EL SISTEMA RSA

Pablo Mariño Boga
2 de Enero 2017

TABLA DE CONTENIDO

En este documento pretendo ofrecer una visión general del criptosistema RSA, su funcionamiento, vectores de ataque más comunes y problemas a los que se deberá enfrentar en el futuro.

INTRODUCCIÓN

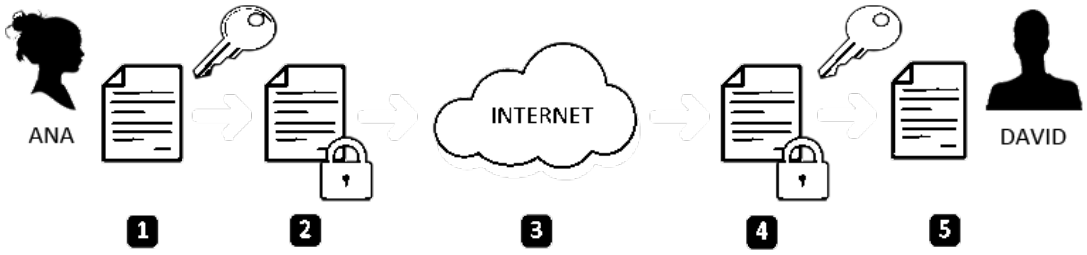
El algoritmo RSA fue descrito en 1977 por Ron Rivest, Adi Shamir y Leonard Adleman, del Instituto Tecnológico de Massachusetts y patentado en 1983 por el MIT. Su patente expiró el 21 de septiembre del 2000.
A punto de cumplir cuarenta años el RSA sigue siendo una pieza clave en los sistemas de comunicación y comercio, proporciona un método de cifrado y firma digital asequible y que hasta el momento ha sabido mantener un alto grado de seguridad.

FUNCIONAMIENTO

RSA se basa en principios de la teoría elemental de números conocidos desde hace más de 250 años. Es un algoritmo de criptografía asimétrica, usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona que ha enviado el mensaje. Una clave es *pública* y se puede entregar a cualquier persona, la otra clave es *privada* y el propietario debe guardarla de modo que nadie tenga acceso a ella.

Si el remitente usa la clave pública del destinatario para cifrar el mensaje, una vez cifrado, sólo la clave privada del destinatario podrá descifrar este mensaje (confidencialidad).

Si el propietario del par de claves usa su clave privada para cifrar el mensaje, cualquiera puede descifrarlo utilizando su clave pública (*identificación* y *autentificación* del remitente). Esta idea es el fundamento de la *firma electrónica*.



1. Ana redacta un mensaje
2. Ana cifra el mensaje con la **clave pública** de David
3. Ana envía el mensaje cifrado a David a través de internet, ya sea por correo electrónico, mensajería instantánea o cualquier otro medio
4. David recibe el mensaje cifrado y lo descifra con su **clave privada**
5. David ya puede leer el mensaje original que le mandó Ana

Figura 1. Cifrado Descifrado RSA. Fuente:https://es.wikipedia.org

GENERACIÓN DE CLAVES

Cada usuario del sistema RSA dispone de un par de claves publica/secreta, las claves se generan en tres pasos.

1. Módulo: Se eligen dos números primos (p, q) y se calcula $n = p \cdot q$
2. Clave Pública: Se elige un número e primo con $\varphi(n)$*
3. Clave Privada: Se calcula d que cumpla que $e \cdot d \equiv 1$** congruente con módulo $\varphi(n)$

---

* $\varphi(n) = (p - 1) \cdot (q - 1)$
** d es e^{-1} en el espacio de módulo $\varphi(n)$

Se han definido recomendaciones para la elección del módulo y las claves. Con esto se busca evitar ataques elementales por debilidades conocidas del RSA. Veamos cómo se generan esas claves y las principales recomendaciones.

El primer lugar para generar la clave pública se eligen dos números primos p y q.

- Empecemos por la longitud de p y q, actualmente es habitual elegir dos primos de manera aleatoria, ambos del mismo tamaño 1024 bits (309 dígitos) o superior y someterlos a un test de primalidad (*Miller-Rabin*) con esto nos aseguramos de que la probabilidad de que un entero positivo sea un falso primo es despreciable

- La distancia entre los dos primos seleccionados ha de ser significativa, en caso contrario el valor que obtendríamos de calcular $y = \frac{q-p}{2}$ con $p < q$ seria pequeño y como además se cumple que $x = \frac{q+p}{2} \approx \sqrt{n}$ se cumpliría la igualdad $x^2 - n = y^2$ solo quedaría probar valores de $x > \sqrt{n}$ hasta encontrar uno valido y tendríamos los factores de n.

- $(p-1)$, $(q-1)$ han de contener factores primos grandes, de no ser así los factores primos de $\varphi(n)$ también serían pequeños y sería sencillo construir todos los candidatos v a ser $\varphi(n)$ y comprobar si el texto cifrado elevado $\frac{(v+1)}{e}$ proporciona un mensaje comprensible.

Una vez elegidos (p, q) de su producto se obtiene n, el módulo de la clave pública y privada, su tamaño en bits será el doble de cualquiera de los primos.

El trabajo realizado hasta este punto es asequible la seguridad de la encriptación RSA depende de la dificultad de factorizar n para obtener de nuevo (p, q) el coste computacional que implica lo hace inviable.

A continuación se elige exponente de la clave pública e:

- e se va a utilizar como exponente de cifrado $m^e \bmod n$, para facilitar la tarea de cifrar el mensaje es recomendable que e no sea demasiado grande.

- Se recomienda que e sea 3 o 65537, porque ambos son números primos, su expresión en binario facilita los cálculos necesarios para cifrar el mensaje.
  Para $e = 3_{(10)} = 2^1 + 1 = 11_{(2)}$
  $$m^3 = m \cdot m \cdot m$$
  Para $e = 65537_{(10)} = 2^{16} + 1 = 10000000000000001_{(2)}$
  $$m^{65537} = m^{2^{16}+1} = m^{2^{16}} * m$$
  para calcular $m^{2^{16}} (\bmod n)$ basta calcular m^2 e ir elevando el resultado al cuadrado (15 veces) con lo que obtendríamos el resultado tras 17 multiplicaciones.

- Debe evitarse e pequeño cuando:
  - El mismo mensaje se va a enviar a varios destinatarios, se podría plantear un sistema de congruencias que utilice el módulo de cada usuario y el mensaje cifrado
    $$m^e \bmod n_1 = c_1$$
    $$m^e \bmod n_2 = c_2$$
    $$m^e \bmod n_3 = c_3$$
    que se resolvería con el *Teorema chino del resto* para obtener el mensaje original.
  - O se van a cifrar mensajes cortos, en caso de que $m < n^{\frac{1}{e}}$ sería posible recuperar m usando la expresión $c = m^e \ (mod\ n)$ de la que obtenemos $m = \sqrt[e]{C} \in \mathbb{N}$

- Evitar e común, si varios usuarios van a compartir el exponente de cifrado, entonces el módulo n ha de ser diferente, sino un usuario utilizando sus claves de cifrado y descifrado e, d podría conocer la clave de descifrado de los demás.

Ahora que tenemos la clave pública necesitamos obtener la clave privada, mediante $e \cdot d \equiv 1$ o $e \equiv d^{-1}$ módulo $\varphi(n)$, podemos calcular d que sería el exponente de la clave privada.

CIFRADO

La aplicación de cifrado se define como:

$$E : \mathbb{Z}_n \to \mathbb{Z}_n \ , \ x \to x^e$$

Para cifrar un mensaje que se va a enviar a un usuario solo es necesario conocer la clave pública ***(n, e)***.

Supongamos que ***A*** envía un mensaje ***M*** confidencialmente a ***B*** a través de un medio de transmisión no seguro, utilizando RSA para obtener confidencialidad siguiendo estos pasos:

1. Obtiene la clave pública del destinatario ***B***, (n_b , e_b)
2. Representa el texto a transmitir como un entero positivo $M < n$
3. Cifra el mensaje $C = (M)^{e_b}$ modulo n_b
4. Finalmente transmite el mensaje cifrado ***C*** por el canal no seguro

DESCIFRADO

La aplicación de descifrado se define como

$$D : \mathbb{Z}_n \to \mathbb{Z}_n \ , \ x \to x^d$$

Para descifrar un texto o crear firma es necesario disponer de la clave privada (***n***, ***d)***.

Cuando ***B*** reciba el mensaje cifrado ***C***, hace lo siguiente:

1. Usa su clave privada (d_b, n_b) para computar $M = (C)^{d_b}$ módulo n_b
2. Recupera el texto original a partir de **M** que se había codificado en origen como un entero

FIRMA DIGITAL

Supongamos ahora que ***A*** quiere enviar un mensaje a ***B***, el mensaje no tiene necesariamente por que ir cifrado, pero ***A*** está interesado en "firmar" el mensaje de forma que ***B*** pueda estar seguro que el mensaje que le llega ha sido originado por ***A,*** los pasos que ***A*** seguirá son:

1. Crea un resumen del mensaje que quiere enviar, utilizando una *función* hash
2. Representar el texto a transmitir como un entero positivo $M < n$

3. Usar su clave privada (d_a, n_a) para calcular la firma $S = (M)^{d}{}_{a}$ modulo n_a
4. Enviar la firma ***S*** al receptor ***B*** con el mensaje original, en destino el receptor se encargara de verificar la firma y el mensaje con lo que ninguno de los dos puede ser modificados durante la transmisión.

Cuando ***B*** recibe la firma **S** y el mensaje de ***A***, sigue estos pasos:

1. Utiliza la clave pública de ***A*** para calcular $\mathbf{V} = (S)^{e}{}_{a}$ modulo n_a
2. Del entero ***V*** obtiene el resumen ***r*** que calculo *A*.
3. Al mismo tiempo usando el mensaje y aplicando la misma función de hash que ***A*** calcula ***r'*** la compara con ***r*** y comprueba que sean iguales

Los ataques al criptosistema se pueden clasificar en tres tipos, en primer lugar los ***ataques elementales*** que dependen de un uso poco seguro, o de la incorrecta implementación del criptosistema, para evitarlo se han creado recomendaciones como las que hemos visto en el apartado de generación de claves.

En segundo lugar existen ataques más sofisticados que se basan en ***algoritmos de factorización*** rápida y heurísticas para resolver el problema.

Y por último ataques que se basan en implementaciones especiales de **hardware**

ATAQUES ELEMENTALES

ATAQUE DE MODULO COMÚN

Si el mismo mensaje se envía a dos usuarios que comparten modulo $\boldsymbol{n}$ y con exponentes de cifrado $\boldsymbol{e_a}$ y $\boldsymbol{e_b}$ primos entre ellos, cada usuario puede descifrar el mensaje, y determinar la clave privada del otro.

Dos usuarios $\boldsymbol{A}$ y $\boldsymbol{B}$ de RSA comparten el mismo modulo $\boldsymbol{n}$, sus claves publicas serian:

$$\boldsymbol{A}\ (\ \boldsymbol{n}\ ,\ \boldsymbol{e_a}\ )\ ,\ \boldsymbol{B}\ (\ \boldsymbol{n}\ ,\ \boldsymbol{e_b}\ )$$

Digamos que $\boldsymbol{A}$ y $\boldsymbol{B}$ reciben un mensaje $\boldsymbol{m} \in \mathbb{Z}_n$ que se cifra $\boldsymbol{c_i} = m^{e_i}$ donde $\boldsymbol{i = a,\ b.}$
Dado que $\boldsymbol{A}$ y $\boldsymbol{B}$ son primos entre sí es posible descifrar el texto utilizando $\boldsymbol{c_i}$, $\boldsymbol{e_i}$ y $\boldsymbol{n}$.
Además utilizando el algoritmo de Euclides, $\boldsymbol{r \cdot e_a + s \cdot e_b = 1}$ podemos obtener los enteros $\boldsymbol{r}$ y $\boldsymbol{s}$, después de lo que solo quedaría resolver $\boldsymbol{m = c_1^r \cdot c_2^s\ mod(n)}$ para obtener el mensaje original.

La probabilidad de que dos usuarios compartan modulo si los factores primos $\boldsymbol{p}$, $\boldsymbol{q}$ se eligen aleatoriamente son insignificantes.

ATAQUES POR PARADOJA DEL CUMPLEAÑOS

> ¿cuál crees que sería la probabilidad de que en un grupo de 23 personas haya al menos dos que cumplan el mismo día? (…) si hay 23 personas reunidas hay una probablidad del 50,7% de que al menos dos personas de ellas cumplan años el mismo día. Para 60 o más personas la probabilidad es mayor del 99%. (…) (Wikipedia, paradoja del cumpleaños)

Si una función matemática ***f(x)*** genera ***n*** resultados diferentes, probabilísticamente sólo necesitamos evaluar la función para $\mathbf{2}\sqrt{\boldsymbol{n}}$ valores hasta un valor que habíamos obtenido con anterioridad se repita.

En definitiva, la idea es que si tenemos una función ***f(m,k)=c***, siendo m el mensaje que se quiere cifrar, k la clave y c el mensaje cifrado, es posible encontrar un $\boldsymbol{k'}$ tal que ***f(m,k')=c***. Por la paradoja del cumpleaños sabemos que probando al azar hay una probabilidad mayor a la que podríamos esperar de encontrar dicho $\boldsymbol{k'}$.

ATAQUES POR CIFRADO CÍCLICO

> Nos permite descifrar un criptograma utilizando únicamente la clave pública que se ha utilizado para el cifrado del mensaje.

Trabajamos dentro de un cuerpo finito, por lo tanto al multiplicar un valor por otro repetidamente, acabaremos obteniendo el resultado inicial.

Si un usuario ***A*** cifra un mensaje $\boldsymbol{c = m^e \ mod(n)}$ para enviárselo a un usuario ***B***, un tercer usuario puede obtener el mensaje cifrado y cifrarlo repetidamente usando la clave pública l usuario ***B***. En algún momento obtendrá de nuevo el valor ***c***, y el valor que cifrado en el paso anterior corresponderá al mensaje ***m***. Se trata de un ataque lento para claves de tamaño grande.

ATAQUES POR FACTORIZACIÓN ENTERA

Existen algoritmos de factorización que permitirían obtener ***(p, q)*** a partir de ***n***

Existen varios algoritmos de factorización de propósito general y específico (*Criba de Eratóstenes, Método de Euler, Criba Cuadrática, Método de las fracciones continuas …*) el que ha demostrado mayor eficacia de media es la *Criba numérica (General Number Field Sieve GNFS).*

Aun así su complejidad computacional es de $\boldsymbol{O = exp\left(\frac{64}{9}b\right)^{\frac{1}{3}} \cdot (logb)^{\frac{2}{3}}}$ para un numero de ***b*** bits. (El criptosistema RSA p.94)

La complejidad de la factorización es exponencial, en la siguiente tabla se puede ver como se pasa de un tiempo de 25 segundos para factorizar un numero de 60 dígitos (198 bits) a 18 horas y 40 minutos para un numero de 90 dígitos (297 bits).

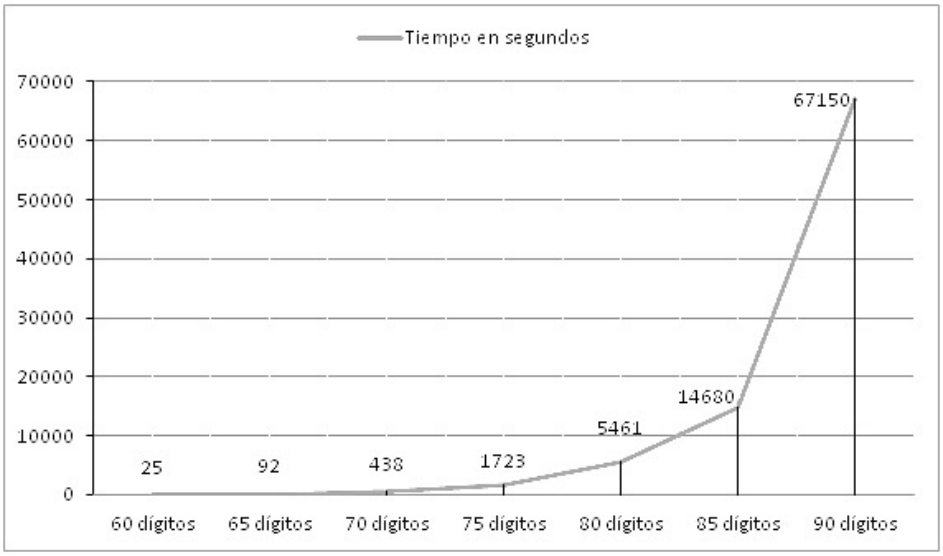


Figura 2. Gráfica del problema de la factorización entera. Fuente http://www.criptored.upm.es

Si los primos ***p*** y ***q*** son demasiado cercanos el algoritmo de Fermat sería muy eficiente y resolvería la factorización en pocos pasos.

Hasta la fecha el problema de la factorización de enteros tiene una clase de complejidad NP y una complejidad computacional exponencial pero al ser un problema abierto puede aparecer un nuevo algoritmo con un mejor comportamiento.

ATAQUES DEPENDIENTES DEL HARDWARE

TWINKLE

Dispositivo que acelera la fase de criba de números primos en la factorización del módulo.

En el Eurocrypt'99 Adi Shamir describió la construcción de un dispositivo de computación electro-óptico que utiliza el método de criba cuadrática para factorizar el modulo, este dispositivo permitía en ese momento acelerar la fase de criba de números entre un 500 y un 1000% sobre un los tiempos de un computador convencional. (Adi Shamir.1999. *Factoring Large Numbers with the TWINKLE Device p.5*)

El diseño se basa en un cilindro con una matriz de LEDs en un extremo y un fotosensor en el otro, cada uno de los leds se asocia a un periodo p_j y un retraso d_j, cada led se enciende en el momento determinado por la fórmula $p_j r+d_j$ con $r \geq 0$ además su intensidad corresponderá a $\log p_j$, superado cierto umbral de la intensidad del conjunto el sensor se dispara señalando la detección de un valor primo. (Adi Shamir.1999. *Factoring Large Numbers with the TWINKLE Device p.6*)

Este sistema permite comprobar al mismo tiempo unos 200000 números con una pérdida de precisión tolerable.

CRIPTOANÁLISIS ACÚSTICO

Se trata de un tipo de ataque que se basa en los *sonidos* emitidos por computadores o máquinas. El criptoanálisis acústico moderno está enfocado a los sonidos producidos por los *teclados* y los componentes internos del computador.

En 2004, Adi Shamir y Eran Tromer llevaron a cabo con éxito un análisis de operaciones criptográficas, usando un micrófono parabólico midieron los sonidos que producía un procesador al descifrar un mensaje con una implementación de RSA.

Las contramedidas para este tipo de ataques son sencillas, pasan por generar sonidos en el mismo espectro de los que podría buscar un atacante generando variaciones para impedir que se pueda crear huellas digitales utilizando la transformada rápida de Fourier (FFT), o usar ruido blanco.

El problema de factorizar un número pasa a tener un tiempo polinómico utilizando un computador cuántico y el algoritmo de Shor.

El algoritmo de Shor permite pasar de un tiempo exponencial a un tiempo polinómico la factorización de un número N, la complejidad temporal pasa a ser ***O((log N)3)*** y espacial ***O(logN)***

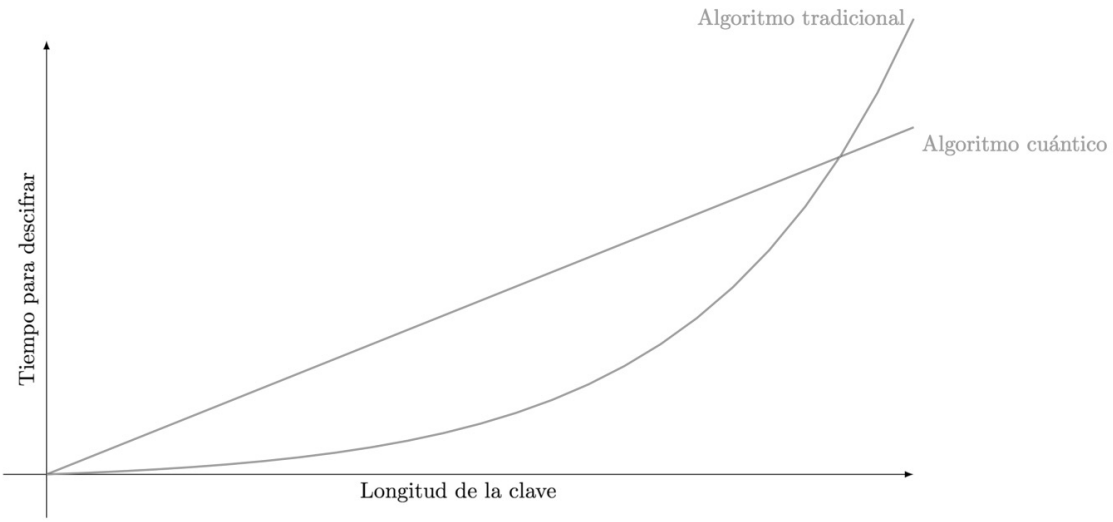


Figura 3. Algoritmo cuántico vs algoritmo tradicional. Fuente https://www.xataka.com

El algoritmo se compone de dos partes, la primera puede ser implementada en un computador tradicional, convierte el problema de descomponer en factores en el problema de encontrar el período de una función.

La segunda parte es la responsable de la mejora de la complejidad del problema, se implementa en un computador cuántico y su misión es encontrar el período de la función usando la transformada de Fourier cuántica y aprovechando la capacidad de una computadora cuántica de estar en muchos estados simultáneamente (superposición cuántica) lo que le permite evaluar la función en todos los puntos al mismo tiempo para obtener la solución.
La solución que se obtiene tiene una alta probabilidad de ser la correcta pero no la certeza así que debe comprobarse o repetir los cálculos y comprobar el resultado para asegurarse de que esa probabilidad aumente.

ALTERNATIVAS

CRIPTOGRAFIA POSTCUÁNTICA

¿Y ahora qué?... Algoritmos seguros, transmisión cuántica de claves…

La atención del mundo académico y la industria se centró en el problema que supone para el cifrado la computación cuántica a raíz de la serie de conferencias *PQCrypto* que se vienen celebrando desde 2006 y más recientemente por varios Talleres del European Telecommunications Standards Institute (ETSI) sobre *Criptografía Segura Cuántica*.

Se han planteado alternativas a los sistemas de cifrado de clave pública…

- Existen algoritmos que a día de hoy se consideran seguro ante el ataque de computadores cuánticos:
  Criptografía basada en sistemas de curvas elípticas
  Criptografía basada en identidad (Hash)
  Criptografía basada en códigos correctores
  (…)

- Transmisión cuántica de claves, ya que no es posible asegurar el mensaje, se puede tratar de asegurar el canal de comunicación, utilizando partículas en un estado entrelazado, la interacción con una de estas partículas puede cambiar el estado de ambas, se basa en la paradoja EPR y se plantea como el futuro medio de comunicación ideal llamado a sustituir nuestras actuales redes superando las limitaciones espacio-temporales de las señales radioeléctricas.

General

· Delfs, H., & Knebl, H. (2007). *Introduction to cryptography: principles and applications*. Berlín: Springer.

· Raul Durán, Luis Hernandez, Jaime Muñoz. (2005). *El criptosistema RSA*. Ra-ma editorial

· Jorge Ramió Aguirre. (2012). *Mooc el Algoritmo RSA.* [Online]. Descargado 12, 15, 2016, de http://www.criptored.upm.es/crypt4you/temas/RSA/leccion0/leccion00.html

· R.L. Rivest, A. Shamir, and L. Adleman. (1977). *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.* [Online]. Descargado 12, 15, 2016, de http://people.csail.mit.edu/rivest/Rsapaper.pdf

· Evgeny Milanov. *The RSA Algorithm*. (2009). [Online]. Descargado 12, 15, 2016, de https://www.math.washington.edu/~morrow/336_09/papers/Yevgeny.pdf

Ataques por paradoja del cumpleaños

· (n.d.). *La paradoja del cumpleaños*. [Online]. Descargado 12, 15, 2016, de http://www.estadisticaparatodos.es/taller/cumpleanos/cumpleanos.html

· Jorge Ramió Aguirre. 2013. *Ataque por paradoja del cumpleaños*. [Online]. Descargado 12, 15, 2016, de http://www.criptored.upm.es/crypt4you/temas/RSA/leccion10/leccion10.html

Ataque de modulo común

· (n.d.). (n.d.). *Ataques contra RSA*. [Online]. Descargado 12, 15, 2016, de http://paraisomat.ii.uned.es/paraiso/cripto.php?id=rsa3

· Alberto García Serrano, (2014). *Ataques y debilidades de RSA*. [Online]. Descargado 12, 15, 2016, de http://www.ellaberintodefalken.com/2014/04/ataques-debilidades-rsa.html

Ataques por cifrado cíclico

· Jorge Ramió Aguirre. (2012). *Ataque por cifrado cíclico*. [Online]. Descargado 12, 15, 2016, de http://www.criptored.upm.es/crypt4you/temas/RSA/leccion9/leccion09.html

Ataques por factorización entera

· (n.d.). (n.d.). *Ataques a DES y módulos factorizados de RSA* [Online]. Descargado 12, 15, 2016, de http://www.revistasic.com/revista40/pdf_40/SIC_40_agora.PDF

· Jorge Ramió Aguirre. (2012). *Ataque por factorización*. [Online]. Descargado 12, 15, 2016, de http://www.criptored.upm.es/crypt4you/temas/RSA/leccion8/leccion08.html

· Daniel Lerch Hostalot. (n.d.). *Ataque de factorización a RSA*. [Online]. Descargado 12, 15, 2016, de http://es.opendomo.org/dlerch/doc/rsa-es.pdf

Twinkle

· Adi Shamir, (1999) *Factoring Large Numbers with the TWINKLE Device*. [Online]. Descargado 12, 15, 2016, de http://www.dima.unige.it/~morafe/MaterialeCTC/twinkle.pdf

· Robert D. Silverman. (1999). *An Analysis of Shamir's Factoring Device.* [Online]. Descargado 12, 15, 2016, de http://www.ussrback.com/crypto/rsa/TWINKLE/twinkle.html

· Arjen K. Lenstra, Adi Shamir. (n.d.). *Analysis and Optimization of the TWINKLE Factoring Device*. [Online]. Descargado 12, 15, 2016, de http://www.iacr.org/archive/eurocrypt2000/1807/18070035-new.pdf

· (n.d.). (n.d.). *Ataques a DES y módulos factorizados de RSA*[Online]. Descargado 12, 15, 2016, de http://www.revistasic.com/revista40/pdf_40/SIC_40_agora.PDF

· (n.d.). (n.d.). *The TWIRL integer factorization device*. [Online]. Descargado 12, 15, 2016, de http://cs.tau.ac.il/~tromer/twirl/

Criptoanálisis acústico

· Daniel Genkin , Adi Shamir, Eran Tromer. (2013.). *RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis.* [Online]. Descargado 12, 15, 2016, de https://www.tau.ac.il/~tromer/papers/acoustic-20131218.pdf

· Roberto Sierra Cabrera. (n.d.). *Criptoanálisis acústico*. [Online]. Descargado 12, 15, 2016, de http://www.robota.net/index.rsws?seccion=5&submenu=1&articulo=1062

Computación cuántica

· (n.d.). (n.d.). *Algoritmo de Shor*. [Online]. Descargado 12, 15, 2016, de http://jorgemtz90.blogspot.com.es/2010/05/algoritmo-de-shor.html

· Francisco R. Villatoro. (2013). *Factorizan un número entero de 20.000 bits utilizando el algoritmo cuántico de Shor pero con "truco*". [Online]. Descargado 12, 15, 2016, de http://francis.naukas.com/2013/07/10/factorizan-un-numero-entero-de-20-000-bits-utilizando-el-algoritmo-cuantico-de-shor-pero-con-truco/

· Javier Jimenez. (2016). ¿El principio del fin de la criptografía actual? Crean la primera computadora cuántica escalable". [Online]. Descargado 12, 15, 2016, de https://www.xataka.com/investigacion/el-principio-del-fin-de-la-criptografia-actual-crean-la-primera-computadora-cuantica-esclable

Hernando Efraín Caicedo-Ortiz. (2010) *Algoritmo de factorización para un computador cuántico*. [Online]. Descargado 12, 15, 2016, de http://www.lajpe.org/may10/16_Hernando_Caicedo.pdf

Criptografía Postcuántica

· (n.d.). (n.d.). Criptografia Postcuántica.(i) [Online]. Descargado 12, 15, 2016, de http://revista.seguridad.unam.mx/numero-18/criptograf%C3%AD-cu%C3%A1ntica

· (n.d.). (n.d.). Criptografia Postcuántica.(ii) [Online]. Descargado 12, 15, 2016, de http://revista.seguridad.unam.mx/numero-19/criptograf%C3%AD-cu%C3%A1ntica-%E2%80%93-parte-ii

· Santiago Campillo. (2015). *Entrelazamiento, así funcionan la computación y la teleportación cuántica.* [Online]. Descargado 12, 15, 2016, de https://hipertextual.com/2015/09/entrelazamiento-cuantico

· (n.d.). (n.d.). *Post-quantum cryptography* [Online]. Descargado 12, 15, 2016, de https://en.wikipedia.org/wiki/Post-quantum_cryptography

· Luca De Feo, David Jao, Jerome Plût. (2011.). *Towards Quantum-resistent Criptosystems From Supersingular Elliptic Curve Isogenies* [Online]. Descargado 12, 15, 2016, de http://eprint.iacr.org/2011/506.pdf

· (n.d.). (2015.). *Rank based Cryptography: a credible post-quantum alternative to classical crypto*[Online]. Descargado 12, 15, 2016, de http://csrc.nist.gov/groups/ST/post-quantum-2015/papers/session1-gaborit-paper.pdf