Las matemáticas y el DNI electrónico

Ventura Sarasa Laborda

Trabajo de fin de grado en Matemáticas
Universidad de Zaragoza

Directores del trabajo: Paz Jiménez Seral y Manuel Vázquez Lapuente

27 de junio de 2021

Abstract

The goal of this project is to understand all the mathematical aspects that take part in our National Identity Document, which since 2006 incorporates an electronic Chip. This Chip turns our National Identity Document into an electronic document. Therefore, we are going to start talking about the most basic mathematical properties that we can find in the physical file of the DNIe and we will end up talking about the most complex factors that take part on the digital signature.

First, we are going to carry out a detailed explanation about what the letter of the DNIe and the numbers on its back represents. Many people have speculated about the meaning of this numbers, so we are going to explain what they are and what they are for. In addition to that, we will explain how you can get the numbers by yourself which is very interesting due to the fact that all of us have an DNIe but few are those who really know what the tax identification number or the control codes that appear on the back are for.

Once we have seen the most basic mathematical aspects, in Chapter 2 we are going to describe the Public-Key Criptography. Specifically we will focus on explaining in-depth the functioning of the RSA Cryptosystem since it is used in the signature protocols with the DNIe. So I believe is necessary to understand what it is and how it works. In section 2.2, we can clearly distinguish three parts that describe the RSA:

- RSA key generation.
- Encrypting a message with RSA.
- Decryption of the message with RSA.

By structuring the section this way, we have tried to give a practical approach so that the reader can understand how this Cryptosystem works and have the chance to reproduce the steps in case they want to maintain an encrypted correspondence. Section 2.3 outlines the problems that the RSA shows, problems that every mathematician would be concerned about. For example, if the RSA works, how to obtain large prime numbers and problems that arise with the exponentation.

Once the operation of the RSA Cryptosystem has been explained, Chapter 3 describes how to implement the RSA in a smart card such as the DNIe. To do so, it is necessary to follow the PKCS standards, which are property of RSA Laboratories. These standards are followed because the basic reference guide of the electronic DNI promoted by the Technical Commision to Support the Implementation of DNIe says so. The generation of RSA keys in the DNIe follows the PKCS#1 v.1.5 standard, so there will be two primes p and q involved in the creation of keys. This chapter explains the RSA-CRT (Chinese Remainder Therorem) protocol, which allows operations to be performed faster than the original RSA decryption, using the Chinese Remainder Theorem. The chapter presents protocols similar to the previous chapter and a solution to the exponentiation problem is given, since the CRT makes the RSA decryption protocol calculations more efficient.

Chapter 4 refers to prime numbers. We will study the solution to the challenge of finding very large prime numbers. For this, a study of Miller-Rabin Test is carried out, which explains what this test is

based on and the way in which it is implemented in reality to find numbers that will be prime with a certain probability, these possible primes are required for the creation of the RSA keys. Section 4.3 performs a probabilistic study of this test, in which its main function is to show the reader how many times it is necessary to perform the MIller-Rabin test to say that a number is prime with a high probability.

To finish this project, Chapter 5 refers to the main function of the DNIe, which is the digital signature. First of all, Law 59/2003, of December 19 defines what the electronic signature is, and classifies its different types. After that, a large part of the chapter will be dedicated to defining and explaining what a hash function is, which is a unidirectional function. Then we will present the most used nowadays family of hash functions and we will describe how the SHA-1 hash function works, which is involved in the process of signing through the DNIe. We will also examine how a public electronic signature protocol works, again explaining it in such way that the reader can imitate the steps to follow and electronically sign a document by himself. And finally, the signature protocol will be exposed by the DNIe, where concepts such as the electronic certificate will be exposed and it will be described by the mathematical notation how a recognized electronic signature is, such as that of the DNIe.

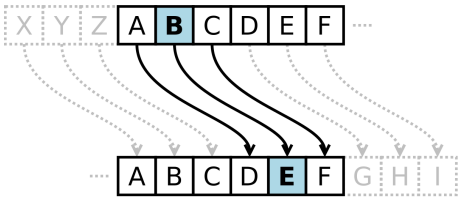
Keywords: DNIe, RSA, public-key, private-key, Chinese Remainder Therorem, Miller-Rabin, prime number, electronic signature, hash function.

Prefacio

La Criptografía está presente en todos los procesos que permiten preservar la integridad de una información, incluidas la confidencialidad, la autentificación y las firmas electrónicas. El objetivo de la Criptografía es conseguir una comunicación segura a través de canales inseguros, es decir, permite que dos personas, por ejemplo Sebastián y Ventura puedan enviarse mensajes por medio de un canal en el que una tercera persona Rafael, pueda captar dichos mensajes pero nunca entender su significado.

Se tiene noción del uso de Criptografía desde tiempos remotos, incluso ha sido históricamente fundamental en las guerras que han sacudido al planeta a lo largo de los siglos.

Un uso muy característico en la antigüedad es el cifrado por *sustitución*, que consiste en realizar una correspondencia entre las letras de un alfabeto. El ejemplo más representativo de este cifrado por *sustitución* es el *cifrado de César* que fue utilizado en el siglo I a.C. y que consistía en sustituir cada letra del alfabeto latino por la tercera letra que le sigue, véase la siguiente imagen.



Ninguno de los cifrados antiguos proliferó debido a que todos ellos podían ser rotos mediante técnicas estadísticas. Estos criptosistemas antiguos fueron sustituyéndose por criptosistemas en los que se conoce su algoritmo utilizado para cifrar pero no así los parámetros que se requieren o las claves que intervienen.

Una de las etapas más relevantes en la historia de la Criptografía es la Segunda Guerra Mundial. Los criptosistemas utilizados en esta época se conocen como criptosistemas de clave secreta o simétrica. Estos criptosistemas se caracterizan por utilizar una clave que solamente es conocida por la persona que envía el mensaje y por la que lo recibe. Por medio de esta clave, el emisor cifra el mensaje y envía el criptograma al receptor, el cual descifrará el mensaje con la misma clave. Estos criptosistemas también dejaron de usarse debido a tres grandes problemas que son: la distribución de la clave, el elevado número de claves que se necesitaban y la imposibilidad de asegurar la procedencia del mensaje.

La solución de estos problemas llega con el nacimiento de la Criptografía de clave pública, la cual puede decirse que nace en 1976 cuando Whitfield Diffie (5 de junio de 1944) y Martin Hellman (2 de octubre de 1945) crean un protocolo por el que dos personas pueden intercambiarse pequeñas informaciones secretas por un canal inseguro. El funcionamiento de la Criptografía de clave pública se explicará con detalle en 2.1. A modo esquemático, se resume con el siguiente flujo:

En 1978 se introduce el Criptosistema RSA por Ronald Rivest, Adi Shamir y Leonard Adleman. Se trata del Criptosistema de clave pública más utilizado hoy en día debido a su elevada seguridad, la cual se basa en la dificultad computacional que supone resolver el problema matemático de descomponer números enteros que son producto de primos muy grandes. Este Criptosistema es esencial en la elaboración de este Trabajo Fin de Grado, puesto que es el algoritmo que se usa para firmar electrónicamente como veremos en el Capítulo 5.

Uno de los puntos más importantes de este trabajo es el de encontrar números primos muy grandes para que el Criptosistema RSA no pueda ser vulnerado. Para ello se utiliza el test de Miller-Rabin, cuya versión original fue propuesta por Gary Lee Miller, dicha versión se basaba en la no demostrada hipótesis generalizada de Riemann. Unos años más tarde Michael Oser Rabin modificó la primera versión de Miller para convertirla en un algoritmo probabilístico que utiliza resultados probados.

El interés de realizar este trabajo se debe a que en las últimas décadas la humanidad ha sido testigo del continuo avance de la tecnología y de la informatización de multitud de procesos administrativos. Esta informatización ha requerido de una inversión y mejora en los procesos criptográficos esenciales en el ámbito de la ciberseguridad.

Un ejemplo es la digitalización de los bancos, donde ya resulta primitivo pensar en ir al banco a consultar tu saldo en la cuenta corriente o realizar un pago importante en efectivo. Las entidades que antes trabajaban de cara al público necesitan ahora una forma de asegurarse de que están tratando o realizando operaciones con una persona que es quien dice ser.

Por ello, hoy más que nunca, en esta sociedad digitalizada, los ciudadanos necesitan una forma eficiente de identificarse en Internet, de modo que nadie pueda suplantarles la identidad. Y es que cada vez son más y más las actividades a las que se puede asistir desde la silla de la habitación, para bien o para mal, hemos visto como personas pasan de asistir a conciertos en directo a pagar una entrada y asistir en *streaming* desde su casa o a la retransmisión de un partido de fútbol en directo donde su cara sale en un monitor gigante. Lo que antes eran agentes de seguridad personados, ahora son páginas web que te solicitan identificación.

La Unión Europea predijo a dónde nos estaba llevando la tecnología y ya en 1999 se apresuró a publicar una directiva que regulase un documento de identidad electrónico para su uso en los países miembros, en el caso de España estaríamos hablando del DNIe. Nuestro legislador, en reiteradas ocasiones, ha promulgado leyes que regulan este documento y aseguran su validez en todos procesos de identificación y firma de documentos. Sin embargo, en nuestro país, nunca se ha promocionado excesivamente su uso ni se ha obligado a las distintas entidades que trabajan en la red a que lo exijan.

Con esto pueden ser varias las preguntas que nos podemos plantear: ¿Son conscientes los ciudadanos españoles de lo que tienen realmente en sus carteras?, ¿es realmente necesario tener un Documento de Identidad electrónico? En este trabajo, que se divide en cinco capítulos, vamos a explicar de una forma extendida qué es, cómo funciona y lo seguro y valioso que resulta para que nadie nos suplante la identidad.

Índice general

Capítulo 1

Matemáticas elementales en el DNI

La tarjeta del DNI y algunos de los números que en ella aparecen tienen interesantes, aunque elementales, propiedades matemáticas.

1.1. El número áureo

El número áureo representa una particular proporción entre dos distancias, nombrado de diversas formas a lo largo de la historia, como número de oro, número de Dios o razón extrema media por los griegos, quienes dotaron de gran importancia a este número. Un ejemplo está en el diseño de la construcción del Partenón de Atenas, donde se puede comprobar que su cara frontal se puede descomponer en rectángulos áureos (véase en [4] pág. 13). Nuestro interés radica en que la tarjeta del DNI está asociada al número áureo.

Definición 1.1. *Se dice que dos números a y b están en proporción áurea si se cumple que:*

$$\frac{a+b}{a} = \frac{a}{b} \tag{1.1}$$

1.1.1. Cálculo del número áureo

De (1.1) tenemos

$$1 + \frac{b}{a} = \frac{a}{b}$$

Si denotamos $\phi = \frac{a}{b}$, nos queda la ecuación

$$1 + \phi^{-1} = \phi \Rightarrow \quad \phi + 1 = \phi^2 \Rightarrow \quad \phi^2 - \phi - 1 = 0$$

Así pues, resolviendo la ecuación de segundo grado, llegamos a que

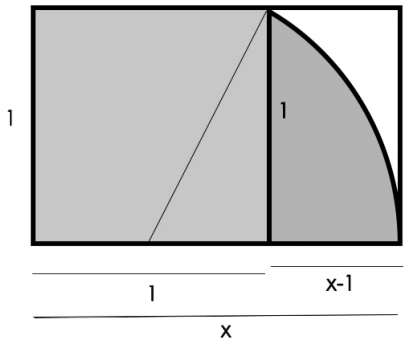
$$\phi = \frac{1+\sqrt{5}}{2} = 1{,}618...$$

1.1.2. Rectángulo áureo

Definición 1.2. *Un rectángulo áureo es aquel cuyos lados están en proporción 1.1.*

Propiedad 1. Un rectángulo es un rectángulo áureo si la proporción entre el lado mayor y el menor del rectángulo es igual a la proporción que queda al quitar el mayor cuadrado posible.

Ejemplo 1. Sea el caso de un rectángulo cuyos lados mayores son de longitud x y su lados menores tienen longitud 1. Por la definición de rectángulo áureo tomamos el mayor cuadrado posible y vemos que el rectángulo que queda es de dimensión $(x-1)\cdot 1$.



Veamos ahora qué ocurre, si consideramos el rectángulo inicial como un rectángulo áureo; es decir, que las proporciones del rectángulo mayor sean iguales a las proporciones del menor:

$$\frac{x}{1} = \frac{1}{x-1} \Leftrightarrow x^2 - x - 1 = 0 \Leftrightarrow x = \frac{1+\sqrt{1+4}}{2} = \frac{1+\sqrt{5}}{2}$$

En efecto, la proporción entre el lado mayor y el lado menor del rectángulo inicial es el número áureo.

1.1.3. El rectángulo áureo y la tarjeta del DNI

Cuando se diseñaron las diferentes tarjetas inteligentes, se pensó en producirlas con unas dimensiones ideales o perfectas, por ello estos diseños se inspiran en los rectángulos áureos. La proporción áurea es un número, del cual se dice, que todo elemento que lo posee mantiene una armonía natural y es símbolo de perfección.

Las tarjetas de crédito y el DNI siguen el estándar **ISO 7810**, el cual define los tamaños y hasta el radio de sus esquinas redondeadas, así como su grosor, el cual debe ser de 0,76 mm. El tamaño que corresponde a las tarjetas de crédito comunes y en particular el DNI es el **ID-1**. El formato ID-1 especifica un tamaño de 85,6×53,98 mm.

Si se toma el lado mayor a y menor b, se obtiene que la proporción que guardan sus lados es $\frac{a}{b} \approx 1,585$. Se aproxima al número áureo pero no alcanza su valor, presenta una desviación del 1.99 % respecto a dicho número.

A modo de ejemplo, la siguiente imagen demuestra que la tarjeta del DNI aunque no llegue con exactitud a preservar la proporción áurea, está inspirada en un rectángulo áureo.

1.2. Número de identificación fiscal

En 1987 se dispuso, mediante Ley, que toda persona física o jurídica, así como las Entidades sin personalidad a que se refiere el artículo 33 de la Ley 230/1963, General Tributaria, tendrá un número de

identificación fiscal para sus relaciones de naturaleza o con trascendencia tributaria.

Más tarde, en el Real Decreto 338/1990 se regula la composición y la correcta utilización del NIF. Se indica que consistirá del número del documento nacional de identidad seguido de un código o carácter de verificación, constituido por una letra mayúscula que habrá de constar en el propio documento nacional de identidad, de acuerdo con sus disposiciones reguladoras.

El 14 de marzo de 1990, el Ministerio de Economía y Hacienda aprueba el modelo de tarjeta donde se encontrará el NIF.

Llegados a este punto nos encontramos con nuestro NIF, el cual consta de 8 dígitos aleatorios, aunque bien es cierto que los NIF expedidos en fechas o años próximos comparten algunas características como los dos primeros dígitos. En este proceso no entran las matemáticas, no así en la selección de la letra de nuestro NIF.

Definición 1.3. *La letra de nuestro NIF corresponde al carácter de verificación correspondiente al número de Identificación Fiscal utilizado en España (con documento nacional de identidad (DNI) o número de identificación de extranjero (NIE) asignados por el Ministerio del Interior).*

Es en 1990 cuando se implantó la letra de control necesaria para formar el Número de Identificación Fiscal. Se crea una correspondencia entre números y letras tal que:

T	R	W	A	G	M	Y	F	P	D	X	B	N	J	Z	S	Q	V	H	L	C	K	E
0	1	2	3	4	5	6	7	8	9	10	11	12	13	4	15	16	17	18	19	20	21	22

Nuestro alfabeto consta de 27 letras, pero se puede observar que la lista con la que se decide la letra del NIF consta de 23 caracteres. Es claro ver que el 23 es el mayor número primo menor que 27. Aunque el Ministerio del Interior afirma que el número 23 resulta de suprimir cuatro letras que pueden inducir a confusión: la letra O con 0, la I con 1, la U con la V y la Ñ con la N.

En la web del Ministerio del Interior se explica al ciudadano el procedimiento por el cuál se obtienen las letras de los NIF, no haciendo mención a que el número 23 es un número primo con importantes cualidades en este proceso, y quedando señalado de la siguiente forma: “Se divide el número entre 23 y el resto se sustituye por una letra que se determina por inspección mediante la tabla anterior”. Matemáticamente, se traduce en escoger la letra que corresponde a la clase de restos módulo 23 del DNI.

Proposición 1. *La letra del DNI permite detectar un error.*

Demostración. Sean $(x_0,\dots,x_7)$ y $(y_0,\dots,y_7)$ dos números DNI. Supongamos que únicamente se diferencian en la cifra $x_j \neq y_j$, mientras que $x_i = y_i$ para todo $0 \leqslant i \leqslant 7$, $i \neq j$. Para ver que en efecto la letra detecta errores, nos planteamos el caso en el que los dos DNI tienen la misma letra, lo que nos lleva a concluir que ambos números divididos entre 23 tienen el mismo resto. Los dos números de DNI los representamos en forma decimal tal que:

$$x = \sum_{i=0}^{7} x_i 10^i, \qquad y = \sum_{i=0}^{7} y_i 10^i.$$

Se tiene que 23 es divisor de $x-y$ ya que los restos de dividir 23 entre x e y son los mismos. Pero tenemos que:

$$x - y = (x_j - y_j)10^j.$$

Así pues, se tiene que 23 divide a $(x_j - y_j)10^j$. Observad que el $\mathrm{mcd}(23,10) = 1$, por lo tanto, concluimos que 23 divide a $x_j - y_j$. Pero x_j y y_j son números en $\mathbb{N}$ comprendidos en $[0,9]$, es claro que $x_j - y_j \in [-9,9]$, luego el único múltiplo de 23 es 0. Por lo tanto $x_j = y_j$.

De esta forma se concluye que la letra del DNI permite detectar un error. □

Ejemplo 2. Veamos si cometiendo dos errores en el DNI la letra puede detectar el error.

Tomamos de nuevo el DNI $x = 73108480J$ y esta vez consideramos el DNI $y = 73108457J$. Difieren en el séptimo y octavo dígito, cambiando el 8 por el 5 y el 0 por el 7. Procediendo como en el ejemplo anterior tenemos que:

$$x - y = 73108480 - 73108457 = 23 \tag{1.2}$$

es claro que el resto de dividir entre 23 es 0. Por lo que 73108457J es un DNI válido y la letra no detecta el error.

Un error típico es intercambiar de forma no intencionada dos cifras consecutivas del DNI, veamos que la letra sí que detecta este error.

Teorema 1.1. *La letra del DNI nos permite detectar el intercambio de dos dígitos consecutivos.*

Demostración. Tomamos x e y en forma decimal al igual que en la proposición anterior. Nos situamos en el siguiente caso:

$$x_j = y_{j+1}, \quad x_{j+1} = y_j, \quad \text{para } 0 \leqslant j \leqslant 7.$$

Y con

$$x_i = y_i, \quad \text{para } 0 \leqslant i \leqslant 7 \quad \text{con i} \notin \{\text{j, j+1}\}.$$

Al suponer que x e y son números DNI correctos y que tienen la misma letra, se tiene que 23 divide a $x - y$ y a su vez es claro que:

$$x - y = (x_{j+1} - y_{j+1})10^{j+1} + (x_j - y_j)10^j = (x_{j+1} - x_j)10^{j+1} + (x_{j+1} - x_{j+1})10^j$$

$$= (x_{j+1} - x_j)(10-1)10^j = (x_{j+1} - x_j)9 \cdot 10^j.$$

Al observar que el $\text{mcd}(23, 9) = 1$ y $\text{mcd}(23, 10) = 1$, se deduce que 23 tendrá que dividir a $x_{j+1} - x_j$, y como antes, la única posibilidad es que $x_{j+1} - x_j = 0$, por lo tanto, se tendría que $x_{j+1} = x_j$.

Luego, en este caso, la letra del DNI nos ha permitido detectar el intercambio de dos dígitos seguidos y por tanto encontrar un error en los DNI.

□

1.3. Códigos de control en el DNI

El reverso del DNI ha sido siempre diana de múltiples leyendas en torno al significado de los números que figuran en la parte inferior. A continuación se presenta el significado de estos códigos y los métodos matemáticos que se utilizan para su obtención.

Estos caracteres son caracteres OCR-B (*Optical character recognition*, reconocimiento óptico de caracteres, tipo B) para la lectura mecanizada sobre la identidad del ciudadano (véase una descripción más detallada en [15]) .

Para tener una mejor visión de lo que son y significan estos números, hay que mencionar a la OACI (Organización de Aviación Civil Internacional). La séptima edición del Doc 9303 [7], representa una reestructuración de las especificaciones de la OACI para documentos de viaje de lectura mecánica. En este documente se regula y explica cómo se calculan los códigos de verificación y cuál es su utilidad.

Volviendo a la combinación de caracteres que se presenta en la parte trasera del DNI, esta combinación está preparada para ser leída por máquinas. Meramente es una cifra de control para verificar que la lectura de todos los datos del DNI ha sido correcta. La zona de datos OCR del DNI electrónico se divide en 15 campos distintos, de los cuales, cuatro de ellos tienen un dígito llamado dígito de control como podemos ver en 1.3, cuya función es asegurar que los datos que figuran en los campos sean correctos.

1.3.1. Obtención y verificación de los dígitos de control

Se ha adoptado un método a la hora de calcular los dígitos de control o verificación a efectos de utilizarlos en los MRTD (*machine-redeable travel document*). Los dígitos se calculan sobre módulo 10 con la sucesión continua de factores de ponderación 731 731 731... Más formalmente, sea un campo del reverso del DNI, $C=\{a_1, a_2, \ldots, a_{r-1}\}$, y sea a_r el código de control. Se deberá cumplir que:

$$a_1 \cdot 7 + a_2 \cdot 3 + a_3 \cdot 1 + \ldots \equiv a_r \ (mod\ 10). \tag{1.3}$$

Los números de serie del DNI tienen tres letras al comienzo de estos, y se sustituyen por números siguiendo las indicaciones: $A \Rightarrow 0$, $B \Rightarrow 1$, $C \Rightarrow 2\ldots$

Ejemplo 3. Tomamos el número de serie de la imagen anterior: BHF133836, según la correspondencia de la tabla tenemos que la letra B pasa a ser 1, la H un 7 y la F un 5.

Así pues, hacemos la siguiente operación:

$$(7\cdot 1)+(3\cdot 7)+(1\cdot 5)+(7\cdot 1)+(3\cdot 3)+(1\cdot 3)+(7\cdot 8)+(3\cdot 3)+(1\cdot 6) = 123 \equiv 3\ mod(10).$$

Luego, tomamos el número 3 que será nuestro primer dígito de control asociado al campo 3 (el número de serie del soporte físico de la tarjeta).

El cuarto dígito de control se consigue de la misma forma con la concatenación de los campos 3, 4, 5, 7, 8, 10 y 11. Recordar que se deben sustituir las letras siguiendo las indicaciones anteriores.

Ejemplo 4. Veamos que el dígito de control nos detecta un error. La fecha de vencimiento correcta de la imagen es 23/07/03 y su correspondiente código de control es el 5. Tomemos ahora la fecha 23/07/04 y procedemos a multiplicar por la secuencia 7, 3, 1. Se tiene que:

$$(7*2)+(3*3)+(1*0)+(7*7)+(3*0)+(1*4) = 76 \equiv 6\ (mod\ 10) \tag{1.4}$$

Por lo tanto al no coincidir 5 con 6, se detectaría un error en el DNI.

Capítulo 2

El Criptosistema RSA

El DNI electrónico tiene la utilidad de reconocer de manera fidedigna al usuario en las comunicaciones electrónicas. Es decir, es capaz de generar firmas electrónicas. Esta utilidad se fundamenta en criptografía, y más concretamente en la criptografía de clave pública. Uno de los sistemas más extendidos, y que es el que utiliza el DNIe, es el sistema RSA.

2.1. Criptografía de clave pública

La criptografía de clave pública (o clave asimétrica) surge como solución a los problemas de la criptografía de clave simétrica. Estos problemas son la distribución de la clave, el número de claves (serán tantas como personas con las que se mantiene una correspondencia) y la imposibilidad de asegurar la identidad de la persona con la que uno se comunica. Puede decirse que la criptografía de clave pública nace en 1976 cuando Diffie y Hellman [6] describen un protocolo por el que dos personas pueden intercambiarse pequeños mensajes secretos por un canal inseguro.

Un usuario de un criptosistema de clave pública posee dos claves, una clave pública E que sirve para cifrar mensajes tal que para un x obtenemos $E(x) = y$, y una clave privada D que sirve para descifrar mensajes, de forma que $D(y) = D(E(x)) = x$.

Este criptosistema basa su seguridad en la dificultad que presenta, conociendo la clave pública E, encontrar tal D que cumpla lo anterior.

Si un usuario S (Sebastián) desea mandar un mensaje cifrado a otro usuario V (Ventura), debe seguir las siguientes pautas:

1. S toma la clave pública de V, E, ya sea de un repositorio de claves públicas o porque V se la brinda.

2. S aplica E al mensaje m que quiere enviar, tal que $E(m) = m'$ y envía m' a V.

3. V recibe m', toma su clave privada D y se la aplica al mensaje recibido m'. Por tanto al ser D la inversa de E, V obtiene $D(m') = m$, que es el mensaje original.

Este caso de comunicación entre dos partes nos permite comprobar que los problemas de la criptografía de clave simétrica se solucionan. Transmitir la clave ya no es un problema, ya que cada usuario tiene una clave privada que únicamente él conoce y una clave pública conocedora por todos. No es necesario tener una clave diferente para cada persona con la que un usuario S vaya a entablar una comunicación, pues si alguien quiere enviar un mensaje a V, este solamente necesita su clave pública. Por último, podemos asegurarnos de que un usuario S es quien dice ser en verdad mediante un protocolo de firma electrónica de carácter público, que veremos en el capítulo de Firma Electrónica.

Hoy en día los criptosistemas de clave pública más extendidos son el criptosistema RSA y el criptosistema de Curvas Elípticas. Dado que el DNIe utiliza el criptosistema RSA para firmar electrónicamente, es necesario dedicar un capítulo al RSA para comprender correctamente qué es y cómo funciona.

2.2. Presentación del RSA

El RSA es un criptosistema de clave pública introducido por Rivest, Shamir y Adleman en 1978 [19]. Es el criptosistema de clave pública más utilizado. Conviene aclarar que en el criptosistema RSA, la clave pública E se asocia con un número entero n y un exponente e, mientras que la clave privada D se asocia con un exponente d. El protocolo RSA consta de tres partes: generar las claves, cifrar los mensajes y descifrarlos.

2.2.1. Generación de claves con RSA

Supongamos que dos usuarios deciden mantener una correspondencia cifrada mediante el criptosistema RSA, sean V, Ventura y S, Sebastián. V debe elegir su clave pública y privada de la siguiente forma:

1. El usuario V toma dos números primos p y q. Calcula el producto $n = p \cdot q$ y también $m = (p-1)\cdot(q-1)$. El cifrado y descifrado RSA, se realiza en el anillo $\mathbb{Z}_n$, donde los cálculos modulares juegan un papel fundamental.

2. V selecciona un entero e tal que $2 < e < m$, de forma que:

$$mcd(e,m) = 1.$$

3. Aplicando el algoritmo de Euclides extendido, V calcula el inverso de e módulo m, que será el entero d, es decir:

$$e \cdot d \equiv 1 (mod\ m)$$

y sabemos que este inverso existe porque $mcd(e,m) = 1$.

4. Por último, la clave pública del usuario V será el par (n,e) y su clave privada será el número d.

Una vez el usuario V ha creado sus claves, será de dominio público el par (n,e), mientras que debe permanecer en secreto la clave privada d y los números p,q y m.

2.2.2. Cifrado de mensajes con RSA

Para que S envíe un mensaje cifrado a V, S deberá realizar las siguientes acciones:

1. S busca y obtiene la clave pública de V, (n,e) de algún directorio de claves o mediante comunicación directa con V.

2. S toma un mensaje y lo transforma en un elemento M de $\mathbb{Z}_n$, si M es muy extenso tendrá que dividirlo para que cada división sea un elemento de $\mathbb{Z}_n$, es decir, en enteros dentro del rango $\{0,1,2,...,n-1\}$.

3. S cifra M con la clave pública de V:

$$M^e = C\ (mod\ \ n) \tag{2.1}$$

y envía el criptograma C, a V.

2.2.3. Descifrado del mensaje

Una vez el usuario V recibe el criptograma, procede del siguiente modo:

1. V recibe el criptograma $C = M^e$, y utiliza su clave privada d para descifrarlo:

$$(C)^d = (M^e)^d (mod\ \ n) \equiv M\ (mod\ \ n). \tag{2.2}$$

Una vez que V obtiene M, saca el mensaje en claro.

La última congruencia se obtiene en la siguiente sección del capítulo, donde se abordarán los problemas que presenta el criptosistema RSA.

Ejemplo 5. Supongamos que S (Sebastián) desea enviar un mensaje a V (Ventura). Con fines didácticos se presenta el siguiente ejemplo para ver el funcionamiento del RSA, y en el que se toman unos p y q pequeños por razones de espacio y complejidad en la explicación (un criptosistema RSA formado por estos números no supondría ninguna dificultad para que un intruso descompusiera n y obtuviera los valores de p y q).

1. V elige en primer lugar, dos números primos, calcula su producto n y también m: $p = 401$, $q = 547$, $n = p \cdot q = 219347$ y $m = (p-1)(q-1) = 218400$.

2. V procede a seleccionar una clave de cifrado $e < 218400$, dicho e debe cumplir que $mcd(e, 218400)=1$. Así V decide tomar $e = 73$.

3. Una vez que V obtiene la clave e, procede mediante el algoritmo de Euclides extendido, y obtiene: $1 = 73 \cdot (95737) - 32 \cdot 218400$. Luego el inverso de e será la clave de descifrado $d = 95737$.

4. Finalmente V da a conocer a S su clave pública, $(n, e) = (219347, 73)$ y mantiene en secreto su clave privada, $d = 95737$, como los demás parámetros p, q y m.

Ventura y Sebastian usarán el alfabeto estándar tal que $A = 0, B = 1, C = 2 \ldots$ S quiere enviar a V el mensaje cifrado: *MEXVOYXAXMADRID*.

S toma la clave pública de V, (219347, 73). Ahora S debe escribir el mensaje como un número menor que $n = 219347$ y que sea primo con él. Dado que la longitud del mensaje no puede exceder el valor del módulo RSA, se analiza si tal hecho ocurre, en cuyo caso el mensaje se descompone en tantos bloques como sea necesario. En este caso se tiene que la longitud de n es igual a 6 y el mensaje pasado a número es $M = 120423211424230023120003170803$; por lo tanto, este número se trocea en bloques de 5 dígitos. Tenemos de esta forma 6 bloques tal que: $M_1 = 12042$, $M_2 = 32114$, $M_3 = 24230$, $M_4 = 02312$, $M_5 = 00031$, $M_6 = 70803$.

Para cifrar el mensaje, S debe cifrar cada bloque, simplemente calculando los valores $M_i^e\ (mod\ \ n) = C_i$. Dichos valores serán: $C_1 = 166745$, $C_2 = 213162$, $C_3 = 110990$, $C_4 = 12547$, $C_5 = 149267$, $C_6 = 53468$. Estos C_i son enviados como criptogramas al usuario V.

V recibe los 6 criptogramas C_i y a partir de su clave privada d calcula $C_i^d\ (mod\ \ n)$ para $i = 1, ..., 5$, obteniendo los valores M_i. A continuación V concatena todos los M_i y recupera el texto original pasando el número a letra, obteniendo así el mensaje: *MEXVOYXAXMADRID*.

2.3. Problemas a resolver en RSA

Dado que el criptosistema RSA es el criptosistema de clave pública más utilizado hoy en día, es también el que más estudios ha generado con el fin de determinar sus debilidades. Por ahora no se ha conseguido romper el cripstosistema y su fortaleza sigue intacta. De todas formas, la descripción del RSA plantea una serie de cuestiones o problemas que es preciso resolver.

2.3.1. ¿Funciona el RSA?

La esencia del criptosistema RSA radica en la congruencia 2.2 vista en la sección anterior, si dicha congruencia falla o no se cumple, el criptosistema RSA no funcionará. Este problema se resuelve con el siguiente teorema:

Teorema 2.1. *Sean dos primos p y q tal que $n = p \cdot q$, $m = (p-1)\cdot(q-1)$ y un $e \in \mathbb{Z}$ tal que m.c.d(e, m) = 1. Sea d tal que $e \cdot d \equiv 1$ (mod m). Entonces para cada $x \in \mathbb{Z}$ se tiene que $x^{ed} \equiv x$ (mod n).*

Demostración. Se tiene que $\mathbb{Z}_n$ es un anillo y el grupo multiplicativo de las unidades tiene $(p-1)(q-1)$ elementos, por lo que sus elementos tienen orden divisor de $(p-1)(q-1)$. Si $x \in \mathbb{Z}$ es tal que m.c.d (x, n) = 1, se tiene que es una unidad de $\mathbb{Z}_n$ y por tanto $x^m \equiv 1$ (mod n). Se tiene que:

$$x^{ed} \equiv x^{1+km} = x \cdot x^{km} \equiv x \ (mod \ \ n).$$

Suponemos ahora que x es múltiplo de p pero no de q, por lo que mcd (x, q) = 1, entonces x sería una unidad de $\mathbb{Z}_q$ y como las unidades de $\mathbb{Z}_q$ tienen $q-1$ elementos se tendría que $x^m = (x^{(q-1)})^{(p-1)} \equiv 1 \ (mod \ \ q)$ y por tanto

$$x^{ed} \equiv x^{1+km} \equiv xx^{km} \equiv x \ (mod \ \ q).$$

Por otra parte, como x es múltiplo de p, se tiene que $x^{ed} \equiv x \ (mod \ p)$ porque $x \equiv 0 \ (mod \ p)$ y por ser p y q primos distintos se tiene que $x^{ed} \equiv x$ (mod n). Lo mismo ocurriría si x es múltiplo de q pero no de p. Y si x es múltiplo de p y de q será múltiplo de n y por tanto $x^{ed} \equiv x$ (mod n).

□

Visto este Teorema hemos comprobado que el protocolo 2.2.3 funciona.

2.3.2. Necesidad de utilizar primos grandes

Cualquier usuario de RSA quiere estar seguro de que el criptosistema no pueda ser alterado o roto por alguien con intenciones maliciosas. RSA como criptosistema de clave pública se cumple siempre y cuando del conocimiento de n y e no se pueda deducir d.

Lo que se busca es elegir los primos p y q de modo que factorizar $n = p \cdot q$ sea computacionalmente muy difícil, puesto que si un intruso en el criptosistema conociera los valores de p y q mediante la factorización de n, también conocería el valor de m y a partir de ese valor y del de e podría calcular sin problema la clave privada d. Para que esto no ocurra p y q deben ser primos grandes, actualmente se recomienda que tengan una longitud mínima de 512 bits, pero ¿cómo saber si un número de una longitud considerablemente grande es primo? Para ello se recurre en el caso del DNIe al Test de Miller-Rabin (véase en el Capítulo 4), que nos dice si un número es primo con una cierta probabilidad.

2.3.3. El problema de la exponenciación

Como hemos visto antes, p y q deben ser números grandes. Por tanto los valores de n, e y d serán también grandes y el problema de la exponenciación será complicado y costoso.

En el caso de que e y d sean grandes, se tendrá un problema a la hora de cifrar y descifrar un mensaje M, ya que serán procesos computacionalmente costosos. Para solucionar este problema existe un procedimiento para calcular de forma más eficiente M^e y C^d.

Se trata de encontrar algún algoritmo que acorte el número de operaciones para calcular una potencia de exponente a. Notar que multiplicar por 2 en binario es lo mismo que añadir un 0. En general para $a_i \in \{0,1\}$, se tiene que:

$$((a_r a_{r-1} \cdots a_0)_2) \cdot 2 = (a_r \cdot 2^r + a_{r-1} \cdot 2^{r-1} + ... + a_0) \cdot 2 =$$

$$= a_r \cdot 2^{r+1} + a_{r-1} \cdot 2^r + ... + a_0 \cdot 2 = (a_r a_{r-1} \cdots a_0 0)_2.$$

Para calcular x^a, debemos tomar en primer lugar a en binario, tal que $a = (a_r a_{r-1} \cdots a_1 a_0)_2$, con $a_r = 1$. Se parte de $x = x^{(1)_2}$ y se va mirando los dígitos a_i de izquierda a derecha, cuando $a_i = 0$, lo que tenemos hasta el momento se eleva al cuadrado y si $a_i = 1$, se eleva al cuadrado y se multiplica por x. Así hasta llegar al término a_i que está más a la derecha.

Ejemplo 6. Sea $a = 87$, expresamos a en binario tal que $87 = (1010111)_2$, siguiendo los pasos del algoritmo anterior, partiendo desde $x = x^{(1)_2}$, tendremos que hacer las siguientes operaciones:

$$x \to x^2 \to x^5 \to x^{10} \to x^{21} \to x^{43} \to x^{87}.$$

Otra forma de solventar el problema de la exponenciación es seleccionar un exponente de cifrado e pequeño, dado que quien debe cifrar un mensaje M, ha de llevar a cabo la operación $M^e \pmod n$, y de esta forma se facilita la tarea de cifrado, para que sea eficiente. Por contra partida, si el exponente e es pequeño querrá decir que d será grande y la operación de descifrado, que será calcular C^d se complica. Como veremos en el siguiente capítulo este problema se solventa mediante el protocolo de descifrado RSA-CRT, que utiliza el Teorema Chino de los Restos.

En la práctica, los exponentes de cifrado más utilizados son $e = 3$, $e = 17$ y $e = 2^{16} + 1$. Esto se debe en parte a su tamaño y a que estos números en binario están formados por pocos 1's, algo que como hemos visto antes es una gran ventaja a la hora de cifrar un mensaje.

Finalmente, existe una tercera manera de solventar el problema de la exponenciación con números grandes. Consiste en utilizar mensajes cortos, bien porque se está utilizando un criptosistema de clave simétrica y el RSA únicamente se utiliza para cifrar la clave simétrica o bien porque solo se cifra un resumen del mensaje, como ocurre en procesos de firma electrónica. Este último caso se estudiara de manera detallada en el Capítulo 5.

Capítulo 3

DNIe y RSA

Para entender el uso del RSA en el DNI electrónico es necesario tener conocimiento sobre los PKCS (*Public-Key Cryptography Standards*). Los PKCS son un conjunto de estándares hechos para la criptografía de clave pública, son mantenidos y propiedad de RSA Laboratories. Estos estándares como así figura en la guía de referencia básica del DNI electrónico impulsada por la Comisión Técnica de Apoyo a la Implantación del DNIe (veáse en [2] pág. 26) son necesarios para una correcta implementación del RSA en el DNIe. En este capítulo utilizaremos la notación de PKCS.

3.1. RSA-CRT y PCKS#1

El PKCS#1 [20] es el primero de la extensa familia de estándares PKCS que proporciona las definiciones y recomendaciones básicas para aplicar el algoritmo RSA en criptosistemas de clave pública. Define las propiedades matemáticas de las claves públicas y privadas, operaciones primitivas para el cifrado y esquemas criptográficos seguros.

En la primera versión del PKCS#1, el par de claves tradicional se basa en el módulo n, que es el producto de dos primos p y q, tal que $n = p \cdot q$. En las ocasiones en las que el RSA se emplea en tarjetas inteligentes, como es el DNIe, se implementa un protocolo de descifrado diferente al protocolo explicado en el capítulo anterior. Este protocolo es conocido como el RSA-CRT (CRT, *Chinese Remainder Theorem*), que permite realizar las operaciones con mayor rapidez en comparación con el descifrado RSA original, y para ello se utiliza el Teorema Chino de los Restos.

La generación de claves RSA en el DNIe sigue el estándar PKCS#1 v.1.5 (véase en [21]). Por lo tanto el número de primos involucrados en la formación del módulo RSA será dos.

3.1.1. Generar las claves RSA-CRT y cifrar un mensaje

Sean p, q, n, m y e como en el capítulo anterior. La diferencia con el protocolo de generación de claves RSA original es que ahora no se calcula el exponente privado d. Sean ahora dP, dQ y $qInv$ definidos mediante:

- $e \cdot dP \equiv 1 \ (mod \ (p-1))$
- $e \cdot dQ \equiv 1 \ (mod \ (q-1))$
- $q \cdot qInv \equiv 1 \ (mod \ p)$

donde dP, dQ y $qInv$ se pueden calcular mediante el Teorema de Euclides extendido.

Con esto un usuario V dispone de su clave pública (n, e) y de su clave privada formada por la tupla $(p, q, dP, dQ, qInv)$.

Para cifrar un mensaje se procede del mismo modo que en el protocolo de cifrado RSA explicado en la sección 2.2.2. El usuario S toma un mensaje y lo convierte en un número M, el cual será un entero tal que $M \in \{1,...,n-1\}$. S eleva M al exponente e tal que:

$$M^e = C \ (mod\ n).$$

S obtiene así el criptograma C y procede a enviárselo a V.

3.1.2. Descifrado con RSA-CRT

El usuario V recibe el criptograma C y toma su clave privada formada por la tupla (p, q, dP, dQ, $qInv$). Primero calcula $C_q \equiv C \ (mod\ q)$ y $C_p \equiv C \ (mod\ p)$, siendo C_q y C_p números menores que q y p respectivamente. Una vez que V tiene C_q y C_p, calcula $C_q^{dQ} = M_1$ y $C_p^{dP} = M_2$. Finalmente aplica la fórmula de Garner (véase pág. 8 de [21]) y obtiene M':

$$M' = M_1 + [(M_2 - M_1) \cdot qInv] \cdot q \ (mod\ n). \tag{3.1}$$

En la siguiente sección se demuestra que $M' = M$, y así es claro que V ha obtenido el mensaje en claro.

3.2. Justificación del descifrado con CRT

Como ya se dijo en el capítulo anterior, en el protocolo de cifrado se suele utilizar un exponente de cifrado e pequeño, y sabemos que cuanto más pequeño sea e más grande será el exponente de descifrado d.

Por esa razón el proceso de descifrado que obliga a calcular C^d módulo n, donde C es un elemento de $\mathbb{Z}_n$, es un proceso lento y costoso. Para acelerar este proceso hay que tener en cuenta que los cálculos módulo p y q son mucho más rápidos que los cálculos módulo n y además se sustituirá la base C y el exponente de descifrado d por números más pequeños. Es aquí donde juega un papel importante el Teorema Chino de los Restos.

Notar que esta ventaja únicamente la puede aprovechar el dueño de la clave privada, este procedimiento es entre 4 y 8 veces mas rápido, véase en [16].

3.2.1. Enunciado del CRT y su uso en el RSA

En esta sección vamos a demostrar que el M' de la fórmula 3.1 es el mensaje pasado a número M.

Teorema 3.1. *Sean p y q primos distintos y $n = p \cdot q$. Sean $m_1, m_2 \in \mathbb{Z}$. El sistema:*

$$x \equiv m_1 \ (mod\ p)$$
$$x \equiv m_2 \ (mod\ q)$$

tiene solución única módulo n. Dicha solución es de la forma:

$$x = m_1 \cdot q \cdot qInv + m_2 \cdot p \cdot pInv$$

Demostración. Es una simple comprobación. □

La siguiente observación es necesaria para comprender la implementación del CRT en el RSA.

Observación 1. Con la notación anterior, se tiene

$$1 \equiv q \cdot qInv + p \cdot pInv \ (mod\ n).$$

Para verlo basta considerar el siguiente sistema de congruencias, cuya solución es trivial:

$$x \equiv 1 \ (mod\ p)$$
$$x \equiv 1 \ (mod\ q).$$

A continuación, se muestra el procedimiento que utiliza el DNIe para el proceso de descifrado en el que no se usa la clave privada tradicional y en el que se aplica el CRT.

Aplicación 1. Este procedimiento toma la clave privada nombrada en la anterior sección compuesta por la 5-tupla (p, q, dP, dQ, $qInv$) y aplica la fórmula 3.1 para obtener M'. A continuación, se demuestra que $M' = M$ donde M es el mensaje resultante de descifrar el criptograma C.

Demostración. Sea M el mensaje original tal que $M \in \mathbb{Z}_n$. Sea el criptograma C tal que $C = M^e \ (mod\ n)$. Se trata de probar que $M = M'$. En primer lugar veamos que

$$M \equiv M^{e \cdot dQ} \ (mod\ q).$$

Puesto que $edQ \equiv 1 \pmod{q-1}$, se sigue que $e \cdot dQ = 1 + k(q-1)$, para algún k. De aquí se sigue que $M^{edQ} = M \cdot M^{k(q-1)}$. Si M y q son primos entre sí, por el pequeño Teorema de Fermat (el cual se verá de forma detallada en el siguiente capítulo) se tiene que $M^{k(q-1)} \equiv 1 \pmod{q}$. También si q divide a M, entonces $M \equiv 0 \pmod{q}$. En cualquiera de los dos casos es $M \equiv M^{edQ} \pmod{q}$. Análogamente se tiene que $M \equiv M^{edP} \pmod{p}$. Así pues M es solución al sistema de congruencias:

$$x \equiv M^{e \cdot dQ} \ \ (mod\ \ q)$$
$$x \equiv M^{e \cdot dP} \ \ (mod\ \ p).$$

Por otra parte, por la observación 2, se sigue que $1 \equiv q \cdot qInv + p \cdot pInv \ \ (mod\ \ n)$. Ahora tomando el M' resultado de la fórmula de Garner (3.1) se tiene que:

$$M' = M_1 + (M_2 - M_1) \cdot q \cdot qInv = M_1 \cdot (1 - q \cdot qInv) + M_2 \cdot q \cdot qInv =$$
$$= M_1 \cdot p \cdot pInv + M_2 \cdot q \cdot qInv$$

y por el Teorema Chino del Resto M' es también solución del sistema de congruencias anterior, luego ambas soluciones son congruentes módulo n, y como las dos son menores que n, deben ser iguales.

□

Ejemplo 7. Un usuario V procede a crearse una clave pública y privada para tener una correspondencia privada con un compañero de trabajo. En primer lugar el usuario V genera dos primos aleatorios como son $p = 73$ y $q = 107$ y calcula el módulo RSA, $n = 7811$ y $m = 7632$.

El siguiente paso para V será seleccionar un exponente de cifrado público e (a poder ser de longitud pequeña), V selecciona $e = 5$. Mediante el algoritmo de Euclides extendido, V calcula el inverso de e módulo $p-1$ y el inverso de e módulo $q-1$. Es decir obtiene $dP = 29$ y $dQ = 85$.

Así V tiene una clave pública (n, e) a disposición de quien quiera y una clave privada en forma de 5-tupla, la cual es (p, q, dP, dQ, $qInv$).

El usuario S manda un mensaje cifrado a V, para ello utiliza la clave pública de V. Este recibe el criptograma $C = 232$ y decide aplicar el protocolo de descifrado RSA-CRT. Para ello, procede a utilizar su clave privada. Primero calcula:

$$C = 232 \equiv 13 = C_p \ \ (mod\ p)$$
$$C = 232 \equiv 18 = C_q \ \ (mod\ q)$$

prosigue con el proceso de descifrado calculando:

$$\begin{aligned} M_1 &\equiv C_q^{dQ} = 18^{85} \equiv 21 \pmod{q} \\ M_2 &\equiv C_p^{dP} = 13^{29} \equiv 11 \pmod{p} \\ qInv &= q^{-1} \equiv 58 \pmod{p}. \end{aligned}$$

Y por último aplica la fórmula de Garner tal que:

$$M = 21 + [(11-21)\cdot 58]\cdot 107 = 21 + 4\cdot 107 = 449.$$

Luego V obtiene $M = 449$ (efectivamente $449^5 \equiv 232 \pmod{n}$), de donde podrá obtener el mensaje en claro.

Capítulo 4

Test de Miller-Rabin

4.1. En busca de números primos

Para generar un par de claves RSA seguras, es necesario encontrar dos números primos grandes. Actualmente se recomienda que estos números tengan unos 310 dígitos, para que el producto de estos factores sea de unos 620 dígitos, que representa unos 2048 bits. Uno de los problemas que surgen para implementar el criptosistema RSA es el de generar estos primos tan grandes. Este problema coincide con uno de los principales problemas de la Teoría de Números, que es determinar si un número n, es primo o compuesto.

Un número es primo si no tiene divisores no triviales. Naturalmente para ver si un número es primo bastará averiguar si tiene algún divisor primo. El método para averiguar si un número tiene o no factores primos es el siguiente:

Test de las divisiones sucesivas. Este test considera un entero n impar y decide si el número es o no primo. La idea es tomar un número impar t, menor que n y ver si este divide o no a n. Si t divide a n y no es ni 1 ni n, entonces el número candidato a primo n será compuesto. En otro caso, n pasa el intento de división por t. Los valores de t abarcan desde el 3, que es el primer número primo impar, hasta el número más cercano a $\sqrt{n}$, es decir, t debe ser primo y cumplir $3 \leqslant t < \sqrt{n}$. Notar que no hace falta probar valores $t > \sqrt{n}$, porque si n es compuesto, al menos tendrá dos factores y primos, y el menor factor primo de n no puede ser mayor que $\sqrt{n}$.

Este procedimiento puede parecer sencillo, pero resulta hartamente costoso para números grandes como los que se utilizan en la generación de claves RSA. Debido a esto, se han creado *métodos* que permiten detectar si un número es primo o no con cierta probabilidad, sin necesidad de estudiar su factorización.

Un *método* consiste en comprobar si un número candidato a ser primo cumple una serie de propiedades que cumplen todos los números primos. En caso de que no cumpla una de esas condiciones el número será compuesto, si cumple esas condiciones, probablemente el número será primo. Sin embargo estos métodos requieren un alto coste computacional si se quiere obtener un número que sea primo con una probabilidad muy alta. Por esta razón, en la práctica se suelen utilizar comprobaciones menos costosas, pero que por contrapartida no dan una respuesta correcta con absoluta seguridad. Se distingue así entre test de primalidad y test de pseudoprimalidad.

Llamaremos *test de primalidad* a un algoritmo determinista que decide si el número candidato a primo realmente lo es o no, basándose en el cumplimiento o incumplimiento de ciertas propiedades por parte del número candidato. En cambio los test de pseudoprimalidad no aseguran la primalidad del candidato.

Un *test de pseudoprimalidad* es un algoritmo que decide si un número candidato a primo lo es con una cierta probabilidad, basándose en unas propiedades que el número candidato debe cumplir.

Para ver si un número es primo o no, en este capítulo nos vamos a centrar en el test de pseudoprimalidad de Miller-Rabin pues es el que se utiliza en el DNIe. En [8] podemos encontrar un estudio bastante extenso sobre los test de primalidad y pseudoprimalidad aplicados al criptosistema RSA y de donde se ha recabado información para la realización de este capítulo. El Test de Miller-Rabin se fundamenta en el pequeño Teorema de Fermat, que es el siguiente.

Teorema 4.1 (**Pequeño Teorema de Fermat**). *Sea $p \in \mathbb{N}$ un número primo, entonces se tiene que $\forall a \in \mathbb{Z}_p$ y $a \neq 0$*

$$a^{p-1} \equiv 1 \pmod{p}. \tag{4.1}$$

Demostración. Por ser p primo, las unidades de $\mathbb{Z}_p$ es grupo multiplicativo de orden $p-1$. Como a es primo con p, a es una unidad, luego se sigue la tesis. □

Este teorema proporciona uno de los test de pseudoprimalidad más famosos, el cual se base en comprobar la fórmula 4.1 para el mayor número de $a \in \mathbb{Z}_p$, no obstante aunque se cumpla la fórmula no se puede asegurar que el número p en cuestión sea primo, en cambio si encontramos un a para el que no se cumpla dicha expresión, entonces se puede asegurar que el número p es compuesto.

Pueden encontrarse números que no son primos y que cumplan el pequeño Teorema de Fermat y por tanto torpedear algunos test de primalidad. Estos números se conocen como números de Charmichael, en concreto, todos los números de Charmichael son producto de tres o más primos distintos entre ellos. El más pequeño de estos números es $p = 3 \cdot 11 \cdot 17 = 561$. La cantidad de números de Charmichael es infinita, pero únicamenete se encuentran 7 números de este tipo entre los 10000 primeros números enteros y menos de 600000 que estén entre 10^{17}. Por lo tanto, la probabilidad de encontrarse con un número al azar con estas características y que este por debajo de 10^{17} es de aproximadamente 10^{-11}. Por el contrario, la probabilidad de encontrar un número primo por debajo de 10^{17} es de $\frac{2}{ln(10^{17})} = \frac{1}{17 \cdot ln(10)} \simeq 0{,}0256$.

4.2. Test de Miller-Rabin

El test que el DNIe utiliza para asegurar con una alta probabilidad que el número escogido es un número primo es el Test de Miller Rabin, viene recogido en la guía de referencia básica del DNIe, promulgada por la Comisión Técnica de Apoyo a la Implantación del DNIe [2]. Dicho test se basa en el siguiente lema.

Lema 1. *Sea p un número primo, y sea x un elemento de $\mathbb{Z}_p$ tal que:*

$$x^2 \equiv 1 \pmod{p}$$

entonces, $x \equiv 1$ o bien $x \equiv p-1$.

Demostración. Del enunciado se sigue $x^2 - 1 \equiv 0 \pmod{p}$, luego $(x+1)(x-1) \equiv 0 \pmod{p}$. Por lo tanto se llega a la conclusión de que como $\mathbb{Z}_p$ es dominio de integridad por ser p primo, $x \equiv 1$ o $x \equiv -1 \equiv p-1$.

□

Teniendo en cuenta los conceptos anteriores, Miller ([12]) y Rabin ([17]) desarrollaron el test que lleva su nombre y que enunciamos como sigue:

Teorema 4.2 (**Test de Primalidad de Miller-Rabin**). *Sea p un número primo, escribimos $p-1=2^u \cdot r$, con r un número impar y $u>0$, entonces $\forall a \in \mathbb{Z}_p$ se tiene*

$$a^r \equiv 1 \pmod{p}$$

o bien existe un $s \in \{0,...,u-1\}$ tal que

$$a^{2^s \cdot r} \equiv -1.$$

Demostración. Supongamos que $a^r \not\equiv 1$ módulo p. Probaremos que entonces se verifica la segunda congruencia. Por el pequeño Teorema de Fermat tenemos que

$$a^{p-1} \equiv a^{2^u \cdot r} \equiv 1 \pmod{p}$$

luego $(a^{2^{u-1} \cdot r})^2 \equiv 1 \pmod{p}$ y por el lema anterior se tiene que

$$a^{2^{u-1} \cdot r} \equiv 1 \pmod{p} \text{ o bien } a^{2^{u-1} \cdot r} \equiv -1 \pmod{p}.$$

En el caso de que $a^{2^{u-1} \cdot r} \equiv -1 \pmod{p}$, obtenemos el resultado. En caso contrario tendríamos que $(a^{2^{u-2} \cdot r})^2 \equiv 1 \pmod{p}$ y de nuevo por el lema anterior se tendría

$$a^{2^{u-2} \cdot r} \equiv 1 \pmod{p} \text{ o bien } a^{2^{u-2} \cdot r} \equiv -1 \pmod{p}.$$

Iterando sucesivamente el razonamiento anterior se concluye que alguno de los términos $a^{2^s \cdot r}$ es congruente con -1 módulo p con $s \in \{0,...,u-1\}$ o bien todos los términos son congruentes con 1 módulo p y en particular $a^r \equiv 1 \pmod{p}$, en ambos casos obtenemos el resultado.

□

Este teorema describe una propiedad de los números primos, la usamos como test de pseudoprimalidad, luego si un candidato a primo no verifica esta propiedad para un cierto a, entonces ese número no es primo. Si la verifica se le podrá aplicar la misma propiedad con otro a, cuantos más a utilicemos más probable será que el número candidato sea primo. En la práctica se utiliza el siguiente algoritmo para generar números que sean primos con una gran probabilidad a partir de dicho test.

Algoritmo 1. (De búsqueda aleatoria de un número primo mediante el Test de Miller-Rabin.) Este algoritmo tiene como entrada los números $k, t \in \mathbb{N}$.

1. En primer lugar, a partir de k se genera un número impar aleatorio p de k bits.
2. Mediante una lista de los 300 primeros números primos, se procede a hacer divisiones simples al número p, si obtenemos que p es compuesto, se cambia p por $p+2$ y se vuelve al paso 2.
3. Después se procede a calcular los números $u, r \in \mathbb{Z}$, tal que $p-1=2^u r$, con r impar.
4. Este algoritmo tiene como entrada la tupla (p,t), donde el número t es un parámetro de seguridad. Se va a determinar si el número p es primo con una probabilidad de acierto determinada. El algoritmo tiene la siguiente forma:

```
for i in [1..t] do:
  Se toma aleatoriamente una base a tal que a ∈ [2, p−2]
  c = a^r (mod p)
  if (c ≠ 1 and c≠ n-1) then do:
    s = 1
    while s ⩽ u−1 & c ≠ p−1 do:
      c = c^2 (mod p)
      if c = 1 → n será compuesto
      s = s+1
    if c ≠ p−1 → p será compuesto
return → p es probablemente primo
```

Nota 1. Según [9] el algoritmo de Miller-Rabin tiene un tiempo de ejecución de:

$$O((lg\, n)^3)$$

Aunque es posible reducir el tiempo de ejecución a $O((lg\, n)^2)$, siempre y cuando se tomen valores pequeños de la base a, eso significaría que dichos valores ya no serían aleatorios.

4.3. Estudio probabilístico del Test de Miller-Rabin

Teorema 4.3. *Sea $n > 1$ un entero compuesto e impar. La proporción de bases a en [2,n-2] que no supera el test de Miller-Rabin es mayor del 75 %. Equivalentemente la proporción de bases $a \in [2, n-2]$ que supera el test de Miller-Rabin para el número n es como mucho del 25 %.*

Demostración. Difícil y extensa, puede verse en [3]. □

Es decir, la proporción de los a que superan el test respecto a los valores que puede tomar y siendo n un número compuesto es inferior a $\frac{1}{4}$, entonces si hacemos t pasadas al test de Miller-Rabin la probabilidad será de $(\frac{1}{4})^t$. Por lo que el test nos da un resultado, siendo este más fiable cuanto mayor sea t, aunque siempre habrá una mínima posibilidad de que el algoritmo falle.

Ejemplo 8. Tomemos un número n impar y compuesto, sea $n = 85$, la base a puede tomar valores en el intervalo [2, 83]. Aplicando el test de Miller-Rabin con una pasada, se comprueba que las bases 13, 38, 47 y 72 no superan el test, según el Teorema 4.3 menos del 25 % de las bases en el intervalo [2, 83] no superarán el test, en este caso en concreto nos encontramos con que 4 bases de 83 posibles no pasan el test, lo que implica que un 4,81 % de las bases posibles para el número 85 no pasan el test de Rabin-Miller (comprobado mediante Jupyter).

Definición 4.1. *La probabilidad de que el Algoritmo 1 se equivoque, es decir, que de un número compuesto como primo se denota por p_{kt}.*

Observación 2. Unos mayores refinamientos a la probabilidad p_{kt} permiten definirle unas cotas superiores teniendo en cuenta el tamaño del número generado por los k bits y las t pasadas que se realizan al test. Estos refinamientos son realizados en la situación de que en el Algoritmo 1 no se han realizado las divisiones simples por el listado de los primeros 300 primos. Dichas cotas provienen de complicados cálculos, para mayor información ver [1].

A partir de las cotas superiores calculadas para p_{kt} se forman unas tablas en función de los valores k y t, la siguiente tabla enumera alguno de los límites superiores mejorados para p_{kt}:

	t									
k	1	2	3	4	5	6	7	8	9	10
100	5	14	20	25	29	33	36	39	41	44
150	8	20	28	34	39	43	47	51	54	57
200	11	25	34	41	47	52	57	61	65	69
250	14	29	39	47	54	60	65	70	75	79
300	19	33	44	53	60	67	73	78	83	88
350	28	38	48	58	66	73	80	86	91	97
400	37	46	55	63	72	80	87	93	99	105
450	46	54	62	70	78	85	93	100	106	112
500	56	63	70	78	85	92	99	106	113	119
550	65	72	79	86	93	100	107	113	119	126
600	75	82	88	95	102	108	115	121	127	133

Nota 2. Un elemento i de la tabla, correspondiente a un k y un t implica que $p_{kt} < (\frac{1}{2})^i$. Véase esta tabla y comentarios en relación a ella en [11].

Ejemplo 9. Si de entrada en el Algoritmo 1 introducimos el par $(450,8)$, es decir pedimos al programa que genere un número de 450 bits y de 8 pasadas al test de Miller-Rabin para comprobar si el número es o no primo. Si el algoritmo nos devuelve un número del que dice ser primo, la probabilidad de que se confunda y ese número n sea en realidad un número compuesto es menor que $(\frac{1}{2})^{100}$=$(\frac{1}{4})^{50}$. Es claro que la probabilidad $p_{k,t}$=$(\frac{1}{2})^{100}$ es mucho menor que la probabilidad teórica $(\frac{1}{2})^{2t}$.

Observación 3. En la práctica se suele tolerar una probabilidad de error de $(\frac{1}{2})^{80}$ en la generación de primos con el algoritmo anterior. A partir de la tabla anterior se pueden deducir los valores t para los que la probabilidad $p_{k,t} \leqslant (\frac{1}{2})^{80}$, los cuales se recogen en las siguientes tablas:

k	t
100	27
150	18
200	15
250	12
300	9
350	8
400	7
450	6

k	t
500	6
550	5
600	5
650	4
700	4
750	4
800	4
850	3

k	t
900	3
950	3
1000	3
1050	3
1100	3
1150	3
1200	3
1250	3

k	t
1300	2
1350	2
1400	2
1450	2
1500	2
1550	2
1600	2
1650	2

k	t
1700	2
1750	2
1800	2
1850	2
1900	2
1950	2
2000	2
2050	2

Es decir a partir de los bits elegidos para generar el número n, sabemos cuántas pasadas serán suficientes para aceptar que el número n es primo con una alta probabilidad. Por ejemplo, si tomamos $k = 1700$, el algoritmo genera un número n de 1700 bits y sabemos que con 2 pasadas será suficiente para afirmar que dicho número n es primo con una probabilidad muy alta.

Ejemplo 10. Nuestro DNIe contiene dos pares de claves distintas, un par se utiliza para la autenticación del ciudadano y el otro para firmar. Como el DNIe utiliza el algoritmo RSA para cifrar y descifrar, es claro que en la creación de las claves intervendrán 4 números primos, dos para cada tipo de claves. Cada uno de estos 4 primos es de alrededor de 500-512 bits. Cada ciudadano español posee 4 números primos y según los últimos datos brindados por la Policía Nacional, en España hay unos 44 millones de DNI expedidos. Lo que quiere decir que se han generado unos 176 millones de números primos. Supongamos que el **Algoritmo 1** ha dado 7 pasadas al test de Miller-Rabin, eso quiere decir que cada uno de los 176 millones de primos es primo con una probabilidad de $1-(\frac{1}{4})^7 \simeq 0{,}99994$, es equivalente a decir que la probabilidad de que el algoritmo falle es de $(\frac{1}{4})^7 \simeq 6{,}1035 \cdot 10^{-5}$, es decir, que en nuestro país podría haber 10742 números compuestos que se dan como primos y que podrían acarrear algún problema de ciberseguridad.

Capítulo 5

La Firma Electrónica

La firma electrónica es un procedimiento electrónico que indica si una persona es la responsable del contenido de un documento electrónico.

5.1. Legislación de Firma Electrónica

El uso de la firma electrónica en España se recoge en la Ley 59/2003, del 19 de diciembre [10], la cual le otorga la misma validez que a la firma manuscrita. En esta ley se distinguen cuatro tipos de firma:

- **La firma electrónica** es un conjunto de datos de forma electrónica asociados a otros y que se pueden utilizar para identificar al firmante. Un ejemplo de lo que puede ser esta firma es el remite de un e-mail o el pin de una tarjeta de crédito.
- **La firma electrónica avanzada** es la firma electrónica que está generada por medio de un dispositivo seguro de creación de firma. Las firmas basadas en clave pública cumplen esta condición. Este tipo de firma es el que se utiliza en el sistema criptográfico PGP (*Pretty Good Privacy*), que es el sistema criptográfico más utilizado en el mundo.
- **La firma electrónica avanzada y certificada** es la firma avanzada garantizada por una autoridad de certificación que cumple algunas de las funciones y obligaciones que la ley exige a los prestadores de servicios de certificación, en particular certifica. Diversas entidades como bancos o tiendas *online* recurren a estas empresas para realizar operaciones electrónicas de forma segura. Por ejemplo la firma pública de un banco como el BBVA está garantizada por la empresa DigiCert Inc, con sede en Estados Unidos.
- **La firma electrónica reconocida** es la firma electrónica avanzada y certificada cuyo prestador de servicios tiene respaldo gubernamental. La firma electrónica que se realiza con el DNIe es una firma electrónica reconocida y está garantizada por la Dirección General de la Policía (DGP).

El uso de firmas electrónicas permite resolver uno de los problemas más representativos de los criptosistemas de clave simétrica, como es que el destinatario de un mensaje no esté seguro de que quién se lo envía sea realmente el remitente.

5.2. Función resumen

Cuando vamos a firmar electrónicamente un documento, no se firma el texto completo sino que se utilizan las funciones resumen que ayudan a reducir considerablemente el coste computacional de los protocolos de elaboración y verificación de firma.

Una de las funciones más importantes en la criptografía son las funciones unidireccionales. Estas funciones son la base de las funciones resumen que se utilizan en los protocolos de creación y verificación de firma electrónica.

Definición 5.1. *Una función unidireccional es una función f definida entre dos conjuntos $f: X \longrightarrow Y$, con $f(x) = y$, de tal forma que es fácil calcular para cualquier $x \in X$ su imagen, mientras que para los elementos $y \in Y$ que son imagen de algún $x \in X$ es difícil, desde un punto de vista computacional, calcular tal $x \in X$ de modo que $f(x) = y$.*

Definición 5.2. *Una función resumen o función hash, r, es una función unidireccional que se aplica a un mensaje o texto y devuelve un resumen que siempre es del mismo tamaño.*

Es obvio deducir que al ser todos resúmenes de una longitud fija, existirán muchos más mensajes o textos que resúmenes, ya que siempre son de una longitud constante. Por lo tanto existirán mensajes diferentes cuyos resúmenes coinciden. Las funciones resumen tienen que cumplir las siguientes propiedades para que no resulten vulnerables:

1. Necesita depender de los bits, es decir, debe depender de cada uno de los bits del mensaje, por lo que si se cambiara un solo bit el resumen cambiaría.

2. Oposición a la preimagen, es decir, dado el resumen m, tiene que ser computacionalmente difícil obtener M de modo que $r(M) = m$.

3. Oposición a la segunda preimagen, dado un mensaje M y conociendo su resumen $r(M) = m$, debe ser computacionalmente complicado obtener otro mensaje H de forma que los resúmenes coincidan, es decir, que $r(M) = r(H)$.

4. Tiene que ser computacionalmente complicado encontrar una colisión, es decir, determinar dos mensajes cualquiera M, H cuyos resúmenes coincidan o colisionen.

Unas de las funciones resumen más utilizadas y seguras de la actualidad son el SHA-1 y la familia de funciones SHA-2.

5.3. Familia de funciones SHA

La familia de funciones SHA (*Security Hash Algoritm*) son una familia de funciones resumen publicadas por el NIST (*National Institute of Standards and Tecnhnology*). En 1995 se publica la función SHA-1 [13], que es la función resumen más utilizada actualmente para protocolos de firma electrónica, proporciona un resumen de 160 bits. En particular, el DNIe utiliza la función SHA-1 para el proceso de identificación del ciudadano y para procesos de firma de documentos o mensajes. El funcionamiento de la función SHA-1 es el siguiente:

1. Se toma un mensaje M de longitud variable menor que 2^{64} bits y se divide en n bloques de 512 bits cada uno. En el caso de que la longitud de M no sea múltiplo de 512, éste se extiende de modo que su n-ésimo bloque tenga esa longitud.

2. Se consideran cinco valores inciales A, B, C, D, E, son de 32 bits cada uno y al ser concatenados proporcionan un vector de inicialización L_1 que será por tanto de 160 bits.

3. En el SHA-1 se utiliza una función que se aplica 80 veces, dicha función depende de cuatro funciones lógicas f_t. Cada f_t opera con tres palabras de 32 bits, que son las palabras B, C y D y produce una palabra de 32 bits como salida. La función f_t se define de la siguiente manera para las palabras B, C, D:

   - $f_t(B,C,D) = (B \wedge C) \vee (\sim B \wedge D)$, para $(0 \leqslant t \leqslant 19)$

- $f_t(B,C,D) = B$ XOR C XOR D, para $(20 \leqslant t \leqslant 39)$
- $f_t(B,C,D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$, para $(40 \leqslant t \leqslant 59)$
- $f_t(B,C,D) = B$ XOR C XOR D, para $(60 \leqslant t \leqslant 79)$

4. Cada bloque M_i del mensaje se divide en 16 palabras de 32 bits. Con las 16 palabras del bloque M_1 y el vector de inicialización L_1 se realiza el siguiente algoritmo con 80 pasadas:

   Algoritmo 2. En primer lugar existen cuatro constantes K_t, después se toma el primer bloque del mensaje M de 512 bits, al que denotaremos como M_1 y se procede de la siguiente manera:

   - Se divide M_1 en 16 palabras $W_0, ..., W_{15}$.
   - Para $t = 16, ..., 79$, se tiene que $W_t = S^1(W_{t-3}$ XOR W_{t-8} XOR W_{t-14} XOR $W_{t-16})$.
   - Se redefinen los vectores A,B,C,D,E tal que: $H_0 = A$, $H_1 = B$, $H_2 = C$, $H_3 = D$, $H_4 = E$.
   - Para $t = 0$ hasta $t = 79$, se hace lo siguiente:

   $$TEMP = S^5(A) + f_t(B,C,D) + E + W_t + K_t$$

   $$E = D;\ \ D = C;\ \ C = S^{30}(B);\ \ B = A;\ \ A = TEMP$$

   De esta forma se tienen nuevos valores para A, B, C, D y E y concatenando los vectores formamos el nuevo vector L_2.

5. Tomamos el segundo bloque del mensaje M_2 y junto al vector L_2 operamos y se saca de nuevo otro vector, en este caso L_3. Este proceso se repite hasta completar todos los bloques del mensaje M.

6. Por último, la salida del algoritmo SHA-1 es el resumen de M, que son los últimos 160 bits obtenidos resultado de las operaciones que se hacen al último bloque M_n, con el último vector L_n de las operaciones anteriores, es decir el valor de L_{n+1} que viene definido como la concatenación de H_0, H_1, H_2, H_3, H_4 definidos esta vez como: $H_0 = H_0 + A$, $H_1 = H_1 + B$, $H_2 = H_2 + C$, $H_3 = H_3 + D$, $H_4 = H_4 + E$, con los vectores A,B,C,D,E las componentes del vector L_n. Finalmente, el resumen de M es $m = L_{n+1}$.

Para una mayor seguridad, en 2002 el NIST presentó una nueva serie de funciones SHA a la que bautizó como SHA-2 [14]. La familia SHA-2 consta de cuatro algoritmos: SHA-224, SHA-256, SHA-384 y SHA-512 que devuelven resúmenes de 224, 256, 384 y 512 bits respectivamente. De hecho aumentar la longitud de los resúmenes hace aumentar considerablemente la seguridad ante un posible ataque.

5.4. Firma electrónica de caracter público

El caso que nos interesa de firma electrónica es la que podemos realizar mediante nuestro DNIe. Este tipo de firma se considera pública, ya que es posible identificar al remitente con información de dominio público como puede ser su clave pública. Y se considera también irrevocable, ya que una vez firmado el texto o mensaje el firmante no puede negar su autenticidad. La firma electrónica basada en criptosistemas de clave pública cumple todos los requisitos de la firma avanzada.

Dadas dos personas S (Sebastián) y V (Ventura), y sean las claves pública y privada de S, [E_S, D_S]. En el supuesto de una situación de correspondencia en la que uno de los implicados quiere firmar electrónicamente un documento y el otro quiere poder demostrar la autoría del firmante, se utilizan dos protocolos: **de elaboración de firma** y **de verificación de firma**.

5.4.1. Protocolo de elaboración de firma

1. S es el autor del mensaje a firmar. Calcula su resumen (p. ej. mediante la función SHA-1) tal que $r(M) = m$, donde el resumen m si se ha obtenido mediante la función resumen SHA-1 será de 160 bits.
2. S procede a firmar el mensaje M aplicando su clave privada al resumen del mensaje $D_S(m) = f$.
3. Por último, S envía a V el par formado por el mensaje y la firma: $[M, f]$. En el caso de que así se decida puede enviar junto al mensaje firmado su clave pública, de tal forma que S enviaría: $[M, f, E_S]$.

5.4.2. Protocolo de verificación de firma

V es conocedor de la clave pública de S y ambos se han puesto de acuerdo previamente en el uso de una función resumen, r, determinada. Para que V verifique que el firmante del mensaje es en realidad S, sigue los pasos siguientes:

1. V obtiene $[M, f, E_S]$ y procede a aplicar la clave pública de S a f, para así recuperar el resumen m, se tiene pues que $E_S(f) = E_S(D_S(m)) = m$.
2. Por último, V comprueba que el resumen obtenido en el primer paso es el mismo que el que ha calculado aplicándole al mensaje M la función resumen r. Si es así, se asegura que el firmante es quien dice ser.

5.4.3. Seguridad ante posibles ataques

El problema en la seguridad de todo este proceso surge ante la posibilidad de que un tercero, llámase R (Rafael), intercepte la tupla $[M, f, E_S]$, enviada por S a V. R podría cambiar o modificar el mensaje (pasando de M a $\widetilde{M}$) y aplicarle al resumen su clave privada D_R, obteniendo así una nueva firma k, para finalmente enviar a V la tupla $[\widetilde{M}, k, E_R]$. V recibiría un mensaje firmado supuestamente por S y si V no desconfía de la identidad del firmante, el ataque habrá sido un éxito. Por ello el receptor de cualquier mensaje firmado tiene que estar seguro de que la clave pública que utiliza para verificar la autenticidad del mensaje es de verdad del firmante.

Para que V esté seguro de que la clave pública es realmente del firmante, existen entre otros los siguientes protocolos para verificar una firma electrónica avanzada:

- S entrega personalmente a V su clave pública, de este modo no habrá ninguna duda en comprobar si la firma electrónica es de S.
- Por teléfono, V reconoce la voz de S mientras le dicta los primeros dígitos de su clave pública.
- Un amigo de la confianza de V firma la clave pública de S, entonces V puede fiarse de que esa clave pública es en verdad de S.
- V ha obtenido de forma segura la clave pública de S y la introduce en su ordenador, pero desconfía de la posible intromisión de un tercero en su ordenador para cambiar la clave pública de S, luego firma con su clave privada la clave pública de S.
- Que S utilice un protocolo de firma electrónica avanza y certificada. Estaríamos ante un caso en el que una autoridad de certificación crea un certificado electrónico. Este certificado electrónico está formado por la clave pública del usuario y datos personales, su función es certificar que la clave pública es realmente del firmante. El certificado está firmado electrónicamente con la clave privada del prestador de servicios. Nos fiamos de él ya que la clave pública del prestador de servicios está instalada en nuestro ordenador. Este tipo de entidades paga a las empresas que fabrican sistemas operativos para que sus claves estén instaladas en los ordenadores.

Ejemplo 11. La universidad de Zaragoza tiene contratada como entidad de verificación a la empresa Geant Vereniging con sede en Holanda. El certificado electrónico que genera Geant Vereniging se compone de los datos de la Universidad de Zaragoza junto con su clave pública y la firma electrónica por parte de Geant Vereniging de todos estos datos. Dicha firma electrónica es de 512 bytes y puede verse parte de ella en la siguiente imagen:

Firma 512 bytes: 25 AA 4F 49 61 93 5D FC D2 24 FD C0 DD 02
67 F5 2A 80 C9 F6 C4 8C 26 3B 91 56 44 FE CA 53 9E
2B 9E F1 90 8E 23 C7 85 BF 3A 18 C2 0F E2 C1 DD F8
64 3C 76 D7 BB 57 CA 96 27 FF F3 76 ED 2B 91 3F 08
BB 06 4B 40 66 86 99 CA DB 28 25 AF 03 9D D8 7B 35
E1 30 79 CD 05 19 D1 61 77 09 F3 07 43 AC BB 15 5B 76
8C 9C 09 EC C6 7B E7 21 75 E0 D7 6B 84 0D 70 4B 75
7B DF A9 ED 10 1C 2D CA 51 EC E4 ED 30 3A C8 0D 3F
14 94 B5 B6 78 DD C4 02 28 23 53 4C 87 BF FC 05 BC
EA 93 85 89 A6 42 FA 3E F8 71 33 E5 A3 6D 2B EF 56
C5 B2 D5 E3 55 00 51 EF 1E 2C 71 4F CB 3B FF 80 54 20
9E AE 40 FD 40 55 26 8D B0 58 F6 7B AA 92 E7 24 0C

5.5. Firma electrónica con DNIe

Históricamente, la función principal de un documento de identidad ha sido la de identificar a su dueño. Hoy en día gracias a los avances tecnológicos, los documentos de identidad han evolucionado. Han mejorado considerablemente su seguridad física y el principal cambio ha sido la inserción de un *chip* que tiene la capacidad de hacer operaciones de carácter criptografico.

5.5.1. Características del DNIe

Las principales características del DNIe son:

- Cuando un DNIe se expide, se generan cuatro números primos, de donde dos de ellos sirven para crear una clave pública y privada para procesos de identificación y los otros dos para crear otra clave pública y privada para procesos de firma. Se genera a su vez sendos certificados, uno para cada una de las dos claves públicas, dichos certificados contienen los datos personales del ciudadano y la clave pública. Este certificado es firmado con la clave privada de la DGP y se almacena junto a las claves públicas y privada del ciudadano en el chip del DNIe.
- El DNIe utiliza RSA para firmar pero carece de funciones criptográficas, es decir, únicamente vale para identificarse y firmar.
- Utiliza el test de Miller-Rabin para encontrar los primos que se utilizan en la creación de claves RSA y utiliza el CRT para realizar cálculos de forma más eficiente. El exponente e es siempre $2^{16}+1=65537$.
- El tamaño de la clave pública del DNIe es de 2048 bits (el módulo n) y tienen un período de caducidad de dos años y medio.
- La firma del DNIe es una firma electrónica avanzada, certificada y reconocida, ya que hay una entidad respaldada por el Gobierno, en este caso la DGP, que verifica mediante un certificado electrónico que el firmante es el propietario del DNIe.

Puede verse en [18] un artículo extenso sobre las características del DNIe publicado en la Gaceta de la RSME (*Real Sociedad Matemática Española*).

Ejemplo 12. La clave pública para procesos de firma con DNIe como hemos dicho antes es de 2048 bits, en este caso la clave pública de Ventura Sarasa tiene el siguiente aspecto:

Clave pública RSA (2048 bits)

30 82 01 22 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01 05 00 03 82 01 0F 00
30 82 01 0A 02 82 01 01 00 9F 48 9A FC BE 89 20 29 62 56 35 EA 2C EB 85
23 DB CD 90 D4 2D CD 7C C4 E1 1F F7 81 84 4D 81 43 A9 35 BD F3 30 7E
69 DA DD 3B 7A 45 30 3C 80 09 76 45 EC EC F7 34 D8 96 07 2C 79 2B 8F
DC F8 96 44 16 20 F6 60 FE 38 6E C6 1C 45 2F 70 E7 90 4D 3E BB 59 B9
A9 71 4A 5A 22 02 47 00 F5 D7 D6 CA DF DE 55 D6 DF 7A 06 5A 82 65 2E
31 45 26 43 6C CC 00 78 EC 2C 95 10 35 87 4F A2 79 C9 F6 93 A1 6D 85
F2 B4 8D 2F DE 58 75 AA 19 70 86 EC 65 F7 C8 A5 CC 29 85 F1 22 B4 0E
7D D7 E5 42 9D 23 DA 62 C3 E2 BF A6 0B 80 84 FF C7 5E 2D 99 DC 9C 17

5.5.2. Protocolo de firma con DNIe

Si el usuario S tiene vigentes las claves de su DNIe puede firmar cualquier mensaje o documento mediante el criptosistema RSA, para ello suponemos que la clave pública de S, es (n,e) y su clave privada d. En primer lugar S escribe y confecciona el mensaje a enviar M. En el proceso de firma se realizan los siguientes pasos:

1. S procede a firma un documento con su DNIe, el protocolo de firma con DNIe utiliza la función resumen SHA-1 (véase en [5]) y sigue los mismos pasos que en 5.4.1, pero como estamos ante una firma reconocida, junto a la firma electrónica del ciudadano debe estar presente el certificado electrónico firmado por la Autoridad de Certificación, en este caso la DGP.

2. Recordar que el DNIe tiene almacenado en el *chip* un certificado electrónico que contiene la información personal del firmante junto con su clave pública. En el momento de expedición del DNIe la DGP genera un documento G, que contiene la anterior información. Posteriormente calcula el resumen de G mediante una función resumen SHA-256, $h(G) = g$, obteniendo así un resumen de 256 bits y finalmente firma el resumen g, mediante su clave privada RSA, d_{PN}:

$$f_{PN} = g^{d_{PN}}$$

   así G junto con la firma electrónica de su resumen constituyen el certificado electrónico.

3. Finalmente el usuario S manda la 4-tupla formada por el mensaje M, su firma electrónica f, el documento electrónico generado por la Policía Nacional G, y la firma electrónica de la policía f_{PN}. Es decir el usuario S manda la 4-tupla: $[M, f, G, f_{PN}]$.

5.5.3. Verificación de firma electrónica con DNIe

El usuario V recibe el mensaje firmado por S, es decir recibe la tupla $[M, f, G, f_{PN}]$. V puede obtener la clave pública de S del certificado electrónico firmado por la Dirección General de Policía. Pero primero se tiene que asegurar que ese certificado ha sido expedido en realidad por la DGP, para ello, sigue los siguientes pasos:

1. Previamente, V ha tenido que descargarse la clave pública de la Dirección General de Policía en su ordenador, de esa forma cuando reciba el mensaje firmado por S, el ordenador de V aplicará la clave pública de la DGP a la firma electrónica de la DGP que se encuentra en el certificado electrónico obteniendo así el resumen de los datos del certificado:

$$(f_{PN})^{e_{PN}} = (g^{d_{PN}})^{e_{PN}} = \widetilde{g}$$

   y comprueba que en efecto $\widetilde{g} = g$. Así, V se asegura que el certificado electrónico que contiene la clave pública de S lo ha expedido la DGP.

2. Ahora V puede comprobar que en efecto S ha firmado el mensaje M, para ello toma la clave pública de S del certificado electrónico expedido por la DGP y sigue el protocolo 5.4.2.

5.6. Conclusión

A modo de conclusión, es preciso señalar la gran utilidad que tiene el DNIe en la sociedad moderna, a causa de la rápida informatización de la vida cotidiana. El DNIe nos ofrece la posibilidad de identificarnos de forma rápida y segura, al mismo tiempo que la firma electrónica nos posibilita firmar una infinitud de documentos. A pesar de las claras ventajas que presenta el DNIe, su uso no se ha extendido entre la sociedad. Esto podría deberse a la necesidad de tener un lector de tarjetas y descargar diversos paquetes en el ordenador. A partir del año 2022 esto cambiará con la introducción del DNIe 4.0, el cual permitirá tener el DNIe integrado en el móvil y por el cual se podrá acceder a sitios web en internet mediante su función de identificación e incluso firmar electrónicamente desde el móvil. Así se prevé que en los años venideros el DNIe cobrará una mayor relevancia en la vida de la sociedad española.

Bibliografía

[1] RONALD JOSEPH BURTHE JR., *Further Iinvestigations with the strong probable prime test, Mathemathics of Computation, Volume 65, Number 213, January 1996, Pages 373-381.*, disponible en https://www.ams.org/journals/mcom/1996-65-213/S0025-5718-96-00695-3/S0025-5718-96-00695-3.pdf.

[2] COMISIÓN TÉCNICA DE APOYO A LA IMPLANTACIÓN DEL DNIE, *Guía de Referecnia Básica del DNIe, Versión 1.4, Julio 2014*, disponible en https://www.dnielectronico.es/PDFs/Guia_de_referencia_basica_v1_4.pdf

[3] KEITH CONRAD, UNIVERSITY OF CONNECTICUT, *An article of The Miller–Rabin Test*, disponible en https://kconrad.math.uconn.edu/blurbs/ugradnumthy/millerrabin.pdf.

[4] FERNANDO CORBALÁN, *La proporción áurea, página 13, RBA, 2010*

[5] DECLARACIÓN DE PRÁCTICAS Y POLÍTICAS DE CERTIFICACIÓN, *Ministerio del Interior, versión 2.2, junio 2017*, disponible en https://www.dnielectronico.es/PDFs/Politicas_de_Certificacion_v2.2.pdf.

[6] WHITFIELD DIFFIE Y MARTIN E HELLMAN, *New directions in cryptography, IEEE Transactions on Information Theory, vol. it-22, no. 6, november 1976*

[7] DOC 9303, *Documentos de viaje de lectura mecánica, séptima edición, 2015*, disponible en https://www.icao.int/publications/Documents/9303_p3_cons_es.pdf.

[8] R. DURÁN DÍAZ, L.HERNANDEZ ENCINAS, J.MUÑOZ MASQUÉ, *El criptosistema RSA, Ra-Ma, Instituto de Física aplicada, C.S.I.C. Madrid, 2005*

[9] R. DURÁN DÍAZ, L.HERNANDEZ ENCINAS, J.MUÑOZ MASQUÉ, *Generación de primos: una perspectiva computacional, 2004*, disponible en http://digital.csic.es/bitstream/10261/33131/1/recsi11.pdf

[10] LFE, *Ley 59/2003, de 19 de diciembre, de firma electrónica.*, disponible en https://www.boe.es/buscar/act.php?id=BOE-A-2003-23399.

[11] A. MENEZES, P. VAN OORSCHOT AND S. VANSTONE, *Handbook of Applied Cryptography, capítulo 4, CRC Press, Octubre 1996*, disponible en https://cacr.uwaterloo.ca/hac/

[12] G. MILLER, *"Riemann's hypothesis and tests for primality", en J. Comput. System Sci., vol.13, pp. 300-317, 1976.*

[13] NIST, *National Institute of Standards and Tecnhnology, 1995*, disponible en https://csrc.nist.gov/publications/detail/fips/180/1/archive/1995-04-17.

[14] NIST, *National Institute of Standards and Tecnhnology, 2002*, disponible en https://csrc.nist.gov/csrc/media/publications/fips/180/2/archive/2002-08-01/documents/fips180-2.pdf.

[15] PORTAL DEL DNIE, *Gráfico descriptivo de la información física visible en la tarjeta*, disponible en http://dnielectronico.us/Asi_es_el_dni_electronico/descripcion.html

[16] J.-J. QUISQUATER AND C. COUVREUR, *Fast Decipherment Algorithm for RSA Public-Key Cryptosystem. Electronics Letters, 18 (21), pp. 905-907, Octubre 1982*

[17] M. RABIN, *"Probabilistic algorithms for testing primality", en J. Number Theory, vol. 12, pp. 128-138, 1980.*

[18] ANA RÍO, *DNIe, La Gaceta de la RSME, Vol. 13 (2010), Núm 2, Págs 283-303*, disponible en https://gaceta.rsme.es/abrir.php?id=927

[19] R.L. RIVEST, A.SHAMIR, AND L. ADLEMAN, *A Method for Obtaining Digital Signatures and Public Key Cryptosystems, MIT Laboratory for Computer Science and Department of Mathematics, February 1978.*

[20] RSA LABORATORIES, *PKCS#1, RSA Encryption Standard , June 3, 1991*

[21] RSA LABORATORIES, B. KALISKI, *PKCS#1: Versión 1.5, Marzo 1998*, disponible en https://datatracker.ietf.org/doc/html/rfc2313.

Firmado por SARASA LABORDA, SEBASTIAN VENTURA (FIRMA) el día 27/06/2021 con un certificado emitido por AC DNIE 005