CIFRADO RSA

Este documento no pretende ser un curso exhaustivo, en el mejor de los casos, únicamente puede considerarse como un conjunto de notas complementarias en alguna asignatura. Por supuesto, todo documento es susceptible de mejora, cualquier sugerencia o comunicación de error u omisión será bienvenida.

RSA

El sistema propuesto por Rivest, Shamir y Adleman en 1978 es uno de los sistemas de cifrado de clave pública más utilizados, tanto para cifrado como para firma de mensajes. En este documento nos ocuparemos de los detalles del sistema de cifrado. Este sistema basa su seguridad en el problema de la factorización de enteros, sin embargo, no se ha establecido la equivalencia entre ellos el problema de factorizar un entero y romper la seguridad de una clave RSA.

El Algoritmo 1 muestra el proceso de generación de una clave RSA. En esencia, primero el proceso escoge un entero n producto de dos números primos p y q. Segundo, teniendo en cuenta el valor n escogido, obtiene un número e y su inverso d módulo $\phi(n)$.

Algoritmo 1 Generación de clave RSA

Salida: Clave RSA $\langle K_{pb},\ K_{pr}\rangle$

1: Generar dos valores primos distintos p y q aproximadamente del mismo tamaño
2: $n = pq$
3: $\phi = (p-1)(q-1)$
4: Seleccionar un valor $1 < e < \phi$ al azar tal que $mcd(e, \phi) = 1$
5: Obtener el valor $1 < d < \phi$ tal que $ed \equiv 1 \pmod{\phi}$
6: $K_{pb} = (n, e)$
7: $K_{pr} = (d)$ //alternativamente (p, q)
8: **Devolver** $\langle K_{pb},\ K_{pr}\rangle$

Una vez que están disponibles las componentes pública y privada de la clave, el cifrado y descifrado son procesos básicamente idénticos y se muestran en los Algoritmos 2 y 3.

Algoritmo 2 Cifrado RSA

Entrada: $K_{pb}^{B} = (n, e)$: Componente pública de la clave RSA del destinatario
Entrada: x: Mensaje a cifrar (valor entero módulo n)
Salida: y: **Cifrado** RSA del mensaje x para el destinatario B

1: $y = x^e \bmod n$
2: **Devolver** y

Algoritmo 3 Descifrado RSA

Entrada: $K_{pr}^{B} = (d)$: Componente privada de la clave RSA
Entrada: y: Criptograma dirigido al destinatario lícito (valor entero módulo n)
Salida: x: **Descifrado** del mensaje cifrado RSA
1: $x = y^d$ mód n
2: **Devolver** x

Con estos algoritmos, el proceso de cifrado de mensajes supone descomponer estos en una secuencia de números en $\mathbb{Z}_n$, aplicando en secuencia el algoritmo de cifrado (o descifrado) a los números que codifican el mensaje a cifrar.

Corrección

El objetivo para probar que el proceso de cifrado y descifrado es correcto se basa en probar que, dados un mensaje x y una clave RSA $\langle K_{pb} = (n, e), K_{pr} = (d)\rangle$, se cumple que:

$$(x^e)^d = x^{ed} \equiv x \pmod{n}$$

Para probar que en efecto es así y que el descifrado de un criptograma corresponde con el mensaje cifrado, recordamos primero que que el proceso de construcción de las claves asegura que e y d son inversos el uno del otro módulo $\phi(n)$, o más formalmente:

$$ed \equiv 1 \pmod{\phi(n)} \;\Rightarrow\; ed = 1 + k\phi(n),\; k \in \mathbb{Z} \tag{1}$$

Para probar la corrección distinguiremos dos casos en función de si el mensaje x es o no es múltiplo de uno de los factores de n. Tomando como habitualmente que $n = pq$, con p y q primos, consideramos que el mensaje no es múltiplo de uno de los factores, por ejemplo p, esto es consideramos que $mcd(m, p) = 1$. En este caso, aplicando el Teorema de Fermat tenemos que:

$$x^{p-1} \equiv 1 \pmod{p}$$

por lo que podemos obtener que:

$$\begin{gathered}(x^{p-1})^{k(q-1)} \equiv 1^{k(q-1)} \pmod{p} \\ x^{k(p-1)(q-1)} \equiv 1 \pmod{p}\end{gathered}$$

y en un segundo paso que:

$$x^{k(p-1)(q-1)}x \equiv x \pmod{p}$$
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{p}$$

lo que permite aplicar directamente la Ecuación 1 y obtener que:

$$x^{ed} \equiv x \pmod{p}$$

En segundo lugar es necesario tratar el caso en el mensaje x es múltiplo de p, esto es $mcd(x,p) = p$. En esta situación $p|x$ y $x \bmod p = 0$, por lo que:

$$x^{ed} \equiv x \equiv 0 \pmod{p}$$

Razonando de forma idéntica se obtiene el mismo resultado para q el otro factor de n. Esto es:

$$\begin{cases} x^{ed} \equiv x \pmod{p} \\ x^{ed} \equiv x \pmod{q} \end{cases}$$

Debido a que p y q son primos distintos, por el Teorema Chino del Resto podemos concluir que:

$$m^{ed} \equiv m \pmod{pq}$$

y, por lo tanto, que el cifrado RSA es correcto.

Seguridad

La seguridad del cifrado RSA se basa en que el cifrado considera operaciones de exponenciación módulo n pero considerando valores e y d, uno inverso del otro módulo $\phi(n)$. En esta situación, conocidos n y e, en general es necesario obtener los factores p y q de n para poder calcular la clave privada d utilizando el algoritmo extendido de Euclides. No se ha demostrado que el conocimiento de e ofrezca ventaja alguna. A continuación se describen algunas debilidades del sistema y posibles soluciones que evitan estos ataques.

Factorización

No se conoce un algoritmo eficiente de propósito general para la factorización de un entero, que en el caso que nos ocupa es de la forma $n = pq$ con p y q primos. Sin embargo existen distintos algoritmos que pueden ser muy eficientes dependiendo de las

características de los factores p y q. La seguridad de una clave RSA recae en que el valor modular no posea características que permitan su factorización eficiente.

Sin intención de ser exhaustivo en la enumeración, entre los algorimos de factorización a tener en cuenta se encuentran:

- Factorización de Fermat
- Método de Pollard-rho
- Método de Pollard $p-1$
- Método de Williams $p+1$
- Factorización basada en curvas elípticas
- Método de Dixon
- Criba cuadrática
- Criba en campos de números
- Cálculo del orden (base del algoritmo cuántico de Shor)

El *RSA Factoring Challenge* promovido por *RSA Labs.* (https://www.rsa.com) enumera una serie de números de talla creciente, de modo que el mayor número de la lista factorizado puede considerarse como bechmark que proporciona una idea del tamaño de clave necesario para disponer de seguridad computacional. A fecha de hoy[1], el mayor número factorizado es RSA-768, por lo que una clave RSA de 1024 bits se considera segura.

Debilidades inducidas por e

La elección del exponente a utilizar en el cifrado RSA en ocasiones no es aleatoria y se basa en criterios de eficiencia. Uno de los exponentes utilizados habitualmente es $e = 3$.

El uso generalizado de este exponente permite un ataque en caso que el mismo mensaje se transmita a más de un destinatario (broadcasting). Puede asumirse que la probabilidad de que n_1, n_2 y n_3 (valor modular de las claves públicas de distintos usuarios) tengan un factor en común es despreciable, por lo que pordría construirse un sistema de ecuaciones y estaríamos en disposición de aplicar el Teorema Chino del Resto:

$$\begin{cases} x \equiv y_1 \pmod{n_1} \\ x \equiv y_2 \pmod{n_2} \\ x \equiv y_3 \pmod{n_3} \end{cases}$$

[1] Mayo 2018

En este caso particular el cifrado del mensaje $m^3 < n_1 n_2 n_3$, por lo que la solución al sistema sería $x = m^3$, siendo posible obtener el mensaje calculando la raíz cúbica de x.

Una modificación que evita esta debilidad es utilizar un *salt* aleatorio que se añade al mensaje (concatenando una pequeña secuencia binaria al mensaje), modificando los distintos mensajes que se envían a los distintos usuarios y evitando la aproximación

Debilidades inducidas por d

Por los mismos motivos de eficiencia descritos anteriormente, en ocasiones es interesante disponer de valores de d pequeños. Es importante tener en cuenta que existe un algoritmo que permite el cálculo de la clave privada d en el caso particular de que $mcd(p-1, q-1)$ sea pequeño (como habitualmente sucede) y d sea un cuarto de la talla de n.

Esta debilidad se evita haciendo que la talla de d sea aproximadamente igual a la de n.

Debilidades inducidas por n

Es posible romper la seguridad del cifrado RSA cuando distintas entidades de un colectivo comparten el mismo valor modular n. En este caso, disponer de una clave RSA del colectivo (disponer de exponentes e_i y d_i) permite la factorización de n y la obtención de las claves privadas del resto de miembros del colectivo.

Ataque basado en criptograma escogido

Aplicando propiedades de la exponenciación modular puede verse que, para cualquier par x_1, x_2 se cumple que:

$$(x_1 x_2)^e \equiv x_1^e x_2^e \equiv y_1 y_2 \pmod{n}$$

Utilizando este hecho (conocido como cifrado *homomórfico* o propiedad homomórfica del cifrado) es posible atacar el sistema mediante criptograma escogido.

Consideremos que el atacante puede utilizar acceder al descifrado de mensajes del usuario A y que quiere descifrar determinado criptograma $y = x^e$ mód n dirigido a A. El atacante puede construir un mensaje a partir de un z aleatorio en $\mathbb{Z}_n^*$ como:

$$y' = yz^e \text{ mód } n.$$

El proceso de descifrado de y' conduce a que:

$$\begin{aligned}(y')^d \bmod n &= (yz^e)^d \bmod n = \\ &= (yz^e)^d \bmod n = \\ &= y^d z^{ed} \bmod n = \\ &= xz \bmod n\end{aligned}$$

Con lo que es posible obtener el mensaje x si z es invertible (si z no lo fuera, el algoritmo extendido de Euclides daría como resultado uno de los factores de n rompiendo también la clave).

Ataques cíclicos

Dados una clave RSA $\langle (n, e),\ (d)\rangle$ y un criptograma concreto $y = x^e \bmod n$, existe un valor k entero tal que:

$$y^{e^k} \equiv y \pmod{n},$$

y debido al proceso de cifrado:

$$y^{e^{k-1}} \equiv x \pmod{n}.$$

Con esto puede abordarse un ataque a un cifrado RSA obteniendo el menor entero t tal que:

$$mcd(y^{e^t} - y, n) > 1.$$

Si $1 < mcd(y^{e^t} - y, n) < n$, entonces se ha obtenido uno de los factores de n y puede obtenerse la componente privada de la clave. Si $mcd(y^{e^t} - y, n) = n$, entonces el ataque cíclico ha tenido éxito y puede obtenerse el criptograma eficientemente calculando:

$$y^{e^{k-1}} \bmod n.$$

Pese a ser posibles, se asume que los ataques cíclicos tienen alta complejidad temporal y no suponen una amenaza.