



¿Por qué suenan las campanas?

Desde hace unos meses hay cierto movimiento en el mundillo criptográfico internacional y, en particular, en todo lo que esté relacionado con las denominadas funciones *hash*. El origen de tal nerviosismo y excitación hay que buscarlo en la fecha del 17 de agosto de 2004, en la ciudad de Santa Bárbara, California, en cuya universidad, bajo los auspicios de la conferencia Crypto'04, un equipo de criptógrafos chinos presentó unos resultados sorprendentes. Estas agitadoras aportaciones eran resultado de unos nuevos métodos de ataque (criptoanalítico) desarrollados pacientemente por ellos^{1,2,3,4}. Estos resultados afectan a funciones *hash* que ya se sabían débiles, como es el caso de las funciones MD4, MD5, y SHA-0, pero aún así mejoraban mucho los resultados ya conocidos. El caso de MD5 es especial ya que, a pesar de ser conocida como una función débil desde hace tiempo, su aplicación para la generación de certificados digitales es todavía bastante común. Estas funciones *hash* todavía se utilizan profusamente para el control de integridad de ejecutables que se distribuyen a través de Internet.

El pasado 13 de febrero, ese mismo equipo de investigación formado por Xiaoyun Wang, Yiqun Lisa Yin, y Hongbo Yu de la Universidad de Shandong en China, anunció nuevos resultados, aunque similares a los anteriores, y que, en esta ocasión, se referían a la función **SHA-1**; función a la que, hasta ese

Un equipo de criptoanalistas de la Universidad de Shandong en China ha publicado, desde el pasado mes de agosto y hasta la fecha de cierre de esta edición, varios comunicados en los que ponían de manifiesto y demostraban poder calcular "colisiones", con mayor o menor esfuerzo, para las funciones hash más importantes utilizadas hoy en día. Esta posibilidad debilita considerablemente el valor criptográfico de esas funciones y de todos aquellos sistemas de seguridad que las utilicen. Es hora de revisar con mucho cuidado cuál es el impacto real de estas novedades sobre la seguridad de nuestros sistemas, independientemente de cuáles sean éstos, ya que la amenaza es universal.

momento, se creía capaz de resistir los nuevos tipos de ataques diferenciales. Mas aún, desde el verano pasado, otros trabajos y desarrollos independientes han lanzado serias dudas sobre el principio de diseño iterativo de las funciones *hash*; principio de Damgård-Merkle que, por otra parte, esta prácticamente en todas las funciones *hash* hoy conocidas^{5,6}.

Con sus ancestros prácticamente fuera de juego, y empezando por su aparentemente insano diseño iterativo, ¿cuánto tiempo nos queda antes de que también las funciones SHA-224/256/384/512 resulten afectadas?

Una función *hash* es aquella que relaciona cualquier mensaje o vector binario con un valor, también binario, de longitud constante (n-bits). Estas funciones son diseñadas de tal modo que sean fáciles de calcular en un sentido (del mensaje al valor *hash*), a la vez que son computacionalmente imposibles de calcular en el sentido contrario. Por este motivo se habla de ellas como de "funciones de sentido único" y son de una importancia esencial en prácticamente todos los protocolos criptográficos en uso. Toda función *hash* está

esencialmente sometida a tres grandes tipos de ataques que definen su seguridad:

Resistencia al cálculo de pre-imágenes: dado un valor de salida $y = h(x)$, pero no el correspondiente valor x de entrada, debe ser prácticamente imposible encontrar x .

Resistencia al cálculo de segundas pre-imágenes: dado un valor de salida

$y = h(x)$ y el correspondiente valor x de entrada, debe ser prácticamente imposible encontrar otra entrada $z \neq x$ tal que las salidas sean iguales $h(z) = h(x)$.

Resistencia al cálculo de colisiones: debe ser prácticamente imposible encontrar cualquier par de valores de entrada, x y z tales que sus valores *hash* de salida sean el mismo $h(x) = h(z)$.

Esta última propiedad es la más difícil de satisfacer y es absolutamente necesaria para proteger a muchos de los esquemas de firma digital de las falsificaciones. En tales es-

quemas, el valor *hash* actúa como representación de todos los bits del mensaje que se firma; así, si un atacante puede encontrar dos entradas x y z que colisionan para una función *hash*, entonces el atacante es capaz de reutilizar una firma legítima y correcta de x , como una firma de z .

Perspectiva histórica

Aunque hay funciones *hash* que se remontan hasta la década de los 80, las que ahora nos preocupan aparecen en la década de los 90. En 1990 Ron Rivest inventa, después de un curioso intento anterior conocido como MD2, la función *hash* que se conoce como MD4. En 1992, ese mismo autor mejora el algoritmo anterior y con ello desarrolla otra función que llama MD5. En 1993 la National Security Agency (NSA) americana, a través del National Institute of Technology and Standards (NIST), publica una función muy similar a la MD5, y la llama SHA. Un poco más tarde, en 1995, alegando una debilidad recién descubierta pero no publicada, la propia NSA hace cambios en la definición de la función SHA, y el nuevo algoritmo pasa a llamarse SHA-1. Hoy en día, las funciones *hash* más populares son la **SHA-1** y la **MD5**, siendo esta última la elección por defecto en muchas aplicaciones y desarrollos ampliamente difundidos.

Desde el pasado 13 de febrero, el equipo de la Universidad de Shandong ha hecho circular, con discreción,

una nota en la que se reúnen sus principales resultados y en la que sus autores aseguran poder generar colisiones para la versión completa (80-etapas) de la función **SHA-1** en **2⁶⁹ operaciones hash**. Esta cantidad es **2.048 veces menor** que la propia de los inevitables ataques por fuerza bruta que requirieron **2⁸⁰ operaciones** para un *hash* de 160-bits (Paradoja del cumpleaños). En ese anuncio, además de la anterior aseveración, también se dan ejemplos de colisiones para la función **SHA-0** completa que se han obtenido con un coste computacional de **2³⁹ operaciones**. También se da otro ejemplo de colisión para una versión reducida a 58-etapas, de la función **SHA-1** y con un coste computacional asociado de **2³³ operaciones**. El algoritmo seguido por el equipo chino para conseguir estos ejemplos **no ha sido publicado todavía**, y habrá que esperar hasta la conferencia Eurocrypt'05 del próximo mes de mayo, para conocerlo con algún detalle.

Aunque la complejidad y coste computacional para atacar la función **SHA-1** pueda hacer pensar que no tiene aplicaciones prácticas con los ordenadores de hoy en día, conviene siempre recordar que en el año 1999, un grupo de investigadores construyeron una máquina muy especializada que denominaron "**DES cracker**". Esta máquina fue capaz entonces de realizar **2⁵⁶ operaciones DES** en sólo 56 horas. El coste total de la máquina fue de unos 250 mil dólares, aunque podrían hacerse copias suyas por sólo 50-75 mil dólares. Extrapolando las cualidades de esa máquina según la **ley de Moore**, una máquina similar a esa, hoy podría realizar **2⁶⁰ cálculos en poco más de dos días**, y **2⁶⁹ cálculos en tres años y tres meses**. El coste de una máquina que

realizase esos **2⁶⁹ cálculos** en esas mismas 56 horas, hoy costaría entre 20 y 30 millones de euros. Si esta cantidad puede parecer mucho dinero, también hay que recordar que no hace mucho tiempo, una iniciativa de computación altruista y distribuida en Internet logró romper un reto construido sobre el algoritmo de cifrado RC5 que suponía un coste computacional de **2⁶⁴ operaciones**. Ese reto requirió algo menos de **6 años** de cooperación entre algunos miles de ordenadores, lo cual puede no ser mucho dependiendo de los beneficios que se consigan con el ataque. A la vista de este resultado conviene recordar que un coste computacional de **2⁶⁹ operaciones** sólo es 32 veces más.

A la vista de estos resultados ¿qué deberíamos hacer en la práctica? Para aquellas aplicaciones que utilicen las funciones hash afectadas, deberemos considerar con cuidado si los escenarios de ataque mediante colisiones aleatorias pueden suponer algún perjuicio para nuestro sistema.

Además de saber si hay riesgo o no al utilizar la función *hash* **SHA-1** y sus **2⁶⁹ operaciones** como nuevo nivel de seguridad máxima, no hay que olvidar que el anuncio de agosto de 2004 pone de manifiesto la **posibilidad de obtener colisiones prácticamente "a la carta"** para un amplio conjunto de funciones *hash* (**MD4, MD5, HAVAL-128, RIPEMD**). El equipo de Wang todavía mantiene en secreto las técnicas utilizadas; sin embargo, durante el otoño del año pasado, algún que otro equipo de criptoanalistas ha tratado de reconstruir esa metodología y en esas investigaciones se han mejorado los tiempos de cálculo. Según los resultados conocidos, esas mejoras han llegando a permitir calcular

una colisión para la función **MD5** en sólo **8 horas** utilizando un **ordenador doméstico** (Intel 1,6 GHz).

El pasado día 1 de marzo, Arjen Lenstra de los Bell Laboratories, Xiaoyun Wang, de la Universidad Shandong, y Benne de Weger, de la Universidad Técnica de Eindhoven, publicaron un interesante anuncio titulado "**Colliding X.509 Certificates**"⁷. En ese breve artículo, los autores declaran haber descubierto un método que les permite construir pares de certificados X.509 válidos, en los que las partes "a ser firmadas" son una colisión de la función *hash* **MD5**. En estas condiciones, la firma de cualquier autoridad de certificación sobre cualquiera de los dos certificados daría el mismo

resultado, tanto si se utiliza **MD5** como su función *hash*. En este ejemplo, los autores ponen de manifiesto cómo las colisiones de la **función MD5** pueden calcularse y elegirse fácilmente, y como con ello **se destruyen los principios básicos que sostienen la confianza en las infraestructuras de clave pública**. El método de Lenstra permite construir certificados X.509 en los que todos los campos, excepto la clave pública, pueden elegirse de forma arbitraria; las claves públicas RSA que se utilizan son "a la medida", pero resultan seguras.

Gracias a estas técnicas ya disponibles, a partir de ahora no se puede estar seguro de que al firmar y emitir un certificado, un contrato o

cualquier otro objeto digital, no exista otro ejemplar con exactamente la misma firma y con valores completamente desconocidos para el firmante. Por ejemplo, un atacante podría solicitar con sus datos reales de identidad, que se le emita el correspondiente certificado digital para usarlo como su identidad en la red (DNI electrónico) y luego, una vez conseguido el certificado legítimo, el atacante extraería su bloque de firma y lo concatenaría con otro certificado en el que **él ha elegido libremente todos los elementos de su falsa y nueva identidad**. A la hora de las verificaciones, este segundo certificado sería tan válido como el primero.

En una reunión del grupo técnico de trabajo en PKI del NIST, William Burr, director del Security Technology Group dijo que no se había tocado una implementación completa de la función **SHA-1** y que, por ello, "*no podía considerarse rota*" y, añadió, "*que no había muchas razones para sospechar que lo pueda ser pronto*"; después del 13 de febrero ya sabemos que todo eso no es cierto y que la situación ha cambiado significativamente. En esas mismas declaraciones, el señor Burr reconocía la posibilidad de avances en esos ataques y, por ello, le parecía prudente descartar el uso de la función *hash* **SHA-1** a partir del año 2010.

Conclusiones

A la vista de estos resultados ¿qué deberíamos hacer en la práctica? Para aquellas aplicaciones que utilicen las funciones *hash* afectadas, deberemos considerar con cuidado si los escenarios de ataque mediante colisiones aleatorias pueden suponer algún perjuicio para nuestro sistema. Si ese es el caso, lo mejor es proceder a analizar

el riesgo que corremos, y buscar si hay factores o medidas disponibles que puedan mitigar el impacto del riesgo y/o disminuir la probabilidad de éste. Si después del análisis vemos que los niveles son suficientemente bajos, uno puede decidir no hacer nada. En cualquier otro caso deberá iniciarse y completarse el cambio de la función *hash* utilizada por otra más resistente. Lo que sí debería estar claro es que, para aplicaciones nuevas, las funciones *hash* afectadas no deben utilizarse. Por el momento, esto limita el repertorio de posibles sustitutas a las extensiones de la función SHA-1, excluyendo a ésta. El NIST recomienda utilizar SHA-256, y todavía no parece estar muy preocupado por el uso de la función *hash* SHA-1 hasta el año 2010.

Dados estos recientes desarrollos, los diseñadores de protocolos puede que necesiten realizar cambios en las funciones *hash* que utilizan, por lo que la revisión de todos los productos, protocolos y sistemas es de lo más recomendable. Cuando se hayan hecho los necesarios exámenes y se vea qué cambios hay que introducir en las nuevas versiones, siempre habrá que asegurar que está preparado frente a los ataques que pueda dejar pendientes las usuales medidas de "compatibilidad hacia atrás".

Hasta nueva orden, todos los nuevos diseños deberán utilizar SHA-256. Los sistemas existentes y que utilicen SHA-1 o MD5 deberán **confirmar que sólo necesitan resistencia a segundas preimágenes y no resistencia a colisiones aleatorias**. El uso del MD5 en certificados de cualquier tipo debe suspenderse a menos que se tomen controles complementarios adecuados.

Por la naturaleza de los ataques publicados, no hay

la necesidad inmediata de preocuparse por la seguridad de las funciones HMAC que utilicen MD5 o SHA-1. Sin embargo, los avances criptoanalíticos sobre el MD5 invitan seriamente a reemplazar el HMAC-MD5 por el HMAC-SHA-1 o, preferiblemente, por el HMAC-h con alguna función *hash* que todavía es resistente a colisiones.

¿Es este el final de la historia? ¿Deberemos ahora utilizar todos la función SHA-

Está claro que la situación no es grave pero sí conviene tomársela en serio. No hay razones para que la industria no reaccione rápida y cabalmente, y pase a utilizar (temporalmente) funciones como la SHA-256 o, incluso, mantenga durante un (breve) tiempo la función SHA-1

256? o bien ¿deberíamos continuar nuestra marcha hacia delante y **rediseñar completamente nuestros métodos para la construcción de funciones *hash***, y así conseguir alternativas que sean netamente superiores a las actuales? Consideremos la situación con más detalle. Todas las funciones *hash*, incluidas MD4, MD5, y las SHA-0/1/224/256/384/512, siguen la misma aproximación básica, tanto en su diseño iterativo como en el tipo de operaciones y ordenaciones que constituyen sus diferentes "etapas". Las diferencias entre la función SHA-1 y las funciones MD4, MD5, y SHA-0, se reducen a pequeños cambios y, además, las funciones de la serie SHA-224/256/384/512 sólo se separan del diseño de la función SHA-1 en que tienen algunos giros adicionales.

Hasta hace unas pocas semanas, se esperaba que los cambios menores ofrecieran protección adecuada ante los ataques por colisiones que se anunciaron en agosto de 2004. El comunicado de

febrero de 2005 respecto a la función SHA-1 ha defraudado irreversiblemente esas esperanzas. Con sus ancestros prácticamente fuera de juego, y empezando por su aparentemente insano diseño iterativo, ¿cuánto tiempo nos queda antes de que también las funciones SHA-224/256/384/512 resulten afectadas? Adi Shamir, uno de los criptógrafos más respetados, recomienda empezar de nuevo, desde

tría no reaccione rápida y cabalmente, y pase a utilizar (temporalmente) funciones como la SHA-256 o, incluso, mantenga durante un (breve) tiempo la función SHA-1, pero sí hay que recordar que no se puede bajar la guardia y que estos nuevos métodos criptográficos pueden muy bien amenazar a todo un paradigma de diseño de funciones *hash*. Me temo que es hora de realizar esfuerzos para el diseño público y contrastado de nuevas primitivas criptográficas. A cualquiera se le alcanza aquella máxima de que "los ataques criptográficos, con el paso del tiempo, sólo pueden mejorar".

Como dijo Jon Callas de PGP Corporation cuando le pidieron su opinión sobre estas novedades: "Es momento de dirigirse, sin correr, hacia las salidas de emergencia. No vemos el humo, pero las alarmas de incendio han saltado". ■

JORGE DÁVILA MURO
Director
Laboratorio de Criptografía
**LSIIS – Facultad
de Informática – UPM**
jdavila@fi.upm.es

REFERENCIAS

- 1 Wang, F. Guo, X. Lai, H. Yu, "Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD", <http://eprint.iacr.org/2004/199>, presentado en Crypto 2004 el 17 de agosto de 2004.
- 2 Wang, H. Yu, "How to Break MD5 and Other Hash Functions", a presentar en la conferencia Eurocrypt 2005 en mayo de 2005.
- 3 Wang, X. Lai, F. Guo, H. Chen, X. Yu, "Cryptanalysis for Hash Functions MD4 and RIPEMD", a presentar en la conferencia Eurocrypt 2005 en mayo de 2005.
- 4 Joux, A.: "Collisions for SHA-0" presentado en la rump session de Crypto 2004 el 17 de agosto de 2004.
- 5 Joux, A.: "Multicollisions in iterated hash functions. Application to cascaded constructions". Proceedings Crypto 2004, Springer-Verlag LNCS 3152 (2004) pp. 306-316.
- 6 Kelsey, J.; Schneier, B.: "Second preimages on n -bit hash functions for much less than 2^n work", <http://eprint.iacr.org/2004/304>; a presentar en la conferencia Eurocrypt 2005 en mayo de 2005.
- 7 Lenstra, A.; Wang, X.; de Weger, Benne: "Colliding X.509 Certificates", Cryptology ePrint Archive, Report 2005/067, <http://eprint.iacr.org/2005/067>