencia de Seguridad Nacional (NSA) y publicada en 2001 por el Instituto Nacional de Estándares y Tecnología (NIST) como un Estándar Federal de Procesamiento de la Información (FIPS).

Una función hash es un algoritmo que transforma ("digiere") un conjunto arbitrario de elementos de datos, como puede ser un fichero de texto, en un único valor de longitud fija (el "hash"). El valor hash calculado puede ser utilizado para la verificación de la integridad de copias de un dato original sin la necesidad de proveer el dato original. Esta irreversibilidad significa que un valor hash puede ser libremente distribuido o almacenado, ya que sólo se utiliza para fines de comparación. SHA significa algoritmo de hash seguro. SHA-2 incluye un significativo número de cambios respecto a su predecesor, SHA-1; y consiste en un conjunto de cuatro funciones hash de 224, 256, 384 o 512 bits.

La seguridad proporcionada por un algoritmo hash es sumamente dependiente de su capacidad de producir un único valor para un conjunto de datos dados. Cuando una función hash produce el mismo valor para dos conjuntos de datos distintos, entonces se dice que se ha producido una colisión. Una colisión aumenta la posibilidad de que un atacante pueda elaborar computacionalmente conjuntos de datos que proporcionen acceso a información segura o para alterar ficheros de datos informáticos de tal forma que no cambiará el valor hash resultante y así eludir la detección. Una función hash fuerte es aquella que es resistente a este tipo de ataques computacionales mientras que una función hash débil es aquella donde existe una creencia casi certera de que se pueden producir colisiones. Finalmente, una función hash quebrantada es aquella sobre la que se conocen métodos computacionales para producir colisiones.

Aplicaciones

Las funciones hash SHA-2 están implementadas en una gran variedad de aplicaciones y protocolos de seguridad, como por ejemplo: TLS y SSL, PGP, SSH, S/MIME, Bitcoin, PPCoin y IPsec.

La moneda criptográfica Bitcoin depende en gran medida en un doble uso del SHA-256. El SHA-256 es usado para identificar los paquetes software de Debian GNU/Linux y en el estándar de mensaje firmado DKIM; SHA-512 es parte del sistema para identificar los vídeos guardados en el Tribunal Penal Internacional para Ruanda. SHA-256 y SHA-512 fueron propuestos para ser usados en DNSSEC.[18] Los proveedores de Unix y Linux están adoptando SHA-2 de 256 y 512 bits para aplicarlo en las contraseñas de seguridad.

SHA-1 y SHA-2 son algoritmos hash de seguridad requeridos por ley en ciertas aplicaciones del gobierno de Estados Unidos, junto con el uso de otros algoritmos y protocolos criptográficos, para la protección de información clasificada y sensible. FIPS PUB 180-1 también alentó la adopción y uso del SHA-1 por parte de organizaciones privadas y comerciales. El gobierno está dejando de utilizar SHA-1, tal como expone el U.S. National Institute of Standards and Technology, "Las agencias federales *deberían* dejar de usar el SHA-1 para... aplicaciones que necesiten resistencia de colisión tan pronto como sea posible, y deberán usar familias de funciones hash SHA-2 para estas aplicaciones después de 2010" (énfasis en el original). La directiva de NIST de que las agencias del gobierno estadounidense deban dejar de usar el SHA-1 después de 2010 y la finalización del SHA-3[2] deberían acelerar la migración del SHA-1.

Las funciones SHA-2 no son tan ampliamente usadas como SHA-1, a pesar de su mejora en seguridad. Las razones pueden incluir una falta de soporte del SHA-2 en sistemas que ejecutan Windows XP SP2 o anteriores, o por una falta de urgencia percibida mientras no se haya descubierto aún colisiones en el SHA-1.

Ventajas y desventajas

Las ventajas y desventajas están determinadas por caso de uso. No es posible responder a esta pregunta sin saber qué es exactamente lo que pretende aplicar y por qué.

Los tres hashes que enumerar tienen propósitos muy diferentes. Preguntar cuáles son sus ventajas y desventajas relativas.

SHA-2 es una función hash criptográfica y, por lo general, es un componente básico para otras construcciones criptográficas. Para satisfacer los requisitos del hash criptográfico, es una función unidireccional que es determinista, rápida de calcular, resistente a los ataques de preimagen y segunda preimagen y es resistente a colisiones.

Scrypt es una función de derivación de claves basada en contraseña. Se usa para convertir una contraseña de baja entropía en una clave criptográfica o un verificador con una entropía efectivamente más alta al ser intencionalmente lenta de calcular. Es ajustable para requerir mayores cantidades de CPU y / o memoria a medida que avanza la tecnología, lo que hace que el hardware dedicado a su computación sea más caro.

X11 de lo que puedo recopilar es una función de prueba de trabajo para monedas basadas en blockchain. Parece ser poco más que una combinación poco elegante de un montón de funciones hash no relacionadas con la ingenua esperanza de que eso de alguna manera lo hará resistente a ASIC y más seguro.

Posee varios niveles de complejidad:

MD5:

El primer estándar de su uso popular fue el MD5, Message-Digest Algorithm 5. Diseñado en el MIT en 1991, es la quinta versión de lo que denominan algoritmos de reducción de

mensajes. Genera salidas de 32 caracteres. Coincidió con la popularización de Internet y fue el sistema utilizado en las primeras capas de seguridad que se implementaron.
Poco después, se documentaron las primeras colisiones, lo que hizo que, como sus predecesoras, perdiera fiabilidad. Una colisión es cuando dos entradas de datos generan una misma salida. Lo que hace posible la falsificación de las claves.
A pesar de ello sigue utilizándose como generador de claves simples que no requieren grandes capas de seguridad.
SHA:
La primera versión de Secure Hash, SHA-1, lanzada en 1995 y diseñada para sustituir a MD5. Mucho más potente con salidas de 40 caracteres y más seguridad en la generación. Su uso se extendió rápidamente hasta que Google documentó la primera colisión.
Ante esta vulnerabilidad no tardaron en liberar una versión más robusta, SHA-256. En la actualidad podríamos decir que es la versión estándar de la industria tecnológica. Genera salidas de 64 caracteres y tiene una buena potencia de cómputo. No se le han detectado vulnerabilidades y el rendimiento es más que aceptable.
Aun así, ya se ha liberado una nueva versión junto con los sistemas Keccak, la SHA-3. Más robusta y con varios niveles de complejidad y de codificación. SHA3-224, SHA3-256, SHA3-384 y SHA3-512 generan diferentes salidas
RIPEMD
La versión europea de los algoritmos de reducción es RIPEMD (RACE Integrity Primitives Evaluation Message Digest). Liberada en 1996, es equivalente a SHA-1 pero sin colisiones conocidas y con niveles configurables de extensión de bits.
Scrypt
Tanto las familias MD como SHA y el europeo RIPEMD han sido implementadas por instituciones con ayudas gubernamentales. Por contra, Scrypt es completamente privada. Fue desarrollada por Colin Percival como respaldo para Tarsnap.
Es considerado el algoritmo más seguro pues gracias al pool de entropía añadido al complejo cálculo matemático. La longitud de la salida es configurable entre 64 y 4096 caracteres y se resiste perfectamente a los ataques de fuerza bruta. Y todo lo consigue sin una penalización en la velocidad de ejecución.
BLAKE
Está basado en la función criptográfica ChaCha de Dan Bernstein y está diseñado para competir con la familia SHA. Presenta unas prestaciones parecidas a SHA3 pero con una velocidad de cómputo mucho más alta.

Referencias:

- NA(2020) Diciembre 1, 2020. Tabla Hash. De Wikipedia, sitio web: https://es.wikipedia.org/wiki/Tabla_hash#:~:text=Las%20tablas%20hash%20se%20suelen,de%20elementos%20en%20la%20tabla.
- NA(2020) Diciembre 1, 2020. SHA-2. De Wikipedia, sitio web: https://es.wikipedia.org/wiki/SHA-2
- Touset, S. (2017). Diciembre 1, 2020. What Are Advantages and Disadvantages of SHA-256? De Stackovernet, sitio web: https://crypto.stackovernet.xyz/es/q/10202
- NA. (2020) Diciembre 1, 2020. Hash. De Fisiotrónica, sitio web: http://fisicotronica.com/hash-el-guardian-de-nuestra-seguridad/