Trabajo final Nro 1

Cátedra:

Arquitectura de redes y servicio

Responsables de cátedra:

Echaiz Javier , Pap Fernando

-U.N.P.S.J.B. FACULTAD DE INGENIERÍA-

Sede Puerto Madryn

Alumno:

Andoro Adrian

Hash

1. **Los algoritmos de hash (md5, sha-x, etc.) no se utilizan para cifrar mensajes. ¿Por qué?**

   Porque utiliza algoritmos que aseguran que con la respuesta (o hash) nunca se podrá saber cuáles han sido los datos insertados, lo que indica que es una función unidireccional, por ese motivo no se utilizan para cifrar mensajes.

2. **Explique conceptualmente la utilidad de algoritmos de hash para:**
   a. **Autenticación de usuarios**
      i. Autenticación de entidades: por ejemplo, es frecuente el uso para este propósito de funciones resumen deterministas con clave secreta que tienen ciertas propiedades (Códigos de autenticación de mensajes). En estos esquemas tanto el servicio de autenticación, o verificador, como la entidad que se quiere autenticar mantienen en secreto la clave de la función resumen. El esquema funciona de la siguiente forma: El que se quiere autenticar genera un mensaje y calcula su valor resumen. Estos dos datos se mandan al verificador. El verificador comprueba que el valor resumen se corresponde con el mensaje enviado y de esta forma verifica que la entidad tiene la clave secreta y por otra parte puede asegurar que el mensaje es íntegro (no ha sido modificado desde que se calculó el valor resumen). Observar que el esquema no tiene la propiedad del no-repudio por parte del que se quiere autenticar ya que el verificador, al disponer de la clave secreta, puede generar también los valores resumen.
      ii. Protección de claves: para comprobar la corrección de una clave no es necesario tener la clave almacenada, lo que puede ser aprovechado para que alguien no autorizado acceda a ella, sino almacenar el valor resumen resultante de aplicar una función resumen determinista. De esta forma para verificar si una clave es correcta basta con aplicar la función resumen y verificar si el resultado coincide con el que tenemos almacenado. Si además queremos que la contraseña sea de un solo uso podemos usar cadenas de resumen como en S/KEY
      iii. Derivación de claves: por ejemplo, en algunas aplicaciones se usan funciones resumen para derivar una clave de sesión a partir de un número de transacción y una clave maestra. Otro ejemplo de aplicación sería el uso de funciones resumen para conseguir sistemas de autenticación con claves de un solo uso o OTP (del inglés One Time Password). En este tipo de sistemas la clave es válida para un solo uso. Estos sistemas se basan en tener un semilla inicial y luego ir generando claves (mediante un algoritmo que puede usar funciones resumen) que pueden tener un solo uso y así evitar ataques de REPLAY.
   b. **Comprobación de integridad de archivos.**

i. En la firma digital
    1. Como dato que se firma: en los algoritmos de firma convencionales normalmente en lugar de firmar todo el contenido se suele ser firmar solo el valor hash del mismo. Algunas de las motivaciones para hacer esto son:
        a. Cuando se usa para firmar algoritmos de firma por bloques donde los mensajes son más largos que el bloque, no es seguro firmar mensajes bloque a bloque ya que un enemigo podría borrar bloques del mensaje firmado o insertar bloques de su elección en el mensaje antes de que sea firmado. Al usar una función hash hacemos una transformación que hace a la firma dependiente de todas las partes del mensaje.
        b. Normalmente los valores hash son mucho más cortos que los datos originales de entrada. Se puede mejorar mucho la velocidad de firma firmando el valor hash en lugar de firmar el dato original
        c. Si los mensajes a firmar pueden tener cierta estructura algebraica y el algoritmo de firma se comporta de forma que el sistema resultante puede ser vulnerable a criptoanálisis con ataques de texto escogido, podemos usar funciones hash para destruir esta estructura algebraica.
    2. Como parte del algoritmo de firma: se han desarrollado algoritmos de firma que usan funciones hash en el propio algoritmo de firma como una herramienta interna del mismo. Ejemplo de este tipo algoritmos son el esquema de firma de Merkle.

ii. Suma de verificación (del inglés checksum): cuando se quiere almacenar o transmitir información, para proteger frente a errores fortuitos en el almacenamiento o transmisión, es útil acompañar a los datos de valores hash obtenidos a partir de ellos aplicando funciones hash con ciertas propiedades de forma que puedan ser usados para verificar hasta cierto punto el propio dato. Al valor hash se le llama Suma de verificación.

iii. Prueba de la integridad de contenidos: por ejemplo, cuando se distribuye un contenido por la red, y se quiere estar seguro de que lo que le llega al receptor es lo que se está emitiendo, se proporciona un valor resumen del contenido de forma que ese valor tenga que obtenerse al aplicar la función resumen sobre el contenido distribuido asegurando así la integridad. A esto se le suele llamar checksum criptográfico debido a que es un checksum que requiere el uso de funciones resumen criptográficas para que sea difícil generar otros ficheros falso que tengan el mismo valor resumen. Otro ejemplo de uso de esta tecnología para verificar la integridad es calcular y guardar el valor resumen de archivos para poder verificar

posteriormente que nada (Ej un virus) los haya modificado. Si en lugar de verificar la integridad de un solo contenido lo que se quiere es verificar la integridad de un conjunto de elementos, se pueden usar algoritmos basados en funciones resumen como los árboles de Merkle que se basan en aplicar reiteradamente las funciones resumen sobre los elementos del conjunto y sobre los valores resumen resultantes.

3. **¿Qué es salt? ¿Para qué se utiliza?**

   Comprende bits aleatorios que se usan como una de las entradas en una función derivadora de claves. La otra entrada es habitualmente una contraseña. La salida de la función derivada de claves se almacena como la versión cifrada de la contraseña. La función de derivación de claves generalmente usa una función hash.

   Para mayor seguridad, el valor de sal se guarda en secreto, separado de la base de datos de contraseñas. Esto aporta una gran ventaja cuando la base de datos es robada, pero la sal no.

   También el beneficio aportado por usar una contraseña con sal es que un ataque simple de diccionario contra los valores cifrados es impracticable si la sal es lo suficientemente larga.