[1]

OTRA MIRADA A"SEGURIDAD DEMOSTRABLE". II

NEAL KOBLITZ AND ALFRED J. MENEZES

Resumen. Discutimos el problema de cómo interpretar los argumentos reduccionistas en criptografía. Se presentan varios ejemplos que demuestran las sutilezas y dificultades de tal pregunta.

1. Introducción

Suponga que se desea tener una confianza razonable en la seguridad de cierto protocolo criptográfico. En el paradigma de "seguridad demostrable", el escenario ideal se da cuando se cuenta con una reducción estrecha o apretada (Véase §4 para una definición y discusión de apretadura), de un problema matemático del cual existe consenso acerca de la dificultad e infactibilidad computacional que presenta a un ataque especifico al protocolo, donde tal ataque ha sido definido previamente. Esto significa que un adversario que pueda atacar al sistema deberá también ser capaz de resolver el problema (supuestamente intratable) con esencialmente el mismo esfuerzo computacional y con la misma probabilidad de éxito. Frecuentemente, sin embargo, lo mejor que los investigadores han sido capaces de alcanzar dista bastante de ese escenario ideal. Algunas veces los argumentos reduccionistas de seguridad han sido hallados para versiones modificadas del protocolo en cuestión, pero no para el protocolo original que es utilizado en la práctica; o para una versión modificada del tipo de ataque examinado, pero no para la definición de seguridad que realmente se busca; o los argumentos reduccionistas aplican a una versión artificialmente modificada y cuidadosamente planificada del problema matemático que es considerado intratable, pero que sin esas modificaciones no tienen fundamento en el problema difícil que tan meticulosamente ha sido estudiado. En otros casos, se sabe de resultados asintóticos que no pueden ser aplicados con parámetros específicos sin hacer un análisis más profundo. En otros casos más, aunque se tiene una reducción, se puede demostrar que no puede haber (o que es improbable que haya) una reducción apretada. En este artículo se dan varios ejemplos que muestran las sutiles cuestiones que surgen cuando se interpretan argumentos reduccionistas en criptografía.

---

[1]Traducido por Francisco Rodríguez-Henríquez

Date: July 3, 2006.

Key words and phrases. Cryptografía, Clave Pública, Seguridad Demostrable.

2. Equivalencia pero sin Prueba Reduccionista

En [13], Boneh y Venkatesan mostraron que una reducción eficiente de factorización al problema RSA (es decir, el problema de invertir la función $y = x^e \bmod N$) muy probablemente no existe. Más precisamente, dichos autores probaron que para un exponente pequeño de cifrado e, la existencia de una reducción algebraica eficiente implicaría que la factorización entera es un problema fácil.

El artículo [13] fue publicado al tiempo que se gestaba una intensa rivalidad entre RSA y Criptografía de Curvas Elípticas (CCE). Como entusiastas incondicionales que somos de este último, nos encantó conocer el resultado Boneh-Venkatesan y nos congratulamos de su interpretación —esa que, usando las mismas palabras del título del artículo diría que: "Romper RSA podría no ser Equivalente a la Factorización Entera"[1]— como un clavo más en el ataud de RSA.

Sin embargo, para ser totalmente honestos, una segunda interpretación es por lo menos igual de plausible. Tanto el problema de factorización como el problema RSA han sido estudiados intensivamente por muchos años. Para el caso general, nadie tiene idea de cómo resolver el problema de RSA sin factorizar el módulo. Así como nuestra experiencia nos indica con fuerza que factorizar (y ciertos otros problemas tales como el problema del logaritmo discreto de curvas elípticas) es un problema difícil, así también tenemos excelentes razones para pensar que en la praxis, el problema RSA *es* equivalente a factorizar. Así, una interpretación alternativa del resultado Boneh-Venkatesan es que demuestra el muy limitado valor de los argumentos reduccionistas, de tal manera que un título alternativo al artículo [13] podría haber sido "La Ausencia de una reducción entre dos problemas Podría no Indicar Inequivalencia".

Por cuál de estas dos interpretaciones se inclina alguien es esencialmente una cuestión de opinión, pues esa preferencia puede estar influenciada, como en nuestro caso, por los juicios que personalmente se tengan, ya sea a favor o en contra, sobre RSA.

3. Resultados que apuntan en Direcciones Opuestas

3.1. Boneh-Venkatesan en Reversa. Un resultado reciente [16] de D. Brown puede ser interpretado como un argumento de apoyo a la explicación alternativa de Boneh-Venkatesan, descrita al final de §2. Para exponentes de cifrado pequeños e,[2] Brown probó que si existe un programa eficiente que dado el módulo N de RSA pueda ser capaz de construir un *programa de línea recta*[3] que resuelva eficientemente el problema RSA,[4] entonces ese programa

[1] "breaking RSA may not be equivalent to factoring" en su título original [N. del T.]

[2] En realidad, el resultado de Brown aplica sólo si e tiene un factor primo pequeño.

[3] straight line program

[4] Ello esencialmente significa que construye un polinomio que invierte la función de cifrado.

puede ser usado también para factorizar eficientemente N. Esto sugiere que para e pequeño, el problema RSA podría muy bien ser equivalente a factorizar. Si uno cree en esta interpretación, entonces se podría concluir que e pequeños son intrínsecamente más seguros que e grandes. En contraste, el resultado Boneh-Venkatesan puede ser interpretado como una sugerencia que valores grandes de e son más seguros que los pequeños.

Tal como el propio Brown señala en §5 de [16], en realidad su resultado no contradice a Boneh-Venkatesan. Su reducción de factorizar a un *programa de línea recta* para hallar raíces e-ésimas no satisface las condiciones de reducción en [13], pues su uso del extractor de la raíz e-ésima no puede ser modelado por un oráculo RSA tal como se estipula en [13]. Esto se debe a que Brown aplica su programa de línea recta a extensiones de anillo de $\mathbb{Z}/N\mathbb{Z}$.[5]

La elección del título del artículo de Brown es especialmente sugestiva: "Romper RSA podría ser tan difícil como Factorizar",[6] todo lo que se tiene que hacer es colocarlo con una disjunción al lado del título de [13], y uno se queda con una declaración inequívoca y que sumariza con precisión lo que se conoce respecto a este tema de investigación.

3.2. ¿Relleno aleatorio antes o después de la función picadillo? Cuando se comparan los esquemas de firma tipo ElGamal, se encuentra que algunos, tales como las firmas Schnorr [35], agregan al mensaje una cadena aleatoria antes de evaluar la función picadillo (*hash* por su nombre en inglés); mientras que otros, tales como el Algoritmo de Firma Digital (DSA por sus siglas en inglés) y el Algoritmo de Firma Digital de Curvas Elípticas (ECDSA por sus siglas en inglés), aplican la función picadillo antes del relleno aleatorio. La pregunta es: ¿Es más seguro hacer el relleno antes o después de aplicar la función picadillo?, y ¿Qué nos dicen los resultados disponibles de "Seguridad Demostrable" acerca de esta pregunta?

Como discutimos en §5.2 de [27], la prueba que falsificar las firmas Schnorr es equivalente a resolver el problema del logaritmo discreto (el bosquejo de esta prueba puede ser consultado en §5.1 de [27] y §8.3 abajo, para una prueba más formal consúltese [33,34]), está esencialmente basada en el hecho que un oponente puede escoger el aleatorio r antes de solicitar la ejecución de la función picadillo. Por esta razón, esa prueba no aplica a DSA, donde únicamente el mensaje m y no r es sometido a la función picadillo. En §5.2 de [27] comentamos que:

> ... El reemplazar $H(m;r)$ por $H(m)$ potencialmente beneficia al falsificador, quien tiene entonces control sobre la elección de k (la cual determina r) pero ningún control sobre el valor del picadillo (que es esencialmente aleatorio). Si H depende

[5] Por ejemplo, cuando $e = 3$, el polinomio que invierte raíces cúbicas se aplica al anillo: $\mathbb{Z}/N\mathbb{Z}[X]/(X^2 - u)$, donde el símbolo de Jacobi está dado como $(\frac{u}{N}) = -1$.

[6] "Breaking RSA may be as difficult as factoring", en su título original [N. del T.].

> tanto de r como de m, la elección de k por parte del falsificador debe hacerse antes de determinar el valor del picadillo, así que el falsificador no tenga la última palabra

Ese fue nuestro intento de dar una explicación intuitiva del hecho que el modelo del oráculo aleatorio de las firmas Schnorr, a diferencia de las muy similares firmas DSA, ha podido ser asociado al problema del logaritmo discreto (PLD) a través de un argumento reduccionista. A partir de nuestro comentario se concluiría que es más seguro agregar el relleno antes de invocar a la función picadillo.

Sin embargo, cometimos un grave error al contribuir a confundir al lector de esa manera. De hecho, existe otro resultado de seguridad demostrable encontrado por D. Brown en [14, 15], el cual apunta en la dirección contraria. Dice que: Si la función picadillo y el generador de bits pseudo-aleatorios satisfacen ciertas suposiciones razonables, entonces ECDSA es seguro ante el ataque del mensaje-escogido (*chosen-message attack*) realizado por un falsificador universal[7], siempre que el "Problema adaptivo del semi-logaritmo" en el grupo de curvas elípticas sea un problema difícil. [8] Brown comenta en [15] que su seguridad reduccionista no funcionaría para una modificación de ECDSA en la cual tanto r como el mensaje m fueran sometidos a la función picadillo.

Brown no afirma que la versión modificada sea menos segura que la versión original de ECDSA en la cual sólo el mensaje es sometido a la función picadillo. Sin embargo, en una comunicación informal [17], él nos explicó cómo alguien podría afirmar tal cosa: la inclusión del aleatorio r junto con m a la entrada de la función picadillo podría ser vista como "Dar al oponente espacio de maniobra extra con dicha función," y esto podría conducir a un hueco de seguridad (Note que tanto los resultados en [33,34] como en [14,15] suponen que la función picadillo es fuerte.)

Una vez más tenemos que los resultados de la seguridad demostrable sugieren respuestas opuestas a una cuestión muy simple y mundana, esto es, ¿Es mejor agregar el relleno antes o después de evaluar la función picadillo? Tal y como pasó en el caso de la pregunta en §3.1, ambas respuestas: "antes" y "después", pueden ser respaldadas con argumentos reduccionistas.

En §8 discutiremos otra pregunta -de si la falsificación de firmas tipo Schnorr es o no equivalente al PLD- para la cual diferentes resultados de la seguridad demostrable brindan evidencia que conducen a respuestas opuestas.

---

[7] Un falsificador es universal (o *selectivo* en la terminología de Brown) si puede falsificar cualquier mensaje m que le sea dado.

[8] El semi-logaritmo de un punto Q con respecto al punto base P de orden primo p es la pareja de enteros $(t; u)$ mód p tal que $t = f(u^{-1}(P + tQ))$, donde la "Función de conversión" f es el mapa desde puntos a enteros módp que se utiliza en ECDSA. El problema adaptivo del semi-logaritmo consiste en el problema de hallar un semi-logaritmo de Q al punto base P dado un oráculo que puede hallar un semi-logaritmo de Q para cualquier base de la forma eP con $e \neq 1$.

4. De la Holgura en las Reducciones

Iniciamos esta sección dando una definición informal de reducción estrecha o apretada. Suponga que se cuenta con un algoritmo que resuelve el problema $\mathcal{A}$ en un tiempo no mayor a T segundos y que es exitoso para una proporción de al menos ϵ instancias de $\mathcal{A}$, donde T y ϵ son parámetros que dependen de la longitud de entrada. Una reducción desde el problema $\mathcal{B}$ al problema $\mathcal{A}$, es un algoritmo que tras invocar un cierto número de veces el algoritmo que resuelve el problema A, es capaz de resolver el problema $\mathcal{B}$ en un tiempo T' para una proporción de por lo menos ϵ' de las instancias de $\mathcal{B}$. Se dice que ésta reducción es apretada si $T \approx T'$ y $\epsilon \approx \epsilon'$. De manera general, la reducción es holgada si $T' \gg T$ o si $\epsilon' \ll \epsilon$.

Suponga ahora que hemos sido capaces de obtener una muy holgada reducción desde un problema matemático difícil al rompimiento de algún protocolo. Enfrentados a esta situación, se observan diversas reacciones típicas, a saber:

1. Incluso una reducción holgada es mejor que nada. Uno debería ver un vaso medio lleno en vez de verlo medio vacío. Intente derivar algún tipo de seguridad con lo que se cuenta, y trate de no pensar demasiado en lo que nos gustaría tener.[9]
2. Aun cuando la reducción no sea apretada es razonable pensar que en el futuro se hallará alguna que sí lo sea.
3. Posiblemente es imposible hallar una reducción apretada para el protocolo en cuestión, pero quizás se pueda hacer una delicada modificación al protocolo original, que permita la construcción de una reducción apretada (donde, claramente, debemos suponer que esta reducción es un mecanismo que nos permite derivar pruebas de seguridad en el protocolo original).
4. Una reducción apretada quizás pueda ser construida si se relaja el problema matemático difícil (por ejemplo, reemplazando el problema computacional Diffie-Hellman con el problema de decisión de Diffie-Hellman).
5. Quizás la definición de seguridad es demasiado estricta, así que uno debería relajarla un poco para permitir la construcción de una reducción apretada.
6. Acaso el protocolo sí es seguro en la práctica, aun cuando su tan deseada reducción apretada podría sencillamente no existir.

[9] Lo que nos recuerda los versos de aquella canción otrora popular:
If you can't be with the one you love,
Love the one you're with,
(*Stephen Stills*, 1970) (*("Si no puedes estar con la persona que amas,|| ama a la persona con la que estás")*. La versión para criptográfos sería:
"Si no puedes probar lo que morirías por probar,
Pretende ser feliz con lo que sea que logres demostrar".

7. Tal vez el protocolo es en efecto inseguro pero ningún ataque ha sido descubierto, todavía.

Estos siete puntos de vista no son mutuamente exclusivos. De hecho, los diseñadores de protocolos normalmente adoptan alguna combinación de las primeras seis interpretaciones, aunque generalmente no la séptima.

4.1. Inseguro pero demostrablemente seguro: Un ejemplo. Damos ahora un ejemplo que, lo admitimos, es un tanto artificial. Abordemos una máquina del tiempo y vayamos 25 años atrás, en una época en que la versión *naive*[10] del cálculo indexado era esencialmente el mejor algoritmo de factorización. Supongamos también que el cómputo de 2^{2a} operaciones es computacionalmente factible, pero que el cómputo de $2^{(2\sqrt{(2)})a}$ operaciones no lo es.

Sea N un módulo RSA de c bits, y sea r un entero con a bits. Sea $F = \{p_1, \ldots, p_r\}$ una base constituida por los primeros r primos. Sea 2^b el tiempo esperado necesario para encontrar un residuo aleatorio x mód N con la propiedad que su residuo cuadrático x^2 mód N, es p_r-suave (lo cual significa que no tiene factores primos mayores que p_r). La estimación típica es que $2^b \approx u^u$, donde $u = c/a$. (en realidad, más precisamente $u = c/(a + log(aln2))$, donde log denota log_2, pero permítasenos ignorar términos de segundo orden.)

Si x tiene la propiedad que x^2 mód N es p_r-suave, entonces entenderemos que su *vector exponente*, es el vector en $\mathbb{F}_2^r$ cuyos componentes ϵ_i son los exponentes de p_i de la porción libre de cuadrados de x^2 mód N.

La versión simple (*naive*) del algoritmo de cálculo indexado implica generar alrededor de r residuos aleatorios x_i para después resolver una matriz $r \times r$ sobre $\mathbb{F}_2$. La primera parte toma aproximadamente $r2^b \approx 2^{a+b}$ operaciones, mientras que la segunda parte toma aproximadamente 2^{2a} operaciones. Así que típicamente se escoge $b \approx a$. Sin embargo, en nuestro protocolo, con el objeto de ser capaces de dar una prueba de seguridad, optimizaremos de una manera distinta, tomando $b \approx 2a$.

Note que para el valor constante c, el valor de a elegido como $b \approx 2a$, es diferente del valor óptimo a' que se escoge para factorizar N. En el primer caso se hace $2^{2a} \approx u^u$, (donde $u = c/a$, esto es, $2^a \approx \frac{c}{a} log u$); mientras que en el segundo se hace $a' \approx \frac{c}{a'} log u'$, (donde $u' = c/a'$). Puesto que u' es del mismo orden de magnitud que u, al dividir estas dos ecuaciones se obtiene aproximadamente, $a' \approx \sqrt{2}a$, lo cual nos lleva a un estimado de $2^{(2\sqrt{(2)})a}$ operaciones necesarias para factorizar N.

Habiendo hecho esas definiciones, procedemos ahora a describir nuestro protocolo. Suponga que Alicia quiere probar su identidad a Betito, esto es, quiere probar que ella conoce los factores de su módulo de dominio público N. Betito le envía un reto el cual consiste de s vectores linealmente independientes en $\mathbb{F}_2^r$, con $0 \leq s \leq r-1$. Alicia debe responder con una x tal que

[10] ingenua en francés e inglés, [N. del T.]

x^2 mód N es p_r-suave y tal que su vector exponente no está en el subespacio vectorial S definido por los vectores que Betito incluyó como parte de su reto. (La idea es prevenir que potenciales impostores puedan dar una solución correcta combinando respuestas previas de Alicia; de esta manera, en la práctica, Betito se aseguraría de incluir en su reto los vectores-exponentes que Alicia haya transmitido en comunicaciones anteriores.) Alicia puede responder el reto rápidamente debido a que es fácil para ella computar raíces cuadradas módulo N dado que conoce los factores de N.

Ahora reduciremos el problema de factorización al ataque de que un impostor personifique a Alicia. Sea IO un oráculo capaz de personificar a Alicia. Para factorizar N hacemos r invocaciones a IO (asegurando que cada nuevo reto incluya todos los vectores-exponente emitidos por el oráculo en comunicaciones previas), para obtener un conjunto de relaciones cuyos vectores-exponentes definen el espacio vectorial $\mathbb{F}_2^r$. Una vez hecho esto, sólo nos resta encontrar k residuos x's aleatoreamente generados, con residuo cuadrático x^2 mód N p_r-suave, para garantizar con una probabilidad de $1 - 2^{-k}$ que sí podremos factorizar N. Encontrar tales x's toma alrededor de $k2^b$ unidades de tiempo. Puesto que tenemos que resolver una matriz cada vez, en realidad el tiempo de cómputo es $k(2^b + 2^{2a})$. Si una llamada a IO toma un promedio de T segundos, entonces el tiempo total para factorizar N es $T' \approx k(2^b + 2^{2a}) + rT \approx k2^{2a+1} + 2^aT$, donde $b = 2a$ y $r \approx 2^a$. Puesto que estamos suponiendo que factorizar N requiere $2^{(2\sqrt{(2))a}}$ operaciones, se obtiene la cota no trivial inferior $T \geq 2^{(2\sqrt{(2)-1)a}}$. Siempre que se logra encontrar de manera formal una cota inferior para el tiempo de ejecución de un adversario, ésta es una cota altamente no trivial y que, aunque posiblemente mucho menor que lo que idealmente se desearía, muchas veces se acerca a los límites de la factibilidad computacional. Tal resultado puede ser visto como una prueba de seguridad (véase el comentario 2 más abajo).

A pesar de todo, el protócolo descrito es inseguro puesto que puede ser roto en un tiempo $2^b = 2^{2a}$.

Este ejemplo no es realista no sólo porque supone que el método naive del cálculo indexado es el mejor método de factorización sino también porque desde el principio debió haber sido bastante obvio que el protocolo es inseguro. En consecuencia, planteamos el siguiente problema abierto:

Problema. Encontrar un ejemplo de un protocolo natural y realista que tenga una prueba reduccionista (holgada) de seguridad, pero que al mismo tiempo sea inseguro cuando se utiliza con parámetros de un tamaño aceptable.

Comentario 1. Cualquiera sea el resultado de este problema, su solución será de interés. Si alguien encuentra un protocolo con seguridad demostrable (no apretada) pero que al mismo tiempo es inseguro, entonces quedará más claro que nunca qué tan crucial es, para la seguridad demostrable, la búsqueda de reducciones apretadas. Por otro lado, si después de mucho esfuerzo, es

imposible encontrar un protocolo con esas características, entonces los desarrolladores podrán sentirse en libertad de ignorar la necesidad de obtener reducciones apretadas.

Comentario 2. Es importante remarcar que un ejemplo de este tipo ya ha sido hallado en clave simétrica, en el contexto de códigos de autenticación de mensaje (MAC por sus siglas en inglés). En [18] Cary y Venkatesan presentaron un esquema MAC para el cual derivaron una prueba de seguridad (incidentalmente, no se trataba de una prueba reduccionista). Su esquema dependía de un parámetro l, y para el valor práctico $l = 32$, su prueba demostraba que ninguna colisión podía ocurrir sin hacer al menos 2^{27} preguntas al oráculo. Aun y cuando este cálculo se queda bastante corto del nivel de seguridad requerido para bloques MAC -es decir, 64 bits de seguridad-, esta prueba fue interpretada como un argumento alentador sobre la seguridad que dicho esquema podía brindar. Sin embargo, en [8] Blackburn y Paterson idearon un ataque capaz de encontrar colisiones tras hacer $2^{48,5}$ peticiones MAC y capaz de hacer falsificaciones tras 2^{55} preguntas al oráculo. Este ejemplo muestra que se deben tomar en serio las garantías numéricas de seguridad que una prueba brinda, pues si no, podríamos toparnos con un sistema que es demostrablemente seguro y también inseguro.

4.2. Resultado Coron para firmas RSA. Discutiremos primero el esquema básico de firma RSA con una función picadillo de dominio completo. Suponga que la usuaria Alicia con clave pública (N, e) y exponente privado d, desea firmar un mensaje m. Para ello, Alicia aplica primero una función picadillo $H(m)$, la cual toma valores en el intervalo $0 \leq H(m) < N$, seguido por el cálculo de su firma mediante $s = H(m)^d$ mód N.

Cuando Betito recibe el mensaje m y la firma s, él verifica la firma calculando $H(m)$, seguido por s^e mód N. Si estos dos valores son iguales, Betito queda convencido que el mensaje fue enviado por Alicia (dado que Betito presume que Alicia es la única persona que conoce el exponente d que invierte la exponenciación $s \mapsto s^e$) y que el mensaje es una copia genuina del original (debido a que Betito presume que cualquier otro mensaje tendría un picadillo diferente).

Describimos a continuación un argumento clásico en seguridad reduccionista para el esquema de firma RSA [6]:

Afirmación de Seguridad Reduccionista. Si el problema de invertir $x \mapsto x^e$ mód N es infactible, entonces la firma RSA con dominio completo de la función picadillo es seguro en el modelo del oráculo aleatorio bajo el ataque de mensaje escogido[11], instrumentado por un falsificador existencial.
Argumento. Suponga que dado un entero arbitrario y en el intervalo $0 \leq y < N$, se nos pide encontrar x tal que $y = x^e$ mód N. La afirmación de

[11] *Chosen message attack*, [N. del T.]

seguridad reduccionista surge de mostrar cómo se podría (con alta probabilidad) encontrar x dado que un falsificador ha montado el ataque de mensaje escogido.

Suponga que tal falsificador está activo y que se le da la clave pública de Alicia (N, e) para que se disponga a formular peticiones (*queries*) de picadillo y/o de firma. En todos los casos excepto uno, una petición de picadillo para un mensaje m_i escogido aleatoriamente, es respondida de la siguiente manera. Se selecciona aleatoriamente $x_i \in \{0, 1, \ldots, N-1\}$ y se define el valor picadillo $h_i = H(m_i)$ igual a x_i^e mód N. Para un único valor m_{i_0}, se responde a la petición de picadillo con el valor $h_{i_0} = y$ (recuérdese que y es el entero cuyo inverso bajo el mapa $x \mapsto x^e$ mód N se nos ha pedido encontrar). Se escoge i_0 aleatoriamente y se espera que $m = m_{i_0}$ sea el mensaje cuya firma intentará ser falseada por el falsificador existencial. Siempre que el falsificador solicite la firma de un mensaje m_i, con $i \neq i_0$, se le responde con x_i. Note que este valor es en realidad la verdadera firma de m_i puesto que: $x_i^e \equiv h_i$ mód N. Si el falsificador logra emitir como salida una firma válida s_{i_0} del mensaje m_{i_0}, implica que tendríamos una solución $x = s_{i_0}$ a nuestra ecuación original $y = x^e$ mód N, con x como incógnita. Pero si adivinamos mal y m_{i_0} no es el mensaje cuya firma el falsificador intenta atacar, entonces no seremos capaces de dar una respuesta válida a una petición de firma para m_{i_0}. El falsificador no logrará hallar la firma o dará como salida datos sin sentido, así que tendremos que comenzar otra vez. Suponga que q_h es una cota al número de peticiones de una función picadillo. Si repetimos el procedimiento descrito arriba k veces, la probabilidad que fracasemos todas las k veces en resolver la ecuación $y = x^e$ mód N para x, no es mayor que $(1 - 1/qh)^k$. Para k grande, esta razón tiende a cero; así que con alta probabilidad terminaremos teniendo éxito. Con esto completamos el argumento.

Note que para hallar la raíz e-ésima módulo N solicitada, el programa falsificador tendrá que ser ejecutado aproximadamente $O(q_h)$ veces (donde q_h es el número de peticiones de picadillo). Un resultado de Coron [19] muestra que este desempeño puede ser mejorado a $O(q_s)$, donde q_s denota una cota al número de peticiones de firma.[12] (De esta manera, $q_h = q_s + q'_h$, donde q'_h es una cota al número de peticiones de picadillo que son permitidos a posteriori para una petición de firma de un mismo mensaje.)

Además, en un artículo posterior [20], Coron prueba que, esencialmente, su resultado no puede ser mejorado con un argumento reduccionista apretado: $O(q_s)$ es una cota inferior al número de llamados que un falsificador necesita hacer para resolver el problema de RSA.

Desde el punto de vista práctico (como se enfatiza, por ejemplo, en [5]), esta holgura es importante. Sus implicaciones son las siguientes. Suponga

[12] De acuerdo al argumento descrito arriba, en vez de responder únicamente a la petición de picadillo i_0 con $h_{i_0} = y$, Coron tuvo la idea de responder un cierto número óptimo $i_0, i_1, \ldots$ con $h_{i_j} = yz_j^e$, con z_j aleatorio.

que usted prevé que un oponente de mensaje escogido puede arreglárselas para completar hasta 2^{20} peticiones de firma. Si desea que su sistema tenga 80 bits de seguridad; esto es, si usted desea garantizar que el falsificador que lo ataca se vea forzado a emplear un tiempo proporcional a 2^{80}, entonces los resultados [19, 20] implican que debe elegir un módulo RSA N lo suficientemente grande como para que usted tenga una razonable confianza que las raíces e-ésimas módulo N no podrán ser encontradas en menos de $2^{100} = 2^{20} \cdot 2^{80}$ operaciones. Ello significa que usted debería utilizar un módulo RSA de al menos 1500 bits.

4.3. La Inverosímil Magia de un bit. Examinemos ahora una construcción de Katz y Wang [25], quienes mostraron que agregando un solo bit aleatorio al mensaje, se puede obtener una reducción apretada.[13] Para firmar un mensaje m, Alicia elige aleatoriamente un bit b y evalúa la función picadillo H con m concatenado con el bit b. Después, ella calcula $s = (H(m, b))^d$ mód N; su firma es el par (s, b). Para verificar la firma, Betito calcula $s^e = H(m, b)$ mód N.

Memorablemente, Katz y Wang probaron que el uso de un solo bit aleatorio es suficiente para poder obtener una reducción apretada del problema de RSA al problema de producir una falsificación de una firma Katz-Wang. Para ver esto, suponga que se tiene un falsificador bajo el modelo del oráculo aleatorio que solicita las firmas de algunos mensajes para producir después una firma válida de algún otro mensaje. Dado un entero arbitrario y, el simulador debe utilizar el falsificador para producir x tal que $y = x^e$ mód N. Sin pérdida de generalidad, podemos suponer que cuando el falsificador pregunta por el picadillo de $H(m, b)$, también obtiene $H(m, b')$ (donde b' denota el complemento de b). Ahora cuando el falsificador solicita tal picadillo, el simulador selecciona un bit aleatorio c y dos enteros aleatorios t_1 y t_2. Si $c = b$, entonces el simulador responde con $H(m, b) = t_1^e y$ y $H(m, b') = t_2^e$; si $c = b'$ responde entonces con $H(m, b) = t_2^e$ y $H(m, b') = t_1^e y$. Si el falsificador posteriormente le pide al simulador que firme el mensaje m, el simulador responde con el valor correspondiente de t_2. Al final de este proceso, el falsificador emite como salida una firma que es ya sea la e-ésima raíz de t_2^e o la ésima raíz de $t_1^e y$ para algún t_1 o t_2 que el simulador conoce. En el último caso, el simulador ha tenido éxito en su tarea. Puesto que esto ocurre con probabilidad 1/2, el simulador tiene casi certidumbre -con probabilidad $1 - 2^{-k}$- que encontrará la raíz e-ésima pedida después de ejecutar el programa falsificador k veces. Lo anterior nos da una reducción apretada del problema de RSA al problema de falsificación.

Desde el punto de vista de "Seguridad demostrable orientada a la práctica", la modificación Katz-Wang brinda una mucho mejor garantía que la dada por la firma RSA sin añadir el bit extra. Con esta versión, para obtener una seguridad de 80 bits de seguridad se necesita únicamente escoger N

[13] Discutiremos una versión ligeramente simplificada del esquema Katz-Wang. En particular supondremos que Alicia nunca firma el mismo mensaje dos veces.

suficientemente grande así que encontrar raíces e-ésimas módulo N requiera 2^{80} operaciones (es decir que se necesita aproximadamente un módulo N de 1000 bits), por lo que agregar un solo bit al mensaje nos permite ¡eliminar 500 bits de nuestro módulo RSA!

Lo anterior es un desafío al sentido común. ¿Cómo puede tal "bit mágico" tener *algún* impacto significativo en la verdadera seguridad de un criptosistema, ya no digamos un impacto tan dramático?. Este ejemplo ilustra que ya sea que un protocolo criptográfico se preste o no a un argumento reduccionista apretado de seguridad, ello no necesariamente está relacionado con respecto a la verdadera seguridad que tal protocolo puede ofrecer.

¿Es realmente importante la reducción apretada en el argumento reduccionista de seguridad? Quizás no, si, como en este caso, un protocolo con una reducción holgada puede ser modificado de una manera trivial para obtener una reducción que sí sea apretada. Por otro lado, el ejemplo en §4.1 muestra que en ciertos escenarios una reducción holgada puede ser inútil o de poco valor. Pareciera pues, que la cuestión de cómo interpretar una reducción no apretada no tiene una respuesta fácil.

Una interpretación de la cota inferior de Coron de apretadura es que si el problema de RSA tiene s_1 bits de seguridad y si suponemos que el oponente puede hacer hasta 2^{s_2} solicitudes de firma, entonces las firmas RSA con dominio completo de la función picadillo tienen una seguridad de tan solo $s_1 - s_2$ bits. Sin embargo, dicha conclusión parece no estar muy bien fundamentada a la luz de la construcción Katz-Wang. Más bien, pareciera razonable ver la cota inferior de Coron de apretadura como un resultado que crea dudas no en la seguridad del esquema básico de firma RSA, si no más bien en la utilidad de los argumentos reduccionistas para medir la seguridad de un protocolo. Este punto de vista es similar a la interpretación alternativa del resultado Boneh-Venkatesan que fuera propuesta en §2.

5. Equivalencia pero sin Reducción Apretada

Sea $\mathcal{P}$ un problema presumiblemente difícil en el cual está basado algún protocolo criptográfico. Es decir, el resolver una instancia de $\mathcal{P}$ recuperará una clave privada de algún usuario. Por ejemplo la versión RSA de factorización es el problema $\mathcal{P}$ cuya entrada es un producto N de dos primos desconocidos de k bits y cuya salida es la factorización de N.

Sea $\mathcal{P}_m$ el problema cuya entrada es la m-tupla de distintas entradas válidas para el problema $\mathcal{P}$, las cuales tienen la misma longitud en bits y cuya salida es la salida de $\mathcal{P}$ para cualquiera de sus entradas. En el contexto criptográfico, m podría ser el número de usuarios en un sistema. En ese caso, resolver $\mathcal{P}_m$ implica encontrar la clave privada de cualquiera de los m usuarios, mientras que resolver $\mathcal{P}$ significa encontrar la clave privada de un usuario en específico. El primer caso es conocido como "recuperación existencial de clave", mientras que al segundo se le llama "recuperación universal de clave". Una propiedad deseable de un criptosistema es que esos dos

problemas sean equivalentes, en otras palabras, que el problema de recuperar la clave privada de un usuario designado por el oponente sea equivalente al problema de pedirle al oponente que halle la clave privada de un usuario predeterminado.

Como un ejemplo de cómo esta situación puede darse en la práctica, suponga que en un cierto criptosistema una pequeña proporción —digamos, 10^{-5}— de las claves privadas asignadas aleatoriamente son vulnerables a un cierto ataque. Desde el punto de vista de un usuario individual, el sistema es seguro; él puede tener una confianza de 99.999 % que su clave secreta está a salvo. Sin embargo, desde el punto de vista del administrador, quien debe responder a millones de usuarios, el sistema es inseguro debido a que un oponente puede estar casi cierto (ver discusión más abajo) que eventualmente obtendrá la clave privada de uno o más usuarios, quienes entonces demandarán al administrador. Concluyendo, un administrador de sistema debe preocuparse de la recuperación existencial de clave, mientras que para un usuario individual sólo la recuperación universal de clave es importante.

5.1. El problema de Factorización RSA. Para el caso de RSA: ¿es $\mathcal{P}_m$ equivalente a $\mathcal{P}$? (Por ahora, nuestra pregunta se refiere a algoritmos que resuelven todas las instancias de un problema; pronto, consideraremos algoritmos que resuelven una proporción no despreciable de todas las instancias.) Es improbable que exista una reducción eficiente desde $\mathcal{P}$ a $\mathcal{P}_m$. Tal reducción implicaría que la siguiente afirmación no podría ser cierta: para toda k, hay un número pequeño $r_k < m$ de módulos N que son mucho más difíciles de factorizar que cualquier otro N de longitud $2k$ bits. Por otro lado, todo nuestro conocimiento y experiencia factorizando algoritmos respalda la creencia que, de hecho, esos dos problemas son en la práctica equivalentes, y que RSA sí disfruta la propiedad de que la recuperación existencial y universal de clave son problemas equivalentes.

Cuando se estudia la seguridad de un protocolo, se suele considerar algoritmos que resuelven únicamente una cierta cantidad no despreciable de todas las instancias posibles.[14] En este caso hay una reducción fácil desde $\mathcal{P}$ hasta $\mathcal{P}_m$: dada una entrada a $\mathcal{P}$, escoja aleatoriamente $m-1$ otras entradas para formar una entrada a $\mathcal{P}_m$. Se puede demostrar que esta acción transforma un algoritmo que resuelve una proporción no despreciable de instancias de $\mathcal{P}_m$ a uno que resuelve una proporción no despreciable de instancias de $\mathcal{P}$.

Sin embargo, la proporción de instancias resueltas puede ser dramáticamente diferente. Un algoritmo $\mathcal{A}$ que resuelve ϵ de las instancias de $\mathcal{P}$, donde

[14]En esta sección, los cálculos de probabilidad son siempre tomados sobre el conjunto de las instancias del problema (para un tamaño dado), y no sobre los conjuntos de posibles elecciones (lanzamiento de volados) hechos en la ejecución de un algoritmo. Si para una instancia determinada el algoritmo analizado soluciona una proporción no despreciable de secuencias de volados, entonces suponemos que el algoritmo será iterado el número de veces que sea suficiente para estar casi ciertos de resolver la instancia bajo análisis.

ϵ es pequeño pero no despreciable, crea a un algoritmo $\mathcal{A}_m$ que resuelve $\nu = 1 - (1-\epsilon)^m$ de las instancias de $\mathcal{P}_m$ (esto es, la probabilidad que al menos uno de los m componentes de la entrada pueda ser resuelto por $\mathcal{A}$). Para ϵ pequeño y m grande, $\nu \approx 1 - e^{-\epsilon m}$. Por ejemplo, si $\epsilon = 10^{-5}$ y $m = 10^6$, entonces ν es mayor que 99,99 %. De esta manera, desde un punto de vista teórico pareciera que para muchos sistemas tales como RSA, hay una distancia significativa entre recuperación universal de clave privada $\mathcal{P}$ y recuperación existencial de clave privada $\mathcal{P}_m$. En otras palabras, no conocemos ningún argumento no reduccionista que muestre que si RSA es seguro desde el punto de vista de un usuario individual, entonces deba también ser seguro desde el punto de vista del administrador del sistema.

Pero una vez más, toda nuestra experiencia e intuición sugiere que no hay una distancia real entre las dos versiones del problema de factorización RSA. Esto se debe a que para todos los algoritmos de factorización de tiempo subexponencial que se conocen, lo cual incluye al cribado de cuerpo numérico (*number field sieve*), se cree que el tiempo de ejecución no es substancialmente diferente para (a) una instancia escogida aleatoriamente, (b) una instancia de dificultad promedio, y (c) la instancia más difícil posible. Nadie sabe cómo probar esa afirmación; ciertamente, nadie ha podido siquiera dar una prueba rigurosa del tiempo de ejecución $L_{1/3}$ para el cribado de cuerpo numérico. Y aún si dicha afirmación pudiese ser probada para el más rápido de los algoritmos de factorización, estaríamos todavía muy lejos de poder probar que nunca podrá existir un algoritmo más veloz, para el cual haya una vasta diferencia entre los tiempos de ejecución del caso promedio y del más difícil. Debido a ello, no existen esperanzas de probar una equivalencia estrecha entre recuperación de clave privada existencial y universal.

5.2. Un Ejemplo no Criptográfico. Considere el problema $\mathcal{P}$ de encontrar los factores primos de un entero arbitrario N. Diremos que N es "k-fácil" si tiene a lo más un divisor primo mayor que 2^k. En el caso que k es un entero pequeño, entonces $\mathcal{P}$ puede ser resuelto eficientemente usando primero prueba por división, quizás combinada con el algoritmo de factorización por curvas elípticas de Lenstra para extraer los factores $< 2^k$, seguido por una prueba de primalidad para averiguar si la porción remanente es mayor que 1.

No es difícil mostrar que la proporción ϵ de enteros de longitud de n bits que son k-fáciles es de al menos k/n. Para $1 \le j < 2^k$ considere módulos N que sean de la forma pj para primos p. El número de tales enteros de n bits es asintótico a

$$\frac{2^{n-1}}{j \ln(2^n/j)} > \frac{2^{n-1}}{\ln 2^n} \frac{1}{j}.$$

De esta manera, la proporción de enteros de n bits que son k-fáciles es mayor que,

$$\frac{1}{\ln 2^n} \sum_{1 \le j < 2^k} \frac{1}{j} \approx \frac{\ln 2^k}{\ln 2^n} = \frac{k}{n}.$$

Como un ejemplo, tómese $n = 2000$, $k = 20$. Entonces, $\epsilon \geq 0{,}01$. Vemos que para $m = 1000$ más del 99.99 % de todas las instancias $\mathcal{P}_m$ pueden ser fácilmente resueltas. En contraste, una proporción significativa de las instancias de $\mathcal{P}$ están fuera de nuestro alcance. Claramente, no es computacionalmente factible factorizar módulos RSA de 2000 bits. Pero hay un conjunto mucho mayor de enteros de 2000 bits que no pueden ser completamente factorizados con la tecnología actual. Así, sea $S_{\geq 1}$ el conjunto de todos los enteros que tienen al menos un factor primo en el intervalo $[2^{300}, 2^{500}]$ y al menos un factor primo mayor que 2^{500}. Al día de hoy, no es factible factorizar un número en $S_{\geq 1}$, incluso si se usa una combinación del método de factorización de curvas elípticas y del cribado de cuerpo numérico; además, dando un argumento heurístico mostramos a continuación que al menos 25 % de todos los enteros con longitud de 2000 bits están en $S_{\geq 1}$.

Para demostrar lo anterior, sea S_k el conjunto de enteros que tienen exactamente k factores primos en $[2^{300}, 2^{500}]$ y al menos un factor primo mayor que 2^{500}. Escribiendo un número $N \in S_1$ de 200 bits en la forma $N = lm$ con l un primo en $[2^{300}, 2^{500}]$ y $m \in S_0$, vemos que el número de tales N es igual a

$$\sum_{l \text{ primo en } [2^{300},2^{500}]} \#\left(S_0 \bigcap \left[\frac{1}{l}2^{1999}, \frac{1}{l}2^{2000}\right]\right).$$

La probabilidad que un entero que esté en el intervalo de arriba satisfaga las dos condiciones que definen S_0 es al menos igual a

$$\text{Prob(no es divisible por un primo } p \in [2^{300}, 2^{500}]) - \text{Prob}(2^{500} - \text{suave})$$

$$\approx \prod_{p \in [2^{300},2^{500}]} (1 - \frac{1}{p}) - u^{-u},$$

donde $u = (2000 - \log_2 l)/500 \geq 3$. El producto es igual a $\exp \sum \ln(1 - \frac{1}{p}) \approx \exp \sum(-1/p) \approx \exp(-\ln\ln 2^{500} + \ln\ln 2^{300}) = 0{,}6$, y así la probabilidad que un entero en $[\frac{1}{l}2^{1999}, \frac{1}{l}2^{2000}]$ esté en S_0 es mayor que 50 %. Por lo que la proporción de enteros N de 200 bits que están en $S_{\geq 1} \supset S_1$ es de al menos

$$\frac{1}{2} \sum_{l \text{ primo en } [2^{300},2^{500}]} \frac{1}{l} \approx \frac{1}{2}(\ln\ln 2^{500} - \ln\ln 2^{300}) = \frac{1}{2}\ln(5/3) \approx 0{,}25,$$

como fue afirmado.

Al parecer, el problema $\mathcal{P}$ analizado aquí no tiene ninguna relevancia en criptografía: es difícil imaginar un protocolo cuya seguridad esté fundamentada en la dificultad de factorizar un entero escogido aleatoriamente. Más bien su interés descansa en el hecho que, a pesar de su aparente semejanza con el problema de factorización RSA, falle tan espectacularmente en exhibir una cierta propiedad — equivalencia estrecha en la solución del problema existencial y universal — propiedad que, intuitivamente, parece ser una característica de la factorización RSA. Este ejemplo también sugiere que probablemente no existe ninguna esperanza de *probar* la equivalencia

estrecha de la recuperación existencial y universal de clave privada en el problema de RSA.

5.3. ¿Se Deben usar Diferentes Curvas Elípticas o Una Sola? Analicemos ahora los problemas de recuperación universal de clave secreta versus la recuperación existencial en el caso de criptografía de curvas elípticas (CCE). Suponga que cada usuario escoge una curva elíptica E sobre un cuerpo finito $\mathbb{F}_q$, subgrupo de $E(\mathbb{F}_q)$ cuyo orden es un primo de k bits, un punto base P dentro del subgrupo y una clave secreta x mód p; siendo la clave pública $Q = xP$. Sea $\mathcal{P}$ el problema del logaritmo discreto en curvas elípticas (ECDLP por sus siglas en inglés), esto es, el problema de recuperar la clave secreta x a partir de la información pública. Sea $\mathcal{P}_m$ el problema cuya entrada es una m-tupla de entradas de ECDLP con distintos órdenes p de los subgrupos y cuya salida es cualquiera de los m logaritmos discretos. Una vez más, parece intuitivamente claro que $\mathcal{P}_m$ es tan difícil como $\mathcal{P}$, aunque es muy improbable que una reducción estrecha desde $\mathcal{P}$ hasta $\mathcal{P}_m$ pueda ser encontrada.

En contraste, suponga que todos los usuarios usan el mismo grupo de curvas elípticas de tal manera que sólo los pares de clave público/privado difieren. En ese caso, CCE *demostrablemente* disfruta la propiedad de equivalencia estrecha de recuperación existencial y universal de clave privada. La razón es que el ECDLP en un grupo fijo es "auto-reducible." Ello significa que dada una instancia que se desea resolver, es fácil construir una m-tupla compuesta por diferentes instancias aleatorias tal que la solución de cada una de ellas dé la solución al problema que queremos resolver. Así, dada una entrada Q, se escogen aleatoriamente m enteros distintos y_i módulo p y se hace $Q_i = y_i Q$. Un oráculo $\mathcal{P}_m$ resuelve una de las instancias ECDLP con entrada Q_i. Una vez que conocemos el logaritmo discreto x_i, inmediatamente se encuentra $x = y_i^{-1} x_i$ mód p. Esto muestra que para el ECDLP en una curva fija, el problema de recuperación universal de clave privada $\mathcal{P}$ se reduce (estrechamente) al problema de recuperación existencial de clave privada $\mathcal{P}_m$.

De esta manera, si se quiere un criptosistema con la propiedad de seguridad *demostrable* de equivalencia estrecha en la recuperación existencial y universal de clave privada, entonces no sólo deberíamos preferir CCE por encima de RSA, sino también insistir en que todos los usuarios trabajen con el mismo grupo de curva elíptica.

Demás está decir que no estamos sugiriendo que este argumento sea una buena razón para escoger un tipo de criptografía en vez de otro. Por el contrario, lo que se muestra con este ejemplo es que algunas veces resulta ingenuo usar la existencia o ausencia de un argumento reduccionista estrecho de seguridad como una guía para determina cuál versión de un criptosistema es preferible.

Comentario 3. Cabe señalar la historia problemática, llena de infructuosos intentos por construir criptosistemas cuya seguridad esté fundamentada en

un problema para el cual los casos con dificultad promedio y con la más alta dificultad sean *demostrablemente* equivalentes.[15] Esto fue finalmente logrado por Ajtai y Dwork [2] en 1997. Sin embargo, el siguiente año, Nguyen y Stern [30] encontraron un ataque que recupera la clave secreta en el sistema Ajtai-Dwork a menos que sus parámetros sean demasiado grandes como para ser prácticos. (véase también [31]).

6. Generadores de bits Pseudo-aleatorios

Un generador de bits pseudo-aleatorios G es una función — en realidad una familia de funciones parametrizadas por n y $M \gg n$ — que toma como entrada una secuencia aleatoria de n bits (conocida como "semilla") y da como salida una secuencia de M bits que tienen apariencia aleatoria. Más precisamente, se dice que G es *asintóticamente seguro en el sentido de indistinguibilidad* si no existe una prueba estadística de tiempo polinomial que pueda distinguir (por un margen no despreciable) la diferencia entre su salida y una salida aleatoria. Una noción alternativa y a primera vista más débil de seguridad es la prueba del "siguiente bit", la cual nos dice que para ningún valor de j existe un algoritmo de tiempo polinomial que dados los primeros $j-1$ bits pueda predecir el j-ésimo bit con una posibilidad de éxito mayor a $\frac{1}{2}+\epsilon$ (donde ϵ es no despreciable como función de n). Un teorema de Yao (véase [26], pp. 170-171) muestra que estas dos nociones de seguridad son en realidad equivalentes. Sin embargo, ese teorema es no-estrecho en el sentido que la tolerancia ϵ para la prueba del siguiente bit corresponde únicamente a una tolerancia de indistinguibilidad ($M\epsilon$).

Para poder analizar de manera más concreta la seguridad de un generador de bits pseudo-aleatorios, es necesario utilizar una definición más precisa que la definición asintótica. De esta manera, dados los valores n y M, se dice que G es (T, ϵ)-*seguro en el sentido de indistiguibilidad* si no hay un algoritmo (prueba estadística) con un tiempo de ejecución no mayor que T tal que el valor absoluto de la diferencia entre la probabilidad de una decisión 'sí" en respuesta a la salida de G y la probabilidad de un 'sí" en respuesta a una secuencia genuinamente aleatoria de M bits sea mayor que ϵ. La relación entre indistinguibilidad y la prueba del siguiente bit es que si se sabe que nuestro generador es $(T, \epsilon/M)$-seguro en el sentido del siguiente bit se puede concluir que éste es (T, ϵ)-seguro en el sentido de indistiguibilidad.

6.1. El Generador Blum–Blum–Shub. Sea N un número de n bits producto de dos primos grandes, donde ambos primos son $\equiv 3 \pmod 4$ (a tales enteros N se les conoce como "enteros Blum"), y sea j un entero pequeño. El generador de bits pseudo-aleatorios Blum–Blum–Shub (BBS)

[15] Los sistemas basados en el logaritmo discreto no tienen esta propiedad porque el problema matemático en que se sustentan es auto-reducible sólo después que el grupo ha sido fijado; claramente no existe forma de reducir una instancia a la otra cuando los grupos tienen diferentes órdenes.

G, toma un entero aleatorio x mod N y produce $M = jk$ bits de acuerdo al siguiente procedimiento. Sea $x_0 = x$, para $i = 1, \ldots, k$ y sea[16]

$$x_i = \text{mín}\{x_{i-1}^2 \text{ mód } N,\ N - (x_{i-1}^2 \text{ mód } N)\}.$$

Entonces la salida de G está formada por los j bits menos significativos de x_i, $i = 1, \ldots, k$.

Obviamente, entre más grande sea j, más rápido generará G, M bits de salida. Sin embargo, la posibilidad de distinguir entre la secuencia generada y una secuencia genuinamente aleatoria se vuelve más grande conforme j se incrementa. En [41] y [3] fue demostrado que $j = O(\log \log N)$ bits pueden ser extraidos por iteración de forma segura bajo la suposición que la factorización es un problema intratable.

Este valor asintótico fue usado como argumento para recomendar valores de j. Por ejemplo, en 1994 la *Internet Engineering Task Force* [21] hizo la siguiente recomendación (en esta y en la siguiente cita el módulo es denotado como n en vez de N):

> En la actualidad el generador que tiene la más sólida prueba de fortaleza es el generador Blum Blum Shub ... Si se usan los $\log_2(\log_2(s_i))$ bits menos significativos, entonces predecir bits adicionales a partir de una secuencia generada de esta manera es demostrable [sic][17] tan difícil como factorizar n.

Esta recomendación ha sido reiterada más recientemente, por ejemplo, en el libro de Young and Yung ([43], p. 68):

> También se piensa que el generador Blum–Blum–Shub PRBG es seguro cuando los $\log_2(\log_2(n))$ bits menos significativos...son usados (en vez de utilizar únicamente el bit menos significativo). Así, cuando n es un entero compuesto de 768-bit, los 9 bits menos significativos pueden ser usados en la secuencia pseudo-aleatoria.

Comparemos esta recomendación con las mejores cotas de seguridad conocidas. En lo que sigue definimos,

$$L(n) \approx 2{,}8 \cdot 10^{-3} \exp\left(1{,}9229 (n \ln 2)^{1/3} (\ln(n \ln 2))^{2/3}\right),$$

que es, heurísticamente, el tiempo de ejecución esperado para que el método de cribado de cuerpo numérico factorice un entero Blum aleatorio de n bits (aquí la constante $2{,}8 \cdot 10^{-3}$, tomada de [40], fue obtenida del tiempo de ejecución reportado para factorizar un entero de 512 bits), y donde se presupone que no existe otro algoritmo que pueda factorizar tal entero en un tiempo esperado menor que $L(n)$.

Para la versión $j = 1$ del Blum–Blum–Shub el mejor resultado concreto de seguridad (para n grande) fue hallado por Fischlin and Schnorr [22],

[16] El generador original descrito en [9] usa $j = 1$ y $x_i = x_{i-1}^2$ mód N.

[17] *provable* en el original [N. del T.]

quienes mostraron que el generador BBS es (T, ϵ)-seguro en el sentido de indistinguibilidad si

$$T \leq \frac{L(n)(\epsilon/M)^2}{6n \log n} - \frac{2^7 n(\epsilon/M)^{-2} \log(8n(\epsilon/M)^{-1})}{\log n}, \tag{1}$$

donde log denota $\log_2$ aquí y en lo que sigue.

Para $j > 1$ la desigualdad Fischlin–Schnorr (1) fue generalizada por Sidorenko and Schoenmakers [40], quienes mostraron que el generador BBS es (T, ϵ)-seguro si

$$T \leq \frac{L(n)}{36n(\log n)\delta^{-2}} - 2^{2j+9} n\delta^{-4}, \tag{2}$$

donde $\delta = (2^j - 1)^{-1}(\epsilon/M)$. Para n grande esta es una mejora sobre la desigualdad

$$T \leq \frac{L(n)(\epsilon/M)^8}{2^{4j+27} n^3}, \tag{3}$$

que fue obtenida de la prueba de seguridad en [3].

Regresando a los parámetros recomendados en [21] y [43], tomemos $n = 768$ y $j = 9$. Suponga que usamos además $M = 10^7$ y $\epsilon = 0{,}01$. De acuerdo a la desigualdad (2), el generador BBS es seguro contra un adversario cuyo tiempo está acotado por -2^{192}. (Sí, ¡ese es un signo negativo!) En este caso, obtenemos un "mejor" resultado de la desigualdad (3), la cual acota el tiempo del adversario por 2^{-264}. (Sí, ¡ese es un exponente negativo!) Estas garantías menos-que-reconfortantes no mejoran demasiado si se cambian los valores de M y ϵ. Por ejemplo, si $M = 2^{15}$ y $\epsilon = 0{,}5$, se obtiene $T \leq -2^{136}$ y $T \leq 2^{-134}$ de (2) y (3), respectivamente. De esta forma, dependiendo de si utilizamos (2) o (3), el tiempo d ejecución del adversario está acotado por ya sea un número negativo o por ¡10^{-40} ciclos de reloj!

Tampoco la recomendación en [21] y [43] funciona bien para valores grandes de n. En la Tabla 1, la primera columna lista algunos valores seleccionados de n; la segunda columna presenta los valores de $L(n)$ a la potencia de dos más cercana (esta es la cota en tiempo de ejecución del adversario que resultaría de una reducción estrecha); la tercera columna lista los valores correspondientes del lado derecho de la desigualdad (2); y la cuarta columna presenta los valores del lado derecho de la ecuación (3), donde se está usando, $j = \lfloor \log n \rfloor$, $M = 10^7$, y $\epsilon = 0{,}01$.

Así, el resultado asintótico en [3, 41], el cual parece garantizarnos que se podrían extraer $j = \lfloor \log n \rfloor$ bits en cada iteración, no ofrece en la práctica lo que promete en la teoría.

Suponga que retiramos la idea de obtener $j = \lfloor \log n \rfloor$ bits por cada iteración, y en vez de esto, usamos el generador BBS para darnos únicamente $j = 1$ bit por iteración. Entonces, las garantías de seguridad dadas por las desigualdades (1) y (3) son mejores, pero no mucho mejores de lo que uno podría esperar. La tabla 2 muestra los valores correspondientes de los lados

n	$L(n)$	Cota tomada de (2)	Cota tomada de (3)
1024	2^{78}	-2^{199}	2^{-258}
2048	2^{108}	-2^{206}	2^{-235}
3072	2^{130}	-2^{206}	2^{-215}
7680	2^{195}	-2^{213}	2^{-158}
15360	2^{261}	-2^{220}	2^{-99}

CUADRO 1. El Generador BBS: cotas en el tiempo de ejecución del adversario con $j = \lfloor \log n \rfloor$.

derechos de (1) (en la tercera columna) y (3) (en la cuarta columna) para $j = 1$, $M = 10^7$, y $\epsilon = 0{,}01$.

n	$L(n)$	Cota tomada de (1)	Cotada tomada de (3)
1024	2^{78}	-2^{79}	2^{-222}
2048	2^{108}	-2^{80}	2^{-194}
3072	2^{130}	-2^{80}	2^{-175}
7680	2^{195}	2^{115}	2^{-114}
15360	2^{261}	2^{181}	2^{-51}

CUADRO 2. El Generador BBS: cotas en el tiempo de ejecución del adversario con $j = 1$.

El punto de cruce para el cual la desigualdad Fischlin–Schorr empieza a dar garantías de seguridad sensatas es alrededor de $n = 5000$ (para el cual el lado derecho de la ecuación (1) es aproximadamente 2^{84}). Desafortunadamente, no es muy eficiente tener que calcular operaciones de cuadrados modulares de 5000 bits para cada bit de la secuencia pseudo aleatoria.

Comentario 4. El valor recomendado $j = \log(\log N)$ en [21] y [43] fue obtenido a partir del resultado asintótico de $j = O(\log(\log N))$, donde se le asignó a la constante implícita C de la notación *big-O* un valor de 1. La elección de $C = 1$ es arbitraria. En muchos resultados asintóticos de la teoría de números la constante implícita es mucho mayor, así que con esta misma justificación arbitraria podríamos decidir tomar un valor de $C = 100$. Resulta asombroso observar que si se hace eso con una N de 1000 bits, se obtendría un generador BBS completamente inseguro. Puesto que $j = 100 \log(\log N) = 1000$, se estarían utilizando todos los bits de x_i. Para esa salida, un oponente podría fácilmente determinar N (haciendo $N_1 = x_2 \pm x_1^2$, $N_i = \gcd(N_{i-1}, x_{i+1}\ pm x_i^2)$, así que $N_i = N$ para $i \geq i_0$ para valores muy pequeños de i_0), después de lo cual la secuencia pasaría a ser determinística a los ojos del oponente.

6.2. El generador Gennaro. Sea p un primo de n bits de la forma $2q+1$ con q primo, y sea c un entero tal que $c \gg \log n$. Sea g un elemento generador de $\mathbb{F}_p^*$. El generador de bits pseudo-aleatorios G de Gennaro toma como entrada un aleatorio x mod $p-1$ y produce $M = (n-c-1)k$ bits de salida como sigue (véase [23]). Sea $x \mapsto \tilde{x}$ la función de enteros de n bits $x = \sum_{l=0}^{n-1} s_l 2^l$ dada por $\tilde{x} = s_0 + \sum_{l=n-c}^{n-1} s_l 2^l$. Sea $x_0 = x$, y para $i = 1, \ldots, k$ sea $x_i = g^{\tilde{x}_{i-1}}$ mód p. Entonces la salida de G consiste desde el segundo hasta $(n-c)$-avo bit of x_i, $i = 1, \ldots, k$ (estos son los bits que son ignorados en $\tilde{x}_i$).

En comparación con el generador BBS, cada iteración de la exponenciación $x_i = g^{\tilde{x}_{i-1}}$ mód p toma más tiempo que la operación de cuadrado modular. Sin embargo, se obtienen muchos más bits cada vez. Por ejemplo, con parámetros $n = 1024$ y $c = 160$, recomendados en [24], cada iteración brinda 863 bits.

En [24], Howgrave-Graham, Dyer, y Gennaro comparan el generador Gennaro (con $n = 1024$ y $c = 160$) con un generador de bits pseudo-aleatorios basado en SHA-1 (es decir, el generador ANSI X9.17) el cual no cuenta con una prueba de seguridad:

> ...La generación de números pseudo-aleatorios basada en SHA-1 es todavía considerablemente más rápida que la generación basada en logaritmos discretos. Sin embargo, la diferencia, un factor de menos de 4 en este hardware, puede ser considerado un precio no muy alto a pagar para aquellos que deseen construir un sistema de generación de números pseudo-aleatorios "demostrablemente seguro," en vez de uno "aparentemente seguro" (es decir, un sistema que, hasta el momento, ha resistido ataques).

La prueba de seguridad para el generador Gennaro puede hallarse en §4 de [23]. Curiosamente, Gennaro usa para derivar sus resultados la prueba del siguiente bit en vez del criterio de indistiguibilidad. Sin embargo, más que la prueba del siguiente bit, es este último criterio el que ha sido ampliamente aceptado como noción de seguridad para un generador de bits pseudo-aleatorios. Como se comentó arriba, para pasar de la prueba del siguiente bit a la indistiguibilidad, se debe reemplazar en las desigualdades ϵ por ϵ/M. En [39] se halla que la prueba de Gennaro brinda la siguiente desigualdad para el tiempo de ejecución del adversario:

$$T \leq \frac{L(n)(n-c)^3}{16c(\ln c)(M/\epsilon)^3}. \tag{4}$$

Para $n = 1024$, $c = 160$, $M = 10^7$, y $\epsilon = 0{,}01$, el lado derecho de (4) es 18. De esta manera, las garantías de seguridad del generador Gennaro no son mucho mejores que las que se dan en §6.1.

Concluimos esta sección repitiendo el comentario hecho en §5.5 de [27]:

> Desafortunadamente, este tipo de análisis (incorporar la medida de no-apretadura en las recomendaciones para los tamaños de los parámetros) es generalmente omitido en los artículos que abogan por nuevos protocolos en base a una "prueba" de su seguridad. Típicamente, los autores de esos artículos proclaman a los cuatro vientos la ventaja de su protocolo por encima de otros protocolos similares que no poseen una prueba de seguridad (o que únicamente tienen una prueba de seguridad basada en el modelo del oráculo aleatorio), para después proceder a dar un argumento reduccionista no-apretado, finalizando con recomendaciones sobre la longitud de la clave que se debe emplear, lo cual tendría sentido si se hubiese dado una prueba apretada. Tales autores no informan a los usuarios potenciales de su protocolo del verdadero nivel de seguridad que queda garantizado por la "demostración" si, digamos, un primo de 1024 bits es utilizado. Creemos que los criptográfos deberían ser consistentes. Si creen con firmeza que los argumentos reduccionistas de seguridad son muy importantes, entonces se deberían dar recomendaciones para los tamaños de los parámetros basadas en un análisis honesto del argumento de seguridad, aun si ello significa admitir que la eficiencia debe ser sacrificada.

7. Firmas Cortas

En los primeros días de la seguridad demostrable, los investigadores se contentaban con dar resultados asintóticos hallados con reducciones en tiempo polinomial. En años recientes, se ha reconocido más y más la importancia de hacer análisis detallados de esas reducciones lo cual permite establecer resultados cuantitativos en términos de cotas, probabilidades y tiempos de ejecución.

Sin embargo, y de manera desafortunada, dichos autores frecuentemente no hacen interpretaciones de las implicaciones prácticas de las fórmulas y cotas de sus lemas y teoremas. Como resultado, aun los mejores investigadores publican resultados que, cuando se analizan de una manera concreta, terminan careciendo de significado en la práctica. En esta sección damos un ejemplo de esta situación.

Primero, recordamos que cuando se analiza la seguridad de un esquema de firma ante un ataque de mensaje escogido en el modelo del oráculo aleatorio, siempre se tienen dos tipos diferentes de peticiones al oráculo — peticiones de firma y peticiones de picadillos — cada una de ellas con una correspondiente cota respecto al número de peticiones que un oponente puede hacer.[18] En

[18] Continuaremos utilizando la notación q_s y q_h para estas cotas, aun cuando también estamos utilizando q para denotar el orden primo del grupo. Nos disculpamos con el lector por el sobre uso de la letra q en nuestra notación.

la práctica, puesto que las peticiones de firma requieren una respuesta de la entidad bajo ataque, dichas peticiones debieran hasta cierto punto estar limitadas en número. Así que es razonable suponer que la cota q_s es del orden de un millón a mil millones. En contraste, una petición al oráculo de picadillo corresponde en la práctica a evaluar una función disponible públicamente. De modo que no hay justificación para suponer que el número de solicitudes de picadillo que haga el oponente estarán limitadas por nada más que no sea el tiempo de ejecución. De esa manera, es razonable suponer que q_h puede llegar a valer 2^{80}, o al menos 2^{50}.

Damos ahora una descripción de tres esquemas de firma propuestos por Boneh-Lynn-Shacham [12] y Boneh-Boyen [11]. Los tres esquemas utilizan pareos bilineales para obtener firmas cortas cuya seguridad contra el ataque del mensaje escogido esté respaldada por argumentos reduccionistas. Sea k el parámetro de seguridad; en la práctica, normalmente se hace $k \approx 80$. En el diseño de implementaciones eficientes, el orden del grupo q generalmente utilizado es de aproximadamente 2^{2k}, lo cual es lo suficientemente grande como para prevenir ataques de la raíz cuadrada en el problema del logaritmo discreto.

En el esquema de firma Boneh-Lynn-Shacham (BLS), las firmas tienen una longitud de sólo $2k$. En [12] se muestra que este esquema es seguro contra el ataque del mensaje escogido en el modelo del oráculo aleatorio si el problema computacional Diffie-Hellman es difícil.

En [11] Boneh y Boyen proponen dos alternativas al esquema BLS. La primera (referido abajo como "el esquema de firma BB ") tiene aproximadamente el doble de la longitud de la firma de BLS, es decir, $4k$, pero puede probarse como segura ante el ataque del mensaje escogido sin usar el modelo del oráculo aleatorio, presuponiendo que el así llamado problema fuerte de Diffie-Hellman (SDH por sus siglas en inglés) es intratable. El segundo esquema de firma propuesto en el artículo (el "esquema picadillo-firma BB") es una variante del primero en el cual el mensaje debe ser sometido a picadillo. Su prueba de seguridad utiliza el modelo del oráculo aleatorio. Como en el esquema BLS, el esquema de firma-picadillo BB tiene una longitud de firma de aproximadamente $2k$ en vez de $4k$; además, tiene la ventaja sobre BLS que la verificación es aproximadamente dos veces más rápida.

Las pruebas en [11] son claras y leíbles, en parte porque los autores introducen una versión simplificada del esquema BB (el esquema BB"básico") para poder así formular un lema auxiliar (Lema 1) que se usa para probar la seguridad tanto del esquema BB completo (sin oráculos aleatorios) y el esquema picadillo-firma BB (con oráculos aleatorios). Nos interesa aquí analizar el segundo de esos resultados (Teorema 2).

Describimos ahora nuestra razón para dudar del verdadero valor de ese resultado. Escribiremos el Lema 1 y el Teorema 2 de [11] en una forma ligeramente simplificada donde se omite mencionar las probabilidades ϵ y ϵ', las cuales no son relevantes para nuestra discusión. Para ambos esquemas

BB, el problema difícil subyacente es SDH, parametrizado por un entero que denotaremos q'_s.

Lema 1. Suponga que q'_s-SDH no puede ser resuelto en un tiempo menor a t'. Entonces el esquema básico de firma es seguro contra un ataque débil de mensaje escogido efectuado por un falsificador existencial cuyas peticiones de firma están acotadas por q''_s y cuyo tiempo de ejecución está acotado por t'', siempre que

$$q''_s < q'_s \qquad \text{y} \qquad t'' \leq t' - \Theta({q'_s}^2 T),$$

donde T es el tiempo máximo para una exponenciación de grupo.

Teorema 2. Suponga que el esquema básico de firma mencionado en el Lema 1 es existencialmente infalsificable ante un ataque de mensaje escogido débil, con cotas q''_s y t''. Entonces el esquema picadillo-firma correspondiente es seguro en el modelo del oráculo aleatorio en contra de un ataque adaptivo de mensaje escogido efectuado por un falsificador existencial cuyas solicitudes de firma están acotadas por q_s, cuyas peticiones de picadillo están acotadas por q_h, y cuyo tiempo de ejecución está acotado por t, siempre que

$$q_s + q_h < q''_s \qquad \text{y} \qquad t \leq t'' - o(t'').$$

Muchos lectores ocasionales probablemente percibirán que este teorema brinda una garantía de seguridad precisa y definitiva, especialmente puesto que los autores comentan: " Nótese que la reducción en el teorema 2 es apretada... Las demostraciones de seguridad para esquemas de firma en el modelo del oráculo aleatorio frecuentemente son mucho menos apretadas." Probablemente, los lectores no padecerán el problema de contrastar las afirmaciones del teorema con las del Lema 1, particularmente puesto que en [11] varias páginas de texto separan al lema del teorema. Pero tal comparación debe hacerse si se desea evitar caer en la penosa situación de la sección previa (ver Tablas 1 y 2), donde el tiempo de ejecución del adversario está acotado por un número negativo.

Si se ponen ambas afirmaciones una junto a la otra y se comparan, veremos que para que lograr que la cota en el tiempo de ejecución del adversario sea un número positivo es necesario que

$$q_h^2 < t' \approx 2^k,$$

donde k es el parámetro de seguridad. En la práctica, ello significa que se necesita que $q_h \ll 2^{40}$.[19] De esta manera, no existe ninguna garantía de seguridad para el esquema de firma-picadillo del teorema 2 a menos que se suponga que el adversario está severamente limitado en el número de picadillos que puede obtener.

La conclusión de este análisis no es, por supuesto, que el esquema de firma del teorema 2 de [11] es necesariamente inseguro, sino más bien que el

[19]Si se tiene que el orden de un grupo es de 160 y se tiene $q_h = 2^{50}$, entonces el Teorema 2 y el Lema 1 darían una cota de $t \leq -2^{100}$ para el tiempo de ejecución del adversario.

resultado de seguridad demostrable hallado para éste no tiene sentido si los parámetros son escogidos para una implementación eficiente.

8. Los Resultados Paillier–Vergnaud para Firmas Schnorr

En [32] Paillier y Vergnaud prueban que es improbable que una reducción — más precisamente, una reducción "algebraica" — pueda ser encontrada desde el problema del logaritmo discreto (DLP por sus siglas en inglés) al falsificado de firmas Schnorr. Tras describir este resultado y correspondiente demostración, lo compararemos con varios resultados positivos que sugieren que existe equivalencia entre la falsificación de firmas tipo Schnorr y el DLP.

8.1. Firmas Schnorr. Describimos primero el esquema de firma Schnorr [35].

Generación de Claves Schnorr. Sea q un primo grande, y sea p un primo tal que $p \equiv 1 \pmod{q}$. Sea g un generador del subgrupo cíclico G de orden q en $\mathbb{F}_p^*$. Sea H una función picadillo que toma valores en el intervalo $[1, q-1]$. Cada usuario Alicia construye sus claves seleccionando un entero aleatorio x en el intervalo $[1, q-1]$ para después calcular $y = g^x \bmod p$. La clave pública de Alicia es y, y su clave privada es x.

Generación de Firma Schnorr. Para firmar un mensaje m, Alicia debe hacer lo siguiente:

1. Seleccionar un entero aleatorio k en el intervalo $[1, q-1]$.
2. Calcular $r = g^k \bmod p$, y hacer $h = H(m, r)$.
3. Calcular $s = k + hx \bmod q$.

la firma correspondiente al mensaje es el par de enteros (h, s).

Verificación de firma Schnorr. Para verificar la firma de Alicia (h, s) de un mensaje m, Betito debe hacer lo siguiente:

1. Obtener una copia autenticada de la clave pública de Alicia y.
2. Verificar que h y s son enteros en el intervalo $[0, q-1]$.
3. Calcular $u = g^s y^{-h} \bmod p$ y $v = H(m, u)$.
4. Aceptar la firma si y sólo si $v = h$.

8.2. Paillier–Vergnaud. Antes de describir el resultado de Paillier–Vergnaud, necesitamos discutir algunos conceptos preliminares. Primero supóngase que se tiene un grupo G generado por g. Por el "logaritmo discreto" de $y \in G$ entenderemos una solución x a la ecuación $g^x = y$. En [32] el problema "DLP uno-muchos", denotado n-DLP, se define como sigue.

n-DLP. Dado $r_0, r_1, \ldots, r_n \in G$ y un oráculo del logaritmo discreto $DL(\cdot)$ que puede ser invocado n veces, encuentre los logaritmos discretos de todos los $n+1$ elementos r_i.

Segundo, por una reducción "algebraica" $\mathcal{R}$ desde el DLP a falsificación, Paillier and Vergnaud entienden una reducción que es capaz de realizar

operaciones de grupo pero que no es capaz de explotar características especiales de la forma en que los elementos del grupo son representados. Además, los autores suponen que las elecciones hechas mientras se computa $\mathcal{R}$ están disponibles para quienquiera que esté ejecutando el algoritmo (en la demostración de abajo se trata de un solucionador n-DLP). Con estas definiciones ellos prueban el siguiente resultado.

Teorema. Suponga que G es un grupo de orden q generado por g. Suponga que $\mathcal{R}$ es una reducción algebraica desde el DLP a la falsificación universal con un ataque sólo por clave que hace n llamados al falsificador. Entonces n-DLP es fácil.

Demostración. Sea $r_0, r_1, \ldots, r_n \in G$ una instancia de n-DLP. Se nos pide hallar todos los $n+1$ logaritmos discretos, y podemos invocar al oráculo $DL(\cdot)$ un total de n veces. La reducción $\mathcal{R}$ encontrará el logaritmo discreto de cualquier elemento si cuenta con un falsificador capaz de romper n instancias diferentes (escogidas por $\mathcal{R}$) del esquema de firma Schnorr. Le pedimos a $\mathcal{R}$ que encuentre el logaritmo discreto de r_0. Entonces en un total de n ocasiones el algoritmo de reducción produce una clave pública Schnorr y_i y un mensaje m_i. En cada instancia, simulamos al falsificador escogiendo $r = r_i$, calculando el valor del picadillo $h_i = H(m_i, r_i)$, para después hacer s_i igual al logaritmo discreto de $r_i y_i^{h_i}$, el cual determinamos a partir del oráculo:

$$s_i = DL(r_i y_i^{h_i}).$$

Enviamos (h_i, s_i), el cual es una firma legítima de m_i con la clave pública y_i, a $\mathcal{R}$. Finalmente, $\mathcal{R}$ da como salida el logaritmo discreto x_0 de r_0.

Con el objeto de calcular la clave pública y_i, $\mathcal{R}$ debe haber realizado operaciones de grupo comenzando por los únicos dos elementos que le fueron dados, es decir, g y r_0. De esta manera, para algunos valores enteros α_i y β_i que nos son dados, tenemos $y_i = g^{\alpha_i} r_0^{\beta_i}$. Una vez que conocemos x_0 (el cual es la salida de $\mathcal{R}$), podemos calcular

$$x_i = s_i - h_i(\alpha_i + x_0\beta_i) \bmod q,$$

el cual es el logaritmo discreto de r_i, $i = 1, \ldots, n$. Conocemos ahora los logaritmos discretos de todos los $n+1$ valores $r_0, \ldots, r_n$. Esto completa la demostración.

Paillier y Vergnaud probaron resultados similares para otros esquemas de firma basados en el DLP, tales como DSA y ECDSA. En los dos últimos casos, los autores se vieron forzados a modificar ligeramente el n-DLP: Para diferentes bases g_i, el oráculo del logaritmo discreto es capaz de dar el logaritmo discreto solicitado.

Intuitivamente, el problema "DLP uno-muchos" parece ser equivalente con el problema DLP, aun cuando hay una reducción obvia solamente en una dirección. Así, los resultados Paillier–Vergnaud podrían ser parafraseados como sigue: *Una reducción desde el DLP a falsificación es improbable a menos que el DLP sea fácil.* En este sentido el teorema de arriba deja el

mismo sabor de boca que el resultado de Boneh y Venkatesan [13] discutido en §2. Como en ese caso, una posible interpretación del resultado Paillier–Vergnaud es que puede existir una debilidad en la seguridad de firmas del tipo Schnorr. Ciertamente, esa interpretación es sugerida por el título "Las Firmas del Logaritmo Discreto Podrían no ser Equivalentes al Logaritmo Discreto"[20] y por el comentario en la introducción que afirma: "nuestro trabajo refuta que Schnorr, ElGamal, DSA, GQ, etc. sean esquemas con seguridad máxima."[21]

Por otro lado, como en §2, una explicación alternativa es que su trabajo da una nueva demostración de las limitaciones de los argumentos reduccionistas. Resulta instructivo comparar el resultado negativo de Paillier-Vergnaud con respecto a la existencia de reducciones, con los siguientes dos resultados positivos para reducciones de seguridad en los esquemas de firmas Schnorr.

8.3. Reducciones de Oráculo Aleatorio. *Argumento Reduccionista de securidad.* En el esquema de firma Schnorr, si la función picadillo es modelada por un oráculo aleatorio, entonce el DLP se reduce a falsificación universal.

Argumento. Suponga que un adversario puede falsificar la firma de m. Después que él obtiene $h = H(m, r)$, suponga que de pronto se le da una segunda función picadillo H'. Puesto que una función picadillo no tiene propiedades especiales de las que pueda beneficiarse el falsificador, cualquier método de ataque que él esté utilizando funcionará igual de bien con H reemplazada por H'. En otras palabras, estamos usando el modelo del oráculo aleatorio para la función picadillo. Así que el falsificador usa $h' = H'(m, r)$ tan bien como $h = H(m, r)$ y produce dos firmas válidas (h, s) y (h', s') para m, con el mismo r pero con $h' \neq h$. Note que puesto que r no cambia, el valor de k es el mismo en ambos casos. Dividiendo la diferencia de los dos valores $s \equiv k + xh$ y $s' \equiv k + xh' \pmod{q}$ por $h' - h$, uno puede utilizar directamente la salida del falsificador para hallar de manera inmediata el logaritmo discreto de x.[22]

El argumento dado arriba es impreciso. Estrictamente hablando, deberíamos considerar la posibilidad que el falsificador obtenga $H(m, r)$ para diversos valores de r y que firme únicamente uno de ellos. En tal caso, debemos adivinar cuál valor será firmado, y deberemos ejecutar el programa falsificador el número de veces que sea necesario para que, aleatoriamente, escojamos la opción correcta. Describimos un argumento riguroso (para una versión más formal de la afirmación de arriba) en §5 de [27], y para una discusión completa puede consultarse [33, 34].

Note que la necesidad de ejecutar el programa falsificador muchas veces conduce a una reducción no estrecha. En [34] se muestra que es suficiente

[20] "A reduction from the DLP to forgery is unlikely unless the DLP is easy", en su título original.

[21] Paillier y Vergnaud sí reconocen, sin embargo, que su trabajo no conduce a "ningún ataque o debilidad en concreto en ninguno de esos esquemas de firma."

[22] Note que no es necesario conocer k.

ejecutar al falsificador aproximadamente q_h veces, donde q_h es una cota del número de peticiones picadillo. En [32] Paillier y Vergnaud prueban que, grosso modo, una reducción algebraica en el modelo del oráculo aleatorio no puede ser más estrecha que $\sqrt{q_h}$. De una manera análoga al procedimiento de Coron en el caso de firmas RSA, Paillier y Vergnaud establecen una cota inferior en la apretadura de la reducción.

¿Cómo podemos aceptar el aparente hecho que, desde el DLP a la falsificación no es posible hacer una reducción estrecha en el modelo del oráculo aleatorio, y más aún, que en el modelo estándar es muy probable que no exista reducción de ningún tipo? Como de costumbre, varias interpretaciones son posibles. Quizás esto muestre que las reducciones en el modelo del oráculo aleatorio son peligrosas, porque ellas nos llevan a resultados de seguridad que no pueden ser alcanzados en el modelo estándar. Por otro lado, tal vez podemos concluir que el modelo del oráculo aleatorio debería ser usado, porque frecuentemente se acerca más a alcanzar lo que nuestra intuición sugiere que debería ser posible. ¿Y qué se puede decir de la holgura? deberíamos ignorarla, o deberíamos ajustar nuestras recomendaciones para tamaños de clave así que tengamos, digamos 80 bits de seguridad, después de haber tomado en cuenta el factor de holgura?

8.4. Resultado de Brown para ECDSA. Finalmente, discutimos otro resultado positivo con respecto a ECDSA. Enunciaremos sin probar una versión informal del teorema de D. Brown [14, 15].

Teorema. Suponga que la curva elíptica es modelada por un grupo genérico. Entonces el problema de encontrar una colisión para la función picadillo se reduce a falsificación de firmas ECDSA

El teorema de Brown está fuera del marco general al que pertenecen los resultados en [32]. Se trata de una reducción no desde el DLL a falsificación, sino más bien desde hallazgo de colisiones hasta falsificación. Y es una reducción estrecha. Haciendo la suposición del grupo genérico, esencialmente se está suponiendo que el DLP es difícil (véase [36]). Si la función picadillo es resistente a colisiones, entonces la presunta dificultad atribuida al DLP (más precisamente a la suposición del grupo genérico) implica dificultad de falsificación. Sin embargo, en [14] no hay reducción desde el DLP a la falsificación. Tanto Brown como Paillier–Vergnaud hacen suposiciones similares acerca del grupo. Los últimos autores implícitamente suponen que el problema n-DLP es difícil, y suponen que una reducción trata al grupo de una "manera genérica," esto es, computa operaciones de grupo sin explotar ninguna característica especial del codificado de los elementos de grupo. De manera similar, Brown supone que el grupo de curvas elípticas actúa, para todos los propósitos prácticos, como un grupo genérico y que, en particular, el DLP es difícil.

Pero sus respectivas conclusiones se oponen una a la otra. Paillier y Vergnaud prueban que no es posible en el modelo estándar construir ningún

tipo de reducción y que no hay reducción estrecha en el modelo del oráculo aleatorio. Brown da una reducción estrecha — de una naturaleza diferente que las consideradas en [32] — la cual, sujeta a ciertas suposiciones, prueba la seguridad de ECDSA.

Pero entonces, ¿*Es* la falsificación de firmas del tipo Schnorr equivalente al DLP? La mejor respuesta que podemos dar es una cita a la famosa declaración hecha por un ex-presidente reciente de Estados Unidos: todo depende de lo que la definición de "es" *es*.[23]

9. Conclusiones

En su artículo de revisión del estado del arte "Por qué la seguridad de criptograma escogido es importante " [37],[24] Shoup explicó la racional que está detrás de la gran importancia que se le otorga a los argumentos reduccionistas de seguridad:

> Esta es la metodología preferida de la criptografía matemática moderna. Allí uno muestra con rigor matemático que cualquier oponente capaz de romper el criptosistema puede ser transformado en un programa eficiente que puede resolver el muy estudiado problema subyacente (por ejemplo, factorización de números grandes), ampliamente aceptado como un problema muy difícil. Invirtiendo este razonamiento: si la "suposición de alta dificultad" es correcta como se presume, entonces el criptosistema es seguro. Esta metodología es de lo mejor que podemos hacer. Si podemos probar seguridad de esta manera, entonces esencialmente eliminamos todos los posibles atajos, incluso algunos que *nosotros no hemos siquiera imaginado*. La única manera de atacar el criptosistema es un ataque total y frontal al problema difícil subyacente. Punto. (p. 15; énfasis en el original)

Posteriormente en [37] Shoup concluyó: "Hay criptosistemas disponibles que son prácticos y demostrablemente seguros, y hay muy pocas excusas para no usarlos." Uno de los dos sistemas por cuyo uso él advocaba, debido a que contaban con pruebas de seguridad, era RSA-OAEP [7].

Desafortunadamente, la historia no ha sido gentil con la opinión citada arriba sobre la imperturbable confianza en la confiabilidad de los resultados de la seguridad demostrable. En 2001, el mismo Shoup [38] encontró un error en la supuesta prueba de seguridad del OAEP general de Bellare y Rogaway. El mismo año, Manger [29] montó un exitoso ataque de cripto-texto escogido

[23]El contexto de esta respuesta fue una explicación de una declaración anterior suya en la que dijo: "there *is* no sexual relationship with Ms. Lewinsky." (*la relación sexual con la señorita Lewinsky, no es*) Una afirmación en el tenor que "Una relación de equivalencia entre el DLP y la falsificación de firmas basadas en logaritmo discreto, no *es*" es, a nuestro juicio, igualmente inverosímil.

[24]El título original es: "Why chosen ciphertext security matters" [N. del T.]

en contra de RSA-OAEP. Curiosamente, no fue la falla en la prueba de Bellare–Rogaway (la cual fue posteriormente corregida en RSA-OAEP) la que hizo posible el ataque de Manger. Más bien, Manger encontró un atajo que en 1998, cuando Shoup escribió su artículo de revisión, no había sido "siquiera imaginado".

Muy frecuentemente es difícil determinar el significado, si alguno, que un argumento reduccionista de seguridad podría tener para la criptografía práctica. En años recientes, los investigadores se han vuelto más conscientes de lo importante de hacer un análisis concreto de sus reducciones. Pero mientras que a través de un largo y doloroso proceso logran encontrar desigualdades precisas, raramente hacen algún esfuerzo para explicar cómo sus resultados, matemáticamente precisos, podrían ser utilizados en la práctica.

Por ejemplo, en [1], los autores construyen cierto tipo de sistema de intercambio de claves basado en contraseña y dan pruebas de seguridad en el modelo del oráculo aleatorio fundamentadas en la dificultad del problema computacional Diffie–Hellman (CDH por sus siglas en inglés). Copiamos abajo el (ligeramente editado) texto de su resultado básico:

(Corolario 1 del Teorema 1, pp. 201-202 de [1]) que establece la relación entre la "ventaja" de un adversario para romper su protocolo SPAKE1 y la ventaja de un adversario para resolver el CDH:

> *Corolario 1.* Sea G un grupo representante de orden p, y sea $\mathcal{D}$ un diccionario uniformemente distribuido de tamaño $|\mathcal{D}|$. Sea SPAKE1 el protocolo de intercambio cifrado de claves basado en contraseña asociado con esas primitivas. Entonces, para cualesquiera números t, q_{start}, q^A_{send}, q^B_{send}, q_H, q_{exe}:
>
> $$\text{Adv}^{\text{ake}}_{\text{SPAKE},\mathcal{D}}(t, q_{\text{start}}, q^A_{\text{send}}, q^B_{\text{send}}, q_H, q_{\text{exe}})$$
>
> $$\leq 2 \cdot \left(\frac{q^A_{\text{send}} + q^B_{\text{send}}}{|\mathcal{D}|} + \sqrt[6]{\frac{2^{14}}{|\mathcal{D}|^2}\text{Adv}^{\text{CDH}}_G(t') + \frac{2^{15} q_H^4}{|\mathcal{D}|^2 p}} \right)$$
>
> $$+2 \cdot \left(\frac{(q_{\text{exe}} + q_{\text{send}})^2}{2p} + q_H \text{Adv}^{\text{CDH}}_G(t + 2q_{\text{exe}}\tau + 3\tau) \right),$$
>
> donde q_H representa el número de solicitudes al oráculo H; q_{exe} representa el número de solicitudes al oráculo Ejecutar; q_{start} y q^A_{send} representan el número de solicitudes al oráculo Enviar con respecto al iniciador A; q^B_{send} representa el número de solicitudes al oráculo Enviar con respecto al respondedor B;; $q_{\text{send}} = q^A_{\text{send}} + q^B_{\text{send}} + q_{\text{start}}$; $t' = 4t + O((q_{\text{start}} + q_H)\tau)$; y τ es el tiempo para calcular una exponenciación en G.

El artículo [1] incluye una prueba de esta descorcentante y más bien intimidante desigualdad. Pero el artículo no da ninguna indicación de qué significado, si alguno, podría tener en la práctica. El lector que pudiera querer usar el protocolo y quisiera encontrar los parámetros que satisfacen garantías

de seguridad y que al mismo tiempo permiten obtener una implementación razonablemente eficiente es dejado a su propia suerte.

En la literatura de seguridad demostrable el desafortunado lector tiene cada vez más posibilidades de encontrarse con complicadas desigualdades que involucran más de media docena variables. (Para otros ejemplos, véase el Teorema 5 en [28] y Teoremas 2 y 3 en [4].) El significado práctico de estas desigualdades casi nunca es explicado. Ciertamente, uno no puede sino preguntarse cuál es el propósito de publicarlas de una forma tan elaborada e indigerible sin dar ninguna interpretación. Cualquiera haya sido la intención de los autores, puede haber muy poca duda que el efecto no es el de instruir a sus lectores, sino más bien el de causarles un sentimiento de confusa fascinación.

* * *

Una vez embarcados en el estudio del campo de "seguridad demostrable," más pronto que tarde uno empieza a sentir que se ha adentrado en un reino que podría haber sido imaginado sólo por Lewis Carroll, y que la Alicia de fama criptográfica se ha fusionado con la heroína de los libros de Carroll:

> Alicia quedó completamente desconcertada. El comentario del sombrerero parecía no tener ningún sentido, y sin embargo, fue ciertamente dicho en perfecto inglés.[25] (*Alicia en el país de las Maravillas* y *Alicia a través del Espejo*, Londres: Oxford Univ. Press, 1971, p. 62.)

El lirón proclama que su generador de bits aleatorios es demostrablemente seguro ante un adversario cuyo poder computacional está acotado por un número negativo. El sombrerero responde que él tiene un generador que es demostrablemente seguro ante un adversario cuyos recursos computacionales están acotados por 10^{-40} ciclos de reloj. El caballero blanco es anunciado por heraldos, pero después de un segundo análisis uno nota que cabalga hacia atrás. El comité de programa está conformado por reinas rojas que, cuando encuentran autores que someten algún artículo sin un teorema de seguridad demostrable, gritan "¡Qué les corten las cabezas!".

La Alicia de Lewis Carroll despierta al final del libro y se da cuenta que todo ha sido sólo un sueño. Para la Alicia criptográfica empero, puede que el regreso al mundo real no sea tan fácil.

Agradecimientos

Quisieramos agradecer a Andrey Sidorenko por sus valiosos comentarios sobre generadores de bit pseudo-aleatorios y a Bart Preneel por contestar nuestras preguntas sobre seguridad demostrable de los algoritmos MAC.

[25] "Alice felt dreadfully puzzled. The Hatter's remark seemed to her to have no sort of meaning in it, and yet it was certainly English.", en el original.

También deseamos agradecer a Ian Blake y Dan Brown por leer y comentar los borradores previos a este manuscrito. No hace falta señalar que las opiniones expresadas en este artículo son de la sola responsabilidad de los autores

Referencias

[1] M. Abdalla and D. Pointcheval, Simple password-based encrypted key exchange protocols, *Topics in Cryptology – CT-RSA 2005*, LNCS 3376, Springer-Verlag, 2005, pp. 191-208.

[2] M. Ajtai and C. Dwork, A public-key cryptosystem with worst-case/average-case equivalence, *Proc. 29th Symp. Theory of Computing*, A.C.M., 1997, pp. 284-293.

[3] W. Alexi, B. Chor, O. Goldreich, and C. P. Schnorr, RSA and Rabin functions: Certain parts are as hard as the whole, *SIAM J. Computing*, **17** (1988), pp. 194-209.

[4] P. Barreto, B. Libert, N. McCullagh, and J.-J. Quisquater, Efficient and provably-secure identity-based signatures and signcryption from bilinear maps, *Advances in Cryptology – Asiacrypt 2005*, LNCS 3788, Springer-Verlag, 2005, pp. 515-532.

[5] M. Bellare, Practice-oriented provable-security, *Proc. First International Workshop on Information Security (ISW '97)*, LNCS 1396, Springer-Verlag, 1998, pp. 221-231.

[6] M. Bellare and P. Rogaway, Random oracles are practical: a paradigm for designing efficient protocols, *Proc. First Annual Conf. Computer and Communications Security*, ACM, 1993, pp. 62-73.

[7] M. Bellare and P. Rogaway, Optimal asymmetric encryption — how to encrypt with RSA, *Advances in Cryptology – Eurocrypt '94*, LNCS 950, Springer-Verlag, 1994, pp. 92-111.

[8] S. Blackburn and K. Paterson, Cryptanalysis of a message authentication code due to Cary and Venkatesan, *Fast Software Encryption 2004*, LNCS 3017, Springer-Verlag, 2004, pp. 446-453.

[9] L. Blum, M. Blum, and M. Shub, A simple unpredictable pseudo-random number generator, *SIAM J. Computing*, **15** (1986), pp. 364-383.

[10] M. Blum and S. Micali, How to generate cryptographically strong sequences of pseudo-random bits, *SIAM J. Computing*, **13** (1984), pp. 850-864.

[11] D. Boneh and X. Boyen, Short signatures without random oracles, *Advances in Cryptology – Eurocrypt 2004*, LNCS 3027, Springer-Verlag, 2004, pp. 56-73.

[12] D. Boneh, B. Lynn, and H. Shacham, Short signatures from the Weil pairing, *Advances in Cryptology – Asiacrypt 2001*, LNCS 2248, Springer-Verlag, 2001, pp. 514-532.

[13] D. Boneh and R. Venkatesan, Breaking RSA may not be equivalent to factoring, *Advances in Cryptology – Eurocrypt '98*, LNCS 1233, Springer-Verlag, 1998, pp. 59-71.

[14] D. Brown, Generic groups, collision resistance, and ECDSA, *Designs, Codes and Cryptography*, **35** (2005), pp. 119-152.

[15] D. Brown, On the provable security of ECDSA, in I. Blake, G. Seroussi, and N. Smart, eds., *Advances in Elliptic Curve Cryptography*, Cambridge University Press, 2005, pp. 21-40.

[16] D. Brown, Breaking RSA may be as difficult as factoring, http://eprint.iacr.org/2005/380

[17] D. Brown, comunicación personal no publicada, Febrero de 2006.

[18] M. Cary and R. Venkatesan, A message authentication code based on unimodular matrix groups, *Advances in Cryptology – Crypto 2003*, LNCS 2729, Springer-Verlag, 2003, pp. 500-512.

[19] J.-S. Coron, On the exact security of full domain hash, *Advances in Cryptology – Crypto 2000*, LNCS 1880, Springer-Verlag, 2000, pp. 229-235.

[20] J.-S. Coron, Optimal security proofs for PSS and other signature schemes, *Advances in Cryptology – Eurocrypt 2002*, LNCS 2332, Springer-Verlag, 2002, pp. 272-287.
[21] D. Eastlake, S. Crocker, and J. Schiller, RFC 1750 – Randomness Recommendations for Security, disponible en: http://www.ietf.org/rfc/rfc1750.txt
[22] R. Fischlin and C. P. Schnorr, Stronger security proofs for RSA and Rabin bits, *J. Cryptology*, **13** (2000), pp. 221-244.
[23] R. Gennaro, An improved pseudo-random generator based on the discrete log problem, *J. Cryptology*, **18** (2005), pp. 91-110.
[24] N. Howgrave-Graham, J. Dyer, and R. Gennaro, Pseudo-random number generation on the IBM 4758 Secure Crypto Coprocessor, *Workshop on Cryptographic Hardware and Embedded Systems* (CHES 2001), LNCS 2162, Springer-Verlag, 2001, pp. 93-102.
[25] J. Katz and N. Wang, Efficiency improvements for signature schemes with tight security reductions, *10th ACM Conf. Computer and Communications Security*, 2003, pp. 155-164.
[26] D. Knuth, *Seminumerical Algorithms*, vol. 2 of *Art of Computer Programming*, 3rd ed., Addison-Wesley, 1997.
[27] N. Koblitz and A. Menezes, Another look at "provable security," to appear in *J. Cryptology*; disponible en: http://eprint.iacr.org/2004/152.
[28] P. Mackenzie and S. Patel, Hard bits of the discrete log with applications to password authentication, *Topics in Cryptology – CT-RSA 2005*, LNCS 3376, Springer-Verlag, 2005, pp. 209-226.
[29] J. Manger, A chosen ciphertext attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as standardized in PKCS #1 v2.0, *Advances in Cryptology – Crypto 2001*, LNCS 2139, Springer-Verlag, 2001, pp. 230-238.
[30] P. Q.Ñguyen and J. Stern, Cryptanalysis of the Ajtai–Dwork cryptosystem, *Advances in Cryptology – Crypto '98*, LNCS 1462, Springer-Verlag, 1998, pp. 223-242.
[31] P. Q.Ñguyen and J. Stern, The two faces of lattices in cryptology, *Cryptography and Lattices – Proc. CALC 2001*, LNCS 2146, Springer-Verlag, 2001, pp. 146-180.
[32] P. Paillier and D. Vergnaud, Discrete-log-based signatures may not be equivalent to discrete log, *Advances in Cryptology – Asiacrypt 2005*, LNCS 3788, Springer-Verlag, 2005, pp. 1-20.
[33] D. Pointcheval and J. Stern, Security proofs for signature schemes, *Advances in Cryptology – Eurocrypt '96*, LNCS 1070, Springer-Verlag, 1996, pp. 387-398.
[34] D. Pointcheval and J. Stern, Security arguments for digital signatures and blind signatures, *J. Cryptology*, **13** (2000), pp. 361-396.
[35] C. P. Schnorr, Efficient signature generation for smart cards, *J. Cryptology*, **4** (1991), pp. 161-174.
[36] V. Shoup, Lower bounds for discrete logarithms and related problems, *Advances in Cryptology – Eurocrypt '97*, LNCS 1233, Springer-Verlag, 1997, pp. 256-266.
[37] V. Shoup, Why chosen ciphertext security matters, IBM Research Report RZ 3076 (#93122) 23/11/1998.
[38] V. Shoup, OAEP reconsidered, *Advances in Cryptology – Crypto 2001*, LNCS 2139, Springer-Verlag, 2001, pp. 239-259.
[39] A. Sidorenko, comunicación personal no publicada, Marzo de 2006.
[40] A. Sidorenko and B. Schoenmakers, Concrete security of the Blum–Blum–Shub pseudorandom generator, *Cryptography and Coding 2005*, LNCS 3796, Springer-Verlag, 2005, pp. 355-375.
[41] U. V. Vazirani and V. V. Vazirani, Efficient and secure pseudo-random number generation, *Proc. IEEE 25th Annual Symp. Foundations of Computer Science*, 1984, pp. 458-463.
[42] A. Yao, Theory and applications of trapdoor functions, *Proc. IEEE 23rd Annual Symp. Foundations of Computer Science*, 1982, pp. 80-91.

[43] A. Young and M. Yung, *Malicious Cryptography: Exposing Cryptovirology*, Wiley, 2004.

Department of Mathematics, Box 354350, University of Washington, Seattle, WA 98195 U.S.A.
E-mail address: koblitz@math.washington.edu

Department of Combinatorics & Optimization, University of Waterloo, Waterloo, Ontario N2L 3G1 Canada
E-mail address: ajmeneze@uwaterloo.ca