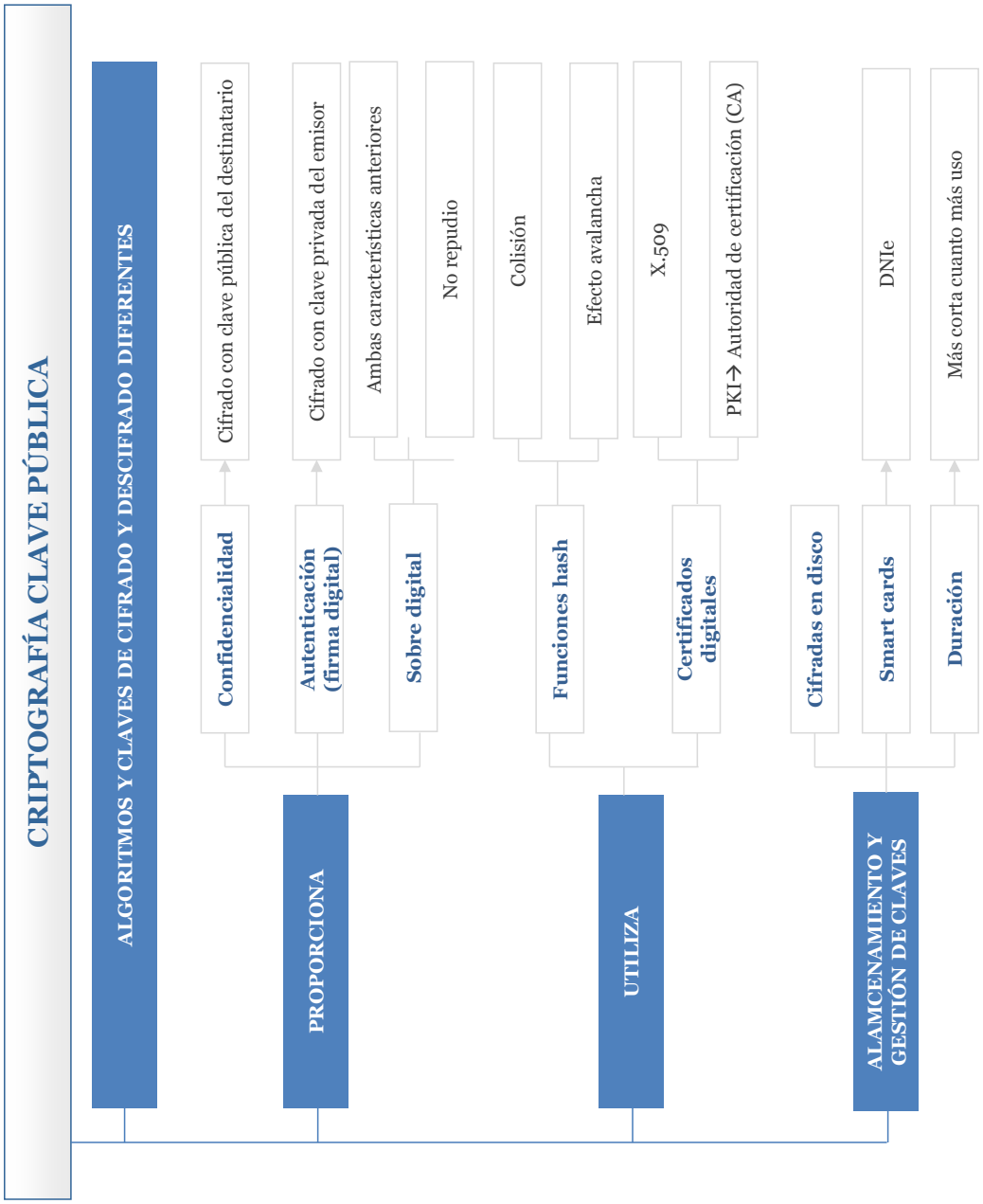
Criptografía asimétrica

TEMA 4

Esquema


CRIPTOGRAFÍA CLAVE PÚBLICA
ALGORITMOS Y CLAVES DE CIFRADO Y DESCIFRADO DIFERENTES
PROPORCIONA
UTILIZA
ALAMCENAMIENTO Y GESTIÓN DE CLAVES
Confidencialidad
Autenticación (firma digital)
Sobre digital
Funciones hash
Certificados digitales
Cifradas en disco
Smart cards
Duración
Cifrado con clave pública del destinatario
Cifrado con clave privada del emisor
Ambas características anteriores
No repudio
Colisión
Efecto avalancha
X.509
PKI→ Autoridad de certificación (CA)
DNIe
Más corta cuanto más uso

Ideas clave

4.1. ¿Cómo estudiar este tema?

El estudio de este tema se realiza a través de los contenidos desarrollados en las **Ideas clave** expuestas a continuación.

En este tema continuaremos con nuestro **recorrido histórico de la criptografía**, llegando a la **década de los años 70** y el **nacimiento de la criptografía de la clave pública**. Este nuevo paradigma supuso una auténtica revolución, ya que permitió resolver problemas que hasta ese momento impedían que la criptografía se hiciera ubicua.

Junto con la explosión de Internet, la **criptografía de clave pública** ha sido un pilar fundamental en el desarrollo del comercio electrónico, la banca online y, en general, cualquier otro área telemática que necesite seguridad en su definición y operación.

En este tema estudiaremos en general, cómo este esquema soluciona los eternos problemas del establecimiento y distribución de claves o la confidencialidad y autenticación, a través de nuevos mecanismos como la **firma** o los **certificados digitales**.

4.2. Orígenes de la criptografía asimétrica o de clave pública

Como hemos visto en los temas anteriores, la **criptografía simétrica o de clave secreta** ha arrastrado siempre a lo largo de toda su larga historia una serie de problemas irresolubles. Entre ellos, el principal es el del **establecimiento y distribución de claves**.

Este problema hace referencia a cómo acordar o distribuir una clave secreta entre las distintas partes involucradas en una comunicación antes de que esta tenga lugar. Cuando el número de comunicantes es pequeño, este problema puede ser resulto con algo de imaginación y trabajo. Pero en la segunda mitad del siglo XX, a partir de 1960, cuando la criptografía salió del ámbito exclusivamente militar y llegó al mundo civil, este problema se volvió un obstáculo insalvable.

Por ejemplo, para una empresa. «¿Cómo iba una empresa, con miles de trabajadores dispersos además geográficamente, a establecer un esquema de cifrado que permitiese las comunicaciones seguras entre todos ellos?» Esta pregunta obtendría por fin respuesta en la década de 1970, cuando dos profesores de la Universidad de Standford, Whitfield Diffie y Martin Hellman, tras trabajar durante años en el problema, encontraron finalmente una solución.

Así, en mayo de 1976 estos profesores publicaron un famosísimo artículo llamado «New Directions in Cryptography» (1976), donde se presentaba un paradigma criptográfico radicalmente nuevo que parecía aportar una solución directa al eterno problema de la distribución de claves.

A diferencia de los tradicionales sistemas simétricos, los de clave pública **utilizan un par de claves distintas y relacionadas matemáticamente entre sí, de forma que lo se cifra con una de ellas sólo puede ser descifrado con la otra y viceversa**. Obviamente, además el sistema debe ser construido de forma que **no sea posible deducir una clave a partir de la otra** (más allá del siempre disponible para un atacante método de fuerza bruta).

De esta forma, en este tipo de algoritmos **sólo se necesita una clave por miembro del sistema**, ya que haremos que la clave no deducible actúe como clave privada, que debe ser mantenida en secreto y únicamente conocida para su propietario, y que la otra sea la clave pública, que puede ser distribuida libremente sin restricciones. Obviamente no es necesario entonces usar canales seguros para el intercambio de dicha clave pública, y mientras la privada permanezca secreta, no hay problema en utilizar el mismo par de clave pública-privada durante largos periodos de tiempo.

¿Quieres saber aplicaciones de actualidad que están basadas en este tipo de criptografía?, la herramienta por excelencia para acceder a la Deep Web, a la Internet profunda: TOR (*The Onion Router*), una red superpuesta sobre internet para el intercambio de información con capacidad de no revelar la identidad de los usuarios, es decir, su IP, manteniendo la integridad y el secreto de la información que viaja por ella.

4.3. Conceptos básicos y funcionamiento

De forma similar a un **esquema de clave simétrica**, uno de clave asimétrica o pública tiene una serie de elementos básicos, como el *texto en claro*, el *texto cifrado o criptograma* y los *algoritmos de cifrado y descifrado* (que, en este caso, suelen ser diferentes). Ya hemos visto que donde radica la principal diferencia, en la clave; en este caso, la clave simétrica tradicional se sustituye por un par de claves, compuesto por una pública y otra privada, que denotaremos K_{pub} y K_{priv}, respectivamente.
Por otro lado, denotaremos como `E[K(A), X] y D[K(A), X]` a las operaciones de cifrado y descifrado con la *clave K*, perteneciente a la *persona A*, del *texto en claro X*, respectivamente.

Armados con estos elementos, ya podemos introducir el funcionamiento de un esquema genérico de clave pública (puedes encontrar un resumen en la **Figura 1**). A grandes rasgos, dos usuarios que deseen comunicarse utilizando este tipo de esquema tendrían que realizar **los siguientes pasos**:

1. **Cada usuario genera un par de claves**, que usará a partir de ese momento para el cifrado y descifrado de los mensajes. Este par de claves puede usarse durante años, por lo que se tendrá que tener cuidado en la custodia de la clave privada.
2. **Cada usuario coloca su clave pública en un repositorio o directorio público**, de forma que cualquier otro usuario pueda encontrarlas.
3. Si Bob quiere enviar un mensaje confidencial a Alicia, éste debe **cifrar el mensaje con la clave pública** de Alicia, que puede encontrar en el repositorio anterior o la propia Alicia puede haberle enviado por cualquier método. En este caso, al tratarse de una clave pública, **no es necesario un canal seguro** (aunque sí auténtico; abordaremos este problema más adelante).
4. Cuando Alicia recibe el mensaje, **lo descifra utilizando su clave privada**, que sólo ella conoce y que como hemos visto, **está vinculada matemáticamente a la clave pública**. Nadie más que quien posea la clave privada correspondiente a la pública con la que se cifró el mensaje podría descifrarlo.

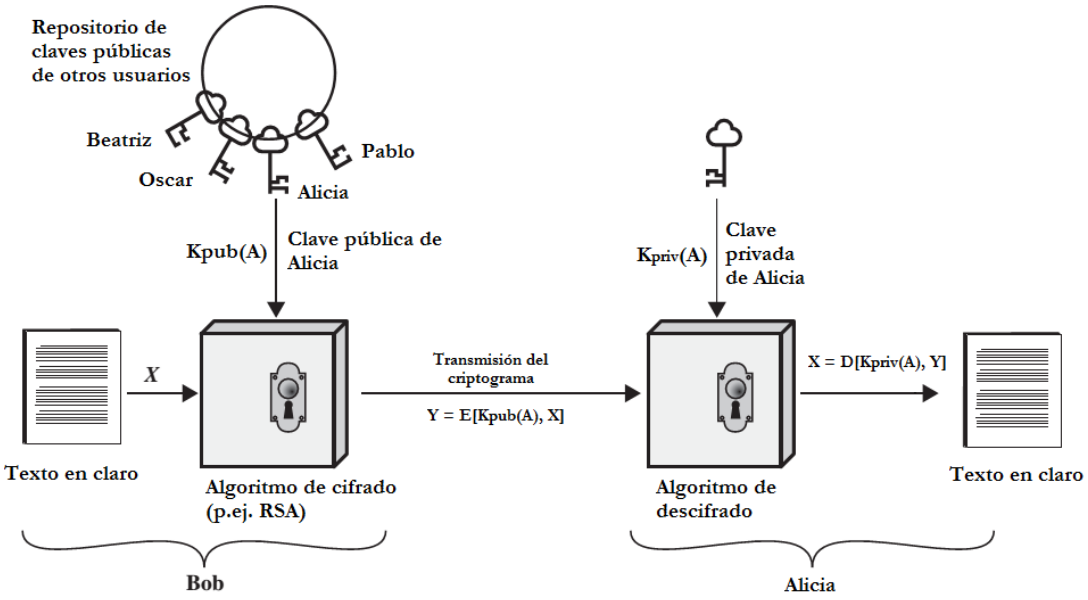


Figura 1. Esquema general de un esquema de cifrado de clave pública

En este esquema es necesario como vemos, que **los participantes tengan acceso a las claves públicas del resto** y que l**as claves privadas correspondientes sean generadas localmente antes de la comunicación**. Por supuesto, un usuario puede modificar su par clave pública-privada en cualquier momento; para ello, sólo debe generar el nuevo par y publicar la clave pública de nuevo en el repositorio correspondiente.

En la **Tabla 1** encontrarás un resumen de las principales diferencias y semejanzas entre los dos grandes paradigmas de cifrado. No olvides las diferencias de nomenclatura: cuando se habla de claves simétricas, éstas deben denominarse claves secretas, para diferenciarlas de su equivalente en esquemas asimétricos, la clave privada).

Criptografía simétrica	Criptografía asimétrica o de clave pública
Requisitos de funcionamiento	
1. El esquema necesita el mismo algoritmo de cifrado y descifrado con la misma clave 2. Emisor y receptor deben compartir algoritmo y clave 3. Para evitar confusiones, éstas calves se denominan secretas (a diferencia de la pública y privada del esquema asimétrico)	1. Se utiliza un algoritmo para el cifrado y otro para el descifrado, con un par de claves, una para cada operación 2. Emisor y receptor deben poseer una (la pública) del par de claves de la otra parte
Requisitos de seguridad	
1. La clave debe permanecer secreta 2. Debe ser imposible (o computacionalmente imposible) descifrar un mensaje sin información adicional 3. La posesión por un atacante de criptogramas y el algoritmo de cifrado debe ser insuficiente para determinar la clave	1. Una de las claves (la privada) debe permanecer secreta 2. Debe ser imposible (o computacionalmente imposible) descifrar un mensaje sin información adicional 3. La posesión por un atacante de criptogramas y una de las claves (la pública) debe ser insuficiente para determinar la otra clave

Tabla 1. Comparativa esquemas simétricos y asimétricos

Como hemos mencionado anteriormente, cualquiera de las dos claves puede ser utilizada para la operación de cifrado, utilizando la otra entonces para el descifrado. Este simple cambio genera todo un nuevo esquema criptográfico, como vamos a ver. Ya sabemos que el esquema propuesto en la Figura 1, en el que se cifra el mensaje con la clave pública de Alicia, $K_{pub}(A)$, proporciona **confidencialidad** al mensaje. Efectivamente, nadie más que Alicia, el receptor del mensaje, podrá recuperar su contenido.

¿Qué ocurre, sin embargo, si en lugar de cifrar con la clave pública del destinatario, lo hacemos con nuestra clave privada (considerando que nos ponemos en lugar del emisor)?. En ese caso, debido a la relación entre las claves (recordemos, lo que se cifra con una solo puede ser descifrado con la otra), cualquiera con nuestra clave pública podría descifrar el mensaje.

Si realiza la operación y efectivamente se recupera el mensaje original, ¿qué significa este hecho? Pues obviamente que sólo el emisor, con su clave privada que sólo él conoce, pudo cifrar el mensaje, puesto que ha sido correctamente descifrado con la clave pública de dicho emisor. Por tanto, acabamos de comprobar en un sólo paso la **autenticidad** (que proviene de quien se supone) y la **integridad** del mensaje (pues que si hubiera manipulado en tránsito, el descifrado hubiera fallado). Esta operación de comprobación se denomina ***firma digital***.

Una forma de no olvidar **qué operación se debe realizar para conseguir confidencialidad y autenticación** es el siguiente esquema:

Cifrar con la clave pública del destinatario → **Confidencialidad**

Cifrar con la clave privada del emisor →**Autenticación (firma digital)**

Sin embargo, el esquema anterior tiene un gran inconveniente: **el espacio de almacenamiento necesario**. Para que el receptor pueda comprobar la firma digital en cualquier momento, se debe conservar el criptograma en el tiempo, cuando normalmente este se desecha al descifrar el fichero. Una forma de solucionar el problema sería cifrar con la clave privada (firmar) únicamente un pequeño bloque de información que sea un función de todo el documento, y que pueda regenerarse fácilmente. Para que funcione, obviamente debe tener la propiedad de que sea imposible cambiar el mensaje sin que cambie el valor de esta función. Afortunadamente existen este tipo de funciones, y se denominan ***funciones hash***.

Funciones hash

Aunque analizaremos este tipo de funciones con mayor detalle en el próximo tema del curso, haremos aquí una breve **introducción y una descripción de sus principales propiedades de las *funciones hash***, puesto que son necesarias para entender correctamente los esquemas de clave pública que estamos estudiando.

Las *funciones hash* son, en esencia, **construcciones criptográficas que reciben un mensaje de longitud arbitraria y generan, en base a dicho contenido, un número identificador único de longitud fija**. Este número, denominado simplemente *hash del mensaje*, sirve por tanto para identificarlo de forma única, pues cualquier pequeño cambio en el mensaje hará cambiar también el resultado de la función hash sobre él.

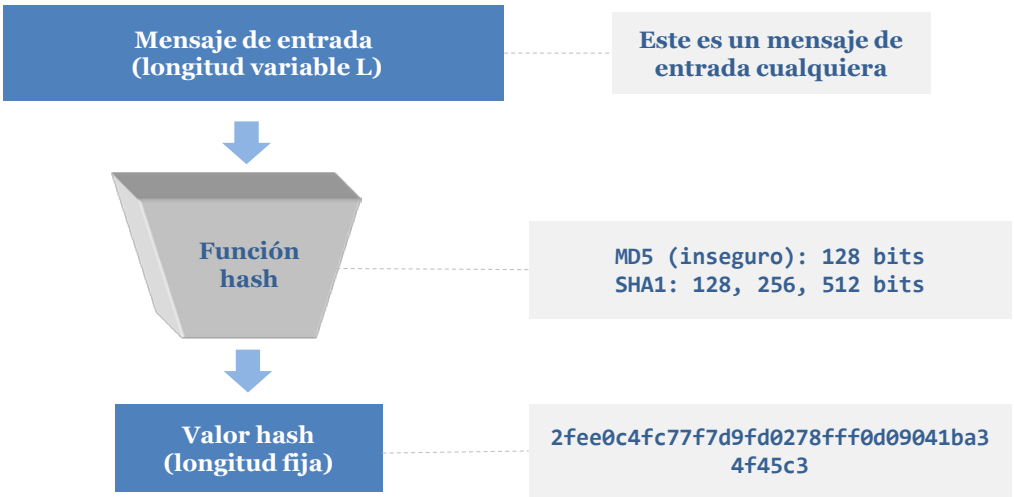


Figura 2. Esquema general de las funciones hash

Además de esta propiedad, una ***función hash* de uso criptográfico** debe cumplir también otra serie de requisitos:

- Es computacionalmente imposible «darle la vuelta» a la función, es decir, conocido el valor hash de un mensaje no debe ser posible recuperar el contenido del mismo (más allá, como es habitual, de un ataque de fuerza bruta; esto es, probar todos los posibles mensajes hasta dar con el que genera el valor hash correcto). En este sentido, su otra denominación, *one way function* o función de un solo sentido, es más descriptiva.

- No es probable encontrar dos mensajes distintos que generen el mismo valor hash, lo que se conoce como **colisión**. Piensa que esto rompería completamente el esquema, pues, dado un valor hash, no podríamos asegurar que pertenece a un mensaje concreto M, o a cualquiera de sus otras colisiones. Para una función hash sin debilidades, que genere una salida de 160 bits, las probabilidades de encontrar una colisión son de una contra 1.000.000.000.000.000.000.000.000.000.000.000.000.000.000.000.00 0.
- El cambio de un único bit en el mensaje de entrada debe modificar, en término medio, la mitad de los bits del valor hash correspondiente, lo que se conoce como **efecto avalancha**. En otras palabras, el más mínimo cambio en el mensaje cambia completamente el valor de su hash.

Existen multitud de *funciones hash*, que se usan constantemente en el mundo de la criptografía, aunque las más conocidas son:

MD5: presentado en 1992 y diseñado por Ron Rivest, famoso criptógrafo, genera una salida, un hash, de 128 bits. Ha sido utilizado durante muchos años, hasta que fueron descubiertos fallos importantes en su diseño, que lo hacen **completamente inseguro**. Es decir, aunque esta función sigue utilizándose y está disponible en casi todas las librerías criptográficas por razones históricas de compatibilidad, **no debe utilizarse** en ningún nuevo desarrollo.
SHA, que fue desarrollado por el NIST, y que genera valores de 160 bits. En 1995 fue revisado, pasándose a denominar SHA1. A pesar de ser más seguro que MD5, en 2005 fueron identificadas también una serie de vulnerabilidades en su diseño, que hacen aconsejable utilizar la última y nueva revisión SHA2, que puede generar resúmenes de 160 hasta 512 bits.

Firma digital

Ahora que entendemos qué es una *función hash* y para qué sirve, podemos retomar en este punto el concepto de **firma digital**. Recuerda que buscamos **una forma de proporcionar autenticación e integridad a un mensaje mediante el esquema de clave pública**.

Una primera idea que podría venirnos a la mente es usar las funciones hash que acabamos de ver. Podríamos generar el hash del mensaje y enviarlo junto con este al receptor. Este recibiría ambos elementos, volvería a calcular el hash del mensaje y si coincide con el recibido, daría por válido el mensaje. Sin embargo, obviamente, este procedimiento no funciona, puesto que un atacante podría interceptar el mensaje en tránsito, modificarlo, recalcular la función hash y dejar al mensaje proseguir su camino. El receptor, sin embargo, no podría notar la manipulación.

Recuerda por tanto, que una función hash por sí misma, no proporciona autenticación, puesto que cualquiera puede regenerar el valor hash tras modificar el mensaje de entrada. Necesitamos, pues, «mezclar» las funciones hash con el esquema de cifrado de clave pública que ya hemos visto para obtener la solución, que puedes encontrar en la **Figura 3**.

Su funcionamiento se resume como sigue:

1. El emisor calcula la función hash del mensaje, h, y cifra este valor con su clave privada, obteniendo así S, que será ya la firma digital del mensaje.
2. El emisor envía el mensaje M junto con su firma digital S al receptor.
3. El receptor descifra el valor de la firma con la clave pública del emisor, recuperando un valor h'.

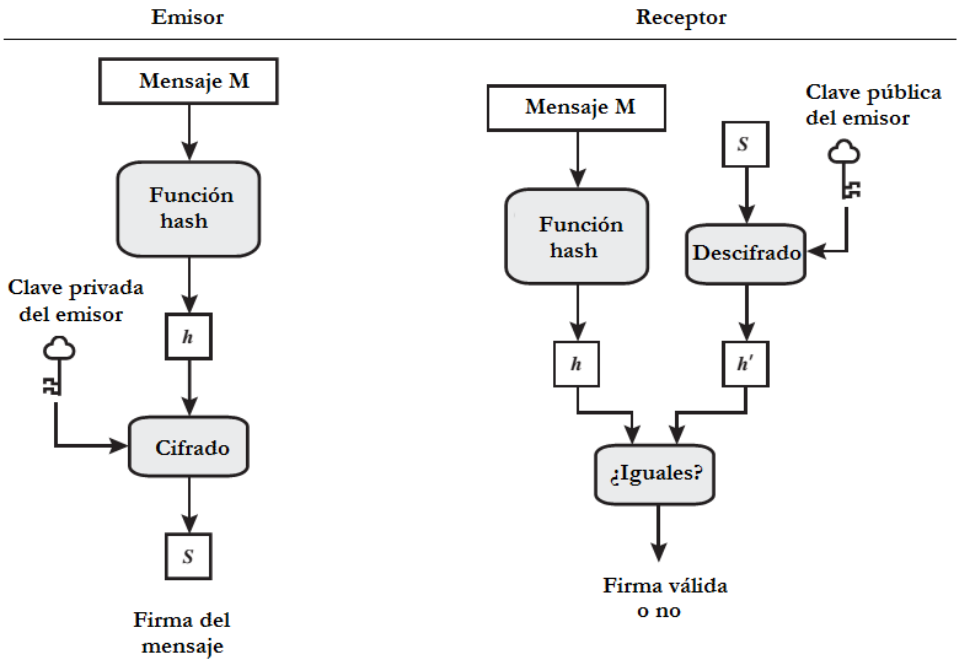


Figura 3. Esquema general de funcionamiento de la firma digital

4. Acto seguido, procesa el mensaje *M* (que viene en claro, recuerda que este esquema el mensaje no se cifra, sólo estamos proporcionando autenticación de momento) a través de la misma función hash que utilizó el emisor y obtiene otro valor *h*.
5. Obviamente si *h* y *h'* son iguales, significa que el emisor firmó el mensaje (puesto que hemos descifrado el valor de *S* con su clave pública) y que éste no ha sido manipulado (pues entonces, el valor de *h* y *h'* no coincidirían).

Además, aunque no seamos conscientes todavía, este esquema tan sencillo cuenta con otra característica necesaria para poder hablar de verdadera seguridad de la información: el **no repudio**. Este concepto hace referencia a un mecanismo que evite que el emisor de un mensaje pueda negar su autoría sobre él.

Esto es especialmente importante en determinados escenarios. Imagina por ejemplo, un sistema de firma digital de contratos electrónicos. Más ahora que en España la firma digital tiene plena validez legal, y ha sido equiparada completamente a la firma manuscrita tradicional. Si fuera posible que una de las partes firmantes negara haber sido el autor de la firma digital, todo el sistema se vendría abajo y no sería confiable.

En el esquema que acabamos de ver, dado que sólo el emisor pudo generar la firma del mensaje, si esta se verifica correctamente, dicho emisor debe ser necesariamente el origen del mensaje, por lo que también podemos garantizar el no repudio del mensaje.

En realidad, estrictamente hablando, lo que podemos asegurar matemáticamente es que la firma la generó alguna persona que tenía acceso a la clave privada correspondiente. El emisor siempre puede alegar que algún tipo de malware robó sus credenciales, o que se le mostró un documento que no era realmente el que él creía estar firmando. Para evitar estos problemas, existen algunas soluciones que estudiaremos en la sección sobre la gestión de la clave privada.

¿Ingenioso, verdad? Sin embargo, recuerda que **con este esquema sólo obtenemos un método para comprobar la autenticidad del mensaje**, puesto que lo hemos enviado en claro. ¿Es posible, por tanto, para cerrar el círculo, cifrar y firmar el mensaje a la vez? Por supuesto, veamos cómo a continuación.

Juntando todas las partes: composición final

Debes recordar del tema anterior, dedicado al cifrado simétrico, que «**una de las diferencias más importantes entre el cifrado simétrico y el asimétrico es la enorme disparidad en la velocidad de cifrado»**. Los algoritmos asimétricos son realmente lentos en comparación con los simétricos; la diferencia puede ser, de hecho, hasta de tres órdenes de magnitud, es decir, más de 1000 veces más lentos.

Otra razón es que los esquemas de clave pública son vulnerables a los ataques denominados de **texto claro escogido** (*chosen-plaintext attacks*, en inglés). En efecto, ya que disponemos de la clave pública de cualquier destinatario, podemos cifrar todos los posibles textos en claro y comparar el resultado con el mensaje interceptado. Si son iguales, podemos deducir el contenido del mensaje sin tener que criptoanalizarlo ni obtener la clave privada correspondiente.

Obviamente, para realizar este ataque en la práctica es necesario conocer previamente algo de información sobre el posible mensaje, pues el espacio de posibles textos en claro, sin información adicional, puede ser tan grande como el de claves simétricas. Sin embargo, el ataque es posible y peligroso en la práctica. Imagina, por ejemplo, que, debido a cierta información privilegiada, sabes que el mensaje cifrado interceptado contiene un modelo de contrato, pero no la cantidad económica exacta que figura en él.

Las combinaciones a probar, entonces, serían muy pocas (unos pocos millones), que podrían probarse en cuestión de minutos.

Por estas razones, los esquemas de clave pública no deberían utilizarse directamente para cifrar datos, sino para cifrar claves simétricas que, a su vez, cifren los datos. Este es un pequeño truco para obtener lo mejor de los dos mundos, llamado **clave de sesión**.

Esta es una clave secreta de corta duración, que se genera y utiliza únicamente para el envío de un mensaje y luego se desecha. La diferencia es importante porque cifrar, por ejemplo un fichero de muchos megabytes, o incluso gigabytes, con un esquema puro asimétrico es muy, muy lento (aparte de inseguro, como acabamos de ver).

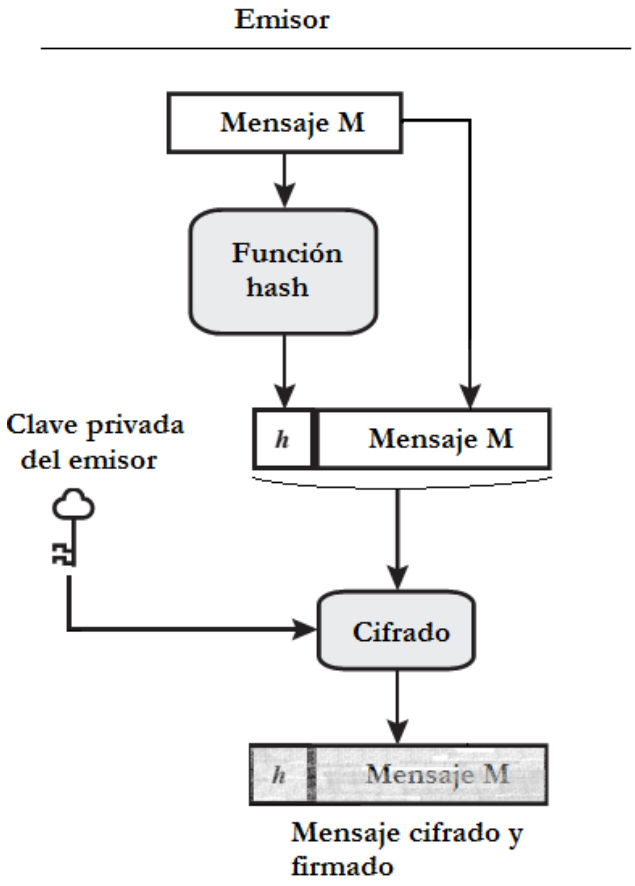


Figura 4. Primera aproximación al esquema genérico de cifrado y firmado simultáneo de un mensaje

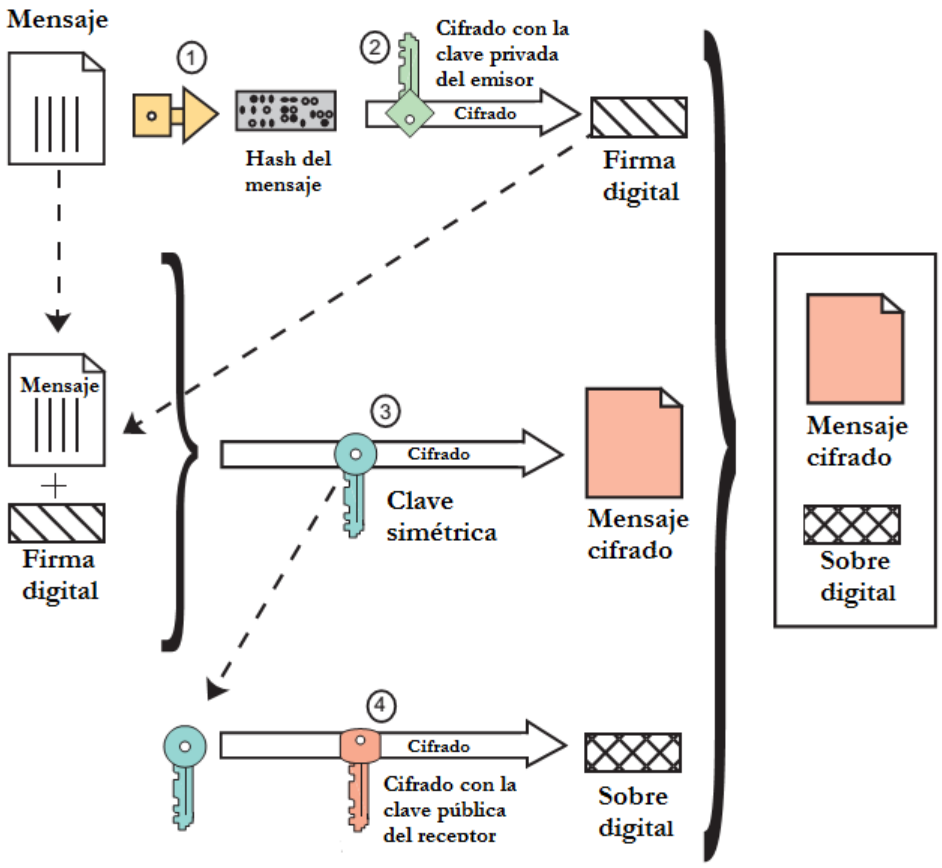


Figura 5. Esquema genérico de esquema híbrido de cifrado moderno, que incluye el uso de un sobre digital

Esta combinación de esquemas simétricos y asimétricos se conoce con el nombre de ***sobre digital***, y es que el que se utiliza en el mundo real para envío de correos electrónicos, ficheros, etc. Aunque no existe un estándar único, en general estos esquemas funcionan de la siguiente manera:

1. Se calcula una **firma digital** del mensaje, cifrando con la clave privada del emisor un hash del mismo (pasos 1 y 2 de la **Figura 5**).
2. Se concatena esta firma al mensaje en claro, y ambos se cifran con una clave simétrica generada para la ocasión (llamada de sesión), dando lugar al **mensaje cifrado** (paso 3).
3. Finalmente, la clave de sesión anterior se cifra con la clave pública del receptor, estructura que suele denominarse **sobre digital**, de forma que sólo dicho receptor pueda descifrar el mensaje.

Es importante destacar que en este esquema se firma el mensaje antes de cifrarlo. ¿Hay una razón para respetar este orden, en lugar del inverso? Lo cierto es que sí. Piensa en una analogía del mundo real para entenderlo mejor: **un sobre de correos**. Cuando firmamos una carta postal, primero la firmamos y luego la introducimos en el sobre. Si firmáramos el sobre en lugar de la carta, alguien podría cambiar el sobre entero con otro contenido.

En el mundo digital, se procede de la misma forma y, en general, es una máxima a cumplir: *primero firmar, luego cifrar*. No sólo es más seguro, sino que también tiene consecuencias legales: **si el texto a firmar no es visible cuando se adjunta la firma, todo el sistema podría ponerse en cuestión**, pues el emisor del mensaje podría argumentar que ese no fue el texto que se le mostró para ser firmado.

Para completar el sistema sólo faltaría un elemento más, denominado ***sello de tiempo*** (*timestamp* en inglés). Como es fácil de intuir, un sello de tiempo indica en qué momento exacto se generó el mensaje y sirve para evitar que éste vuelva a ser utilizado en el futuro (este tipo de ataque se llama *ataque de repetición*).

Llegados a este punto, ahora sí, **este esquema proporciona confidencialidad del mensaje (puesto que se transmite cifrado), autenticación e integridad (si se verifica correctamente la firma)**.

4.4. El algoritmo RSA

Como hemos comentado, Diffie y Hellman sentaron las bases de la **criptografía de clave pública** con su artículo seminal de 1976. Sin embargo, en dicho artículo se presentaba un esquema «genérico», pero ningún protocolo concreto que lo llevara a la práctica.

Habría que esperar un año a que los profesores del MIT norteamericano Ron Rivest, Adi Shamir y Len Adleman encontraran una solución práctica, basada en un viejo problema matemático: **la factorización de números**. Desde entonces, el ***esquema RSA*** (cuyo nombre proviene, obviamente, de las iniciales de cada uno de sus diseñadores) ha sido sin duda el algoritmo de clave pública más utilizado y aceptado.

Sin embargo, como ha ocurrido otras tantas veces en la historia, el verdadero mérito de este descubrimiento no corresponde a los profesores del MIT. Ya en el año 1969 el Government Communications Headquarters (GCHQ) en Gran Bretaña comienza a trabajar en la idea de la distribución de claves por métodos no simétricos. De hecho, en 1973, cinco años antes de la publicación de RSA, el matemático Clifford Cocks obtuvo un método virtualmente idéntico al algoritmo norteamericano. Sin embargo, dado que Cocks trabajaba en una organización secreta, su trabajo fue clasificado como alto secreto por el gobierno británico, lo que impedía radicalmente su publicación.

El ***algoritmo RSA*** se basa en el conocido problema de la **factorización de enteros**. Como sabes, factorizar un número significa encontrar su descomposición en producto de números primos. El **Teorema Fundamental de la Aritmética** asegura que para cualquier entero positivo esta factorización existe y es única.

Recordarás también el método de factorización más básico, la criba de Erastótenes, que no es más que un proceso iterativo de prueba y error. Obviamente este método sólo funciona para números de unas pocas cifras. Conforme los números crecen el método se vuelve terriblemente ineficiente. Por ejemplo, averiguar que el número `19383241754651` es el producto de los dos primos `3920201` y `4944451` puede llevar ya un buen rato a mano (aunque un ordenador sigue tardando centésimas de segundo).

Obviamente sabiendo los factores, reconstruir el número del que parten es trivial, pues basta con multiplicarlos. Este tipo de problemas basados en cuestiones que son muy fáciles de resolver en un sentido (multiplicar) y muy costosas en el otro (factorizar) se denominan **funciones unidireccionales**.

Descripción del algoritmo

El *algoritmo RSA* es un cifrador de bloque en el que el texto en claro y el criptograma se codifican como enteros entre `0` y $n-1$ para algún n. Un tamaño típico para n son `1024`, `2048` o incluso 4096 bits, por lo que n será siempre menor de 2^{1024}, 2^{2048} o 2^{4096}, respectivamente.

Para cifrar y descifrar se utilizan **exponenciaciones modulares** (para mantener los números por debajo de los valores anteriores). Así, para algún *texto en claro M* y *criptograma C*:

Cifrado: $C = Me \bmod n$

Descifrado: $M = Cd \bmod n = (Me)d \bmod n = Med \bmod n$

Donde *d* es un valor que sólo el receptor conoce. Por tanto, ya tenemos los elementos de un esquema de clave pública; en este caso, la **clave pública** la forman los valores $\{e,n\}$ y la **privada** los valores $\{d,n\}$. e suele denominarse **exponente público**, n **módulo** y d **exponente privado**. Para que este algoritmo funcione tienen que cumplirse además una serie de condiciones:

1. Es fácil encontrar valores de *e*, *d* y *n* tales que $M^{ed} \bmod n = M$, de forma que puedan descifrarse los mensajes.
2. Es fácil calcular $Me \bmod n$ y $Cd \bmod n$ para todos los valores de *M*.
3. Es imposible determinar *d* (la clave privada) dados *e* y *n* (la clave pública).

Seguridad del algoritmo

Existen **cuatro tipos básicos de ataques** al algoritmo RSA:

- **Ataques matemáticos** a la factorización, aunque a día de hoy ninguno ha conseguido romper su seguridad.
- **Ataques de temporización**, que se basan en el tiempo que tardan las operaciones de descifrado.
- **Ataques de texto cifrado elegido**, que hemos presentado en las secciones anteriores.
- **Ataques por fuerza bruta**.

Como ya hemos visto, la defensa contra el último tipo de ataques, la fuerza bruta, pasa por utilizar una clave de longitud adecuada (a día de hoy, sería recomendable 2048 bits en lugar de 1024). Obviamente, debe existir un equilibrio, pues a mayor longitud de clave mayor tiempo de cifrado y descifrado.

Del resto de ataques, en el que más se ha investigado en los últimos años ha sido en la optimización de la factorización, que ha ido mejorando considerablemente. Ya en 1977 los propios inventores de RSA retaron a los lectores de la revista Scientific American a descifrar un mensaje que se publicó en la columna de «Juegos matemáticos» del conocido Martin Gardner.

Ofrecieron 100$ de recompensa a quien pudiera romper el criptosistema y recuperar el texto en claro, algo que predijeron que ocurriría en unos 40 cuatrillones de años. Sin embargo, en abril de 1994 un equipo que formó un equipo de trabajo a través de Internet reclamó el premio, pues consiguió recuperar el mensaje cifrado tras ocho meses de trabajo.

Durante estos años RSA ha seguido ofreciendo premios a quien factorizara claves de cada vez mayor tamaño. La **Tabla 2** recoge la evolución en el tiempo del reto, y cómo se han ido consiguiendo romper las distintas longitudes de clave. Finalmente, en 2007 RSA decidió acabar oficialmente con los retos (algunos dicen que porque se asustaron de lo rápido que avanzaban), argumentando que «ahora que la industria tiene un conocimiento de las técnicas criptoanalíticas de los algoritmos de clave simétrica y asimétrica, estos retos han perdido el sentido».

Nº de dígitos	Nº de bits	Fecha de consecución
100	332	Abril de 1991
110	365	Abril de 1992
129	398	Junio de 1993
130	431	Abril de 1996
140	465	Febrero de 1999
155	512	Agosto de 1999
160	530	Abril de 2003
174	576	Diciembre de 2003
200	664	Mayo de 2005
232	768	Diciembre de 2009

Tabla 2. Retos de factorización de RSA y fechas de consecución

Software criptográfico

Uno de las *suites* criptográficas más conocidas, y que utilizaremos a menudo en esta asignatura es **OpenSSL**. Crear una clave RSA con este programa es muy sencillo, pues basta con un comando como el siguiente, que genera un par de claves pública-privada de 2048 bits de longitud:

```
test$ openssl genrsa -out privkey.pem 2048
Generating RSA private key, 2048 bit long modulus
.....................................+++
.+++
e is 65537 (0x10001)
```

Para extraer del archivo anterior las componentes públicas y privadas, puedes utilizar el siguiente comando:

```
test$ openssl rsa -pubout -in privkey.pem -out pubkey.pem
```

¿Tienes curiosidad por ver qué aspecto tiene, por ejemplo, la clave pública? Su formato más habitual, denominado **PEM**, la codifica en **base64**, de forma que pueda visualizarse fácilmente:

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2j7JQcKUxFtbdKOdKzgi
kL1NzH9zsZ7uE8t5NM2Wqx+nOj1sKgxc8jJ2f93iS55+bU1czV+irrI0mw4qjHjb
8avC5LxEtz9Ltx/70alDQn88AoCHiPsZY6K2n2+XRJjgtwgWpyynL7c1bbUgmv8o
U6fxO1vbLZKZoHrL+SDlirvOE3XclycVx4qid/NL/Uw+nJG9RXsuwu9sZt5CfRui
GUshEib48sPX+i5d1u0AFaSQ/Z1YQtfGhIa2AFgX8d3cZDIHSNDCOYIhYtd1xuNe
65hOGLPj8hT1YK2+npyI8U324XPHA+vns07s6gQ3ofvuWdYy3udauIh1/w4WqweZ
6wIDAQAB
-----END PUBLIC KEY-----
```

Pero hemos dicho que una clave pública está formada por un exponente y un módulo. ¿Dónde están estos valores? Se encuentran codificados en el texto anterior; para visualizarlos utiliza un comando como éste:

```
test$ openssl rsa -pubin -in pubkey.pem -text
```

Que genera la siguiente salida:

```
Public-Key: (2048 bit)
Modulus:
    00:da:3e:c9:41:c2:94:c4:5b:5b:74:a3:9d:2b:38:
    22:90:bd:4d:cc:7f:73:b1:9e:ee:13:cb:79:34:cd:
    96:ab:1f:a7:3a:3d:6c:2a:0c:5c:f2:32:76:7f:dd:
    e2:4b:9e:7e:6d:4d:5c:cd:5f:a2:ae:b2:34:9b:0e:
    2a:8c:78:db:f1:ab:c2:e4:bc:44:b7:3f:4b:b7:1f:
    fb:d1:a9:43:42:7f:3c:02:80:87:88:fb:19:63:a2:
    b6:9f:6f:97:44:98:e0:b7:08:16:a7:2c:a7:2f:b7:
    35:6d:b5:20:9a:ff:28:53:a7:f1:3b:5b:db:2d:92:
    99:a0:7a:cb:f9:20:e5:8a:bb:ce:13:75:dc:97:27:
    15:c7:8a:a2:77:f3:4b:fd:4c:3e:9c:91:bd:45:7b:
    2e:c2:ef:6c:66:de:42:7d:1b:a2:19:4b:21:12:26:
    f8:f2:c3:d7:fa:2e:5d:d6:ed:00:15:a4:90:fd:9d:
    58:42:d7:c6:84:86:b6:00:58:17:f1:dd:dc:64:32:
    07:48:d0:c2:39:82:21:62:d7:75:c6:e3:5e:eb:98:
    4e:18:b3:e3:f2:14:f5:60:ad:be:9e:9c:88:f1:4d:
    f6:e1:73:c7:03:eb:e7:b3:4e:ec:ea:04:37:a1:fb:
    ee:59:d6:32:de:e7:5a:b8:88:75:ff:0e:16:ab:07:
    99:eb
Exponent: 65537 (0x10001)
```

Obviamente, cuando crees tu propio par de claves, los valores numéricos concretos serán diferentes. No dudes en jugar con este software y estos comandos hasta comprender bien cómo funcionan.

4.5. Certificados digitales

Hemos comenzado el tema que uno de los grandes problemas que la criptografía de clave pública soluciona (o, al menos, facilita respecto a la de clave simétrica) es la **gestión de claves**. En esta sección veremos exactamente cómo.

Comenzaremos, sin embargo, viendo que los esquemas de clave pública tienen también sus propios problemas. Por ejemplo, sabemos que cada persona tiene su propia clave pública. Por tanto, cuando Alicia quiere comunicarse con Bob, tiene que obtener primero su clave pública. Puede conseguir esto de varias formas:

- La obtiene directamente de Bob.

- La obtiene de una base de datos centralizada.
- La obtiene de su propia base de datos (que previamente ha debido generar, por lo que esta opción, en realidad, no es una solución).

¿Puedes intuir el problema con la primera solución? Incluso si dejamos a un lado lo poco práctico de la solución (que la haría inviable en un entorno real), es posible realizar también un ataque simple de **suplantación**. Imagina que Alicia quiere comunicarse con Bob y le envía un correo electrónico (o le llama por teléfono) para pedirle su clave pública. El atacante sólo debe interceptar la comunicación y reenviar el correo cambiando la clave de Bob por la suya propia. A partir de ese momento podría hacer de intermediario, recibiendo un correo de Alicia, descifrándolo, volviéndolo a cifrar con clave pública de Bob y reenviándoselo. Ni Alicia ni Bob podrían darse cuenta del engaño.

En necesario, pues, un mecanismo que nos asegure que las claves públicas pertenecen realmente a quien creemos que pertenecen. Y eso es, ni más ni menos, un **certificado digital**: un vínculo criptográfico entre una clave pública (y su correspondiente clave privada) y la identidad de una «entidad». Habitualmente esta entidad será una persona, pero es muy común que sea un ordenador, o una aplicación web. Podríamos decir que, en cierta manera, un certificado es la versión digital de los tradicionales DNI (documentos de identidad) o carnés de conducir.

En este punto podríamos preguntarnos, ¿y quién es el encargado de crear el certificado? Es decir, de verificar y dar fe de que realmente la clave pública pertenece a una persona física, por ejemplo. Esta es la misión de las denominadas **Autoridades de Certificación** (o CA, *Certification Authority*, en inglés).

En función de la identidad a certificar, y el nivel de seguridad que se requiera, los requisitos concretos del proceso de certificación serán diferentes. Por ejemplo, si se trata de una persona se le podría exigir un documento de identidad, comprobar sus huellas dactilares o, incluso, medidas biométricas. Una vez satisfecha, la CA generará el certificado digital correspondiente y se lo entregará a dicha persona.

Entonces, ¿qué datos concretos contiene un certificado y cómo se genera? Existen varios formatos, de los cuales el más conocido es la *norma X.509v3*, pero, de forma genérica, un certificado no es más que una serie de datos identificativos de la entidad (como su nombre) firmados digitalmente por la CA correspondiente (véase la **Figura 6**). Como ahora el certificado es un fichero de datos firmados, para comprobar la identidad de una persona, basta con comprobar la firma de la CA sobre el certificado. Si es correcta, sabremos que el certificado presentado pertenece a quien dice ser.

El conjunto de certificados digitales, Autoridades de Certificación (y otros elementos que como Autoridades de Registro o listas de revocación de certificados, que no veremos en detalle) se conoce con el nombre genérico de **PKI**, acrónimo inglés de *Public Key Infraestructure*.

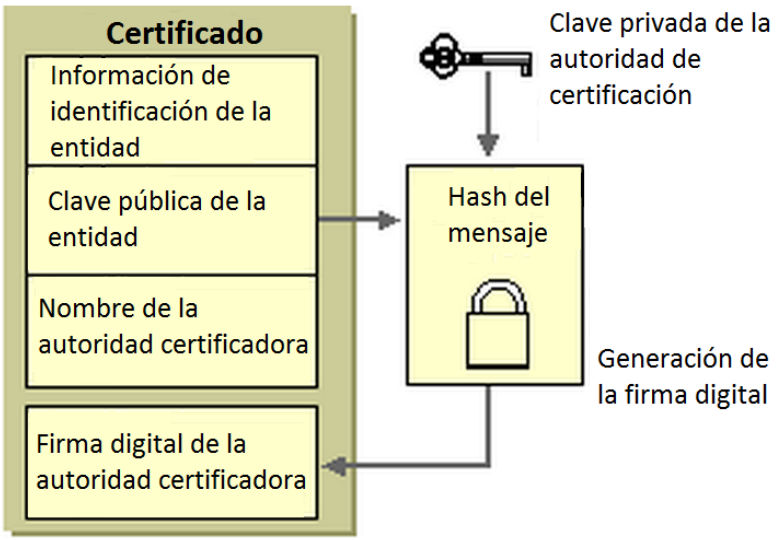


Figura 6. Estructura genérica de un certificado digital

Certificados X.509

Veamos ahora con mayor detalle el formato de **certificados X.509**, sin duda el más utilizado actualmente. Este formato es un **estándar ISO** que se publicó por primera vez en 1988, y ha ido refinándose con el tiempo. La versión actual es la número 3, publicada en 1996, y sus elementos concretos son:

- **Versión:** hace referencia al número de versión del certificado. Los posibles valores son 1, 2 y 3.

- **Número de serie:** es un número identificador único, asignado por la CA en el momento de la creación del certificado.
- **Identificador del algoritmo de firmado:** identifica el algoritmo empleado para firmar el certificado (como RSA).
- **Nombre del emisor:** identifica la CA que ha emitido y firmado el certificado.
- **Fechas de validez:** por razones de seguridad, un certificado o, en general, una clave criptográfica no puede durar eternamente. Este campo contiene el periodo de tiempo durante el que el certificado es válido, compuesto de una fecha inicial, en la que empieza a poder ser utilizado (es válido) y la fecha después de la cual el certificado deja de serlo.
- **Nombre del sujeto:** campo que identifica a la entidad certificada. Este «nombre» puede ser el nombre real de una persona, o el nombre de una máquina, en el caso de un ordenador.
- **Clave pública:** el campo más importante, que contiene la clave pública en sí.
- **Huella digital:** valor hash del certificado que, por tanto, sirve también para identificarlo de forma única.

Puedes ver el aspecto de un certificado real en tu navegador Web. Accede a la sección de seguridad (su localización variará en función del tipo concreto) y podrás ver los certificados instalados en el sistema (en la **Figura 7** se muestran ejemplos del *SO Windows*).

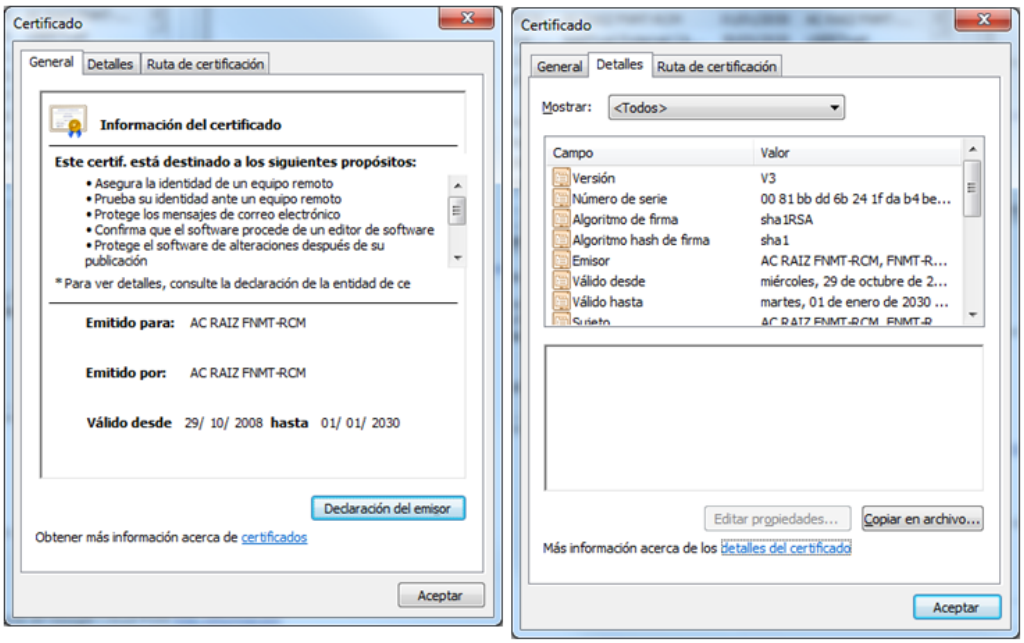


Figura 7. Certificado digital de la FNMT española. Observa, entre otros atributos, las fechas de validez

Usos de los certificados digitales

Los usos de los **certificados digitales** son innumerables, y tienen aplicaciones en prácticamente todos los ámbitos de la criptografía y de la seguridad de la información. A grandes rasgos, **un certificado digital puede utilizarse para garantizar**:

- **Confidencialidad:** puesto que un certificado contiene una clave pública, puede utilizarse para enviar información, normalmente a través de una clave simétrica de sesión, tal y como hemos visto en apartados anteriores.
- **Integridad:** firmando los mensajes con la clave privada asociada al certificado (recuerda que toda clave pública tiene una contrapartida privada). Como consecuencia positiva, una firma digital también asegura el no-repudio (siempre que se pueda garantizar que sólo el propietario ha podido tener acceso a la clave privada correspondiente).
- **Control de acceso y autenticación:** de nuevo, un certificado nos identifica de forma única, por lo que puede ser también utilizado para autenticar usuarios (sustituyendo las peligrosas contraseñas), o para controlar accesos a un edificio, acceso a servicios Web sanitarios, gubernamentales, etc.

Sin embargo, a pesar de sus múltiples ventajas, las PKI no son tampoco la panea de la seguridad de la información ni la solución a todos los problemas. Llevan, de hecho, muchos años en escena y no han terminado nunca de cuajar ni convertirse en esa tecnología definitiva que prometían ser.

Entre algunos de sus inconvenientes más importantes podemos citar:

- Una PKI es **técnicamente muy compleja y económicamente muy costosa**. Desplegarla en una gran empresa suele ser un proyecto costoso, lento y que suele sufrir, además, la reticencia de los trabajadores a su uso.
- Existen problemas asociados que son difíciles de resolver; por ejemplo, la **pérdida de un certificado** por parte de un usuario. En ese caso, hay que invalidar el certificado anterior y generar uno nuevo, lo que tiene un coste en tiempo y dinero.
- Existe también un problema de **confianza**, que está fuera del ámbito digital. Todo el sistema está sustentado en el hecho de que *confiamos* en las autoridades de certificación de forma ciega. Si una de ella decide emitir certificados falsos bajo soborno, por ejemplo, toda la cadena de confianza falla.

De hecho, se han producido incidentes reales como éste, cuando CAs como GlobalSign, Comodo o DigiNotar fueron *hackeadas*.

El último problema está relacionado con un concepto que aún no hemos presentado: el del **modelo de confianza** de las PKI. Este modelo está basado en el hecho de que una CA puede firmar certificados no sólo a entidades "finales", sino a otras CA, *delegando* su confianza en ellas. Firmar el certificado de otra CA significa, por tanto, que la primera le da permiso a la segunda para certificar en su nombre.

De esta forma se forman una serie de estructuras jerárquicas (cuyo primer nodo se llama **nodo o CA raíz,** CA_1 en la **Figura 8**), que pueden llegar a ser bastante complejas. Obviamente el certificado de la CA raíz debe ser auto-firmado, lo que significa que la CA ha emitido su propio certificado.

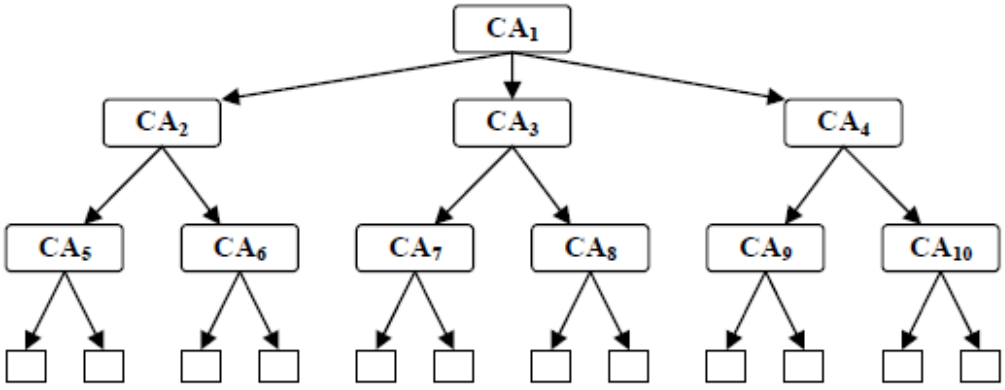


Figura 8. Jerarquía de certificación de distintas CAs

4.6. Almacenamiento y gestión de claves

Como ya sabemos, la criptografía de clave pública mejora mucho el problema de la distribución de claves pero, al tener también una componente privada, que debe permanecer *secreta*, debe resolver también la cuestión del **almacenamiento y gestión de claves**.

El método más sencillo y habitual es guardar la clave privada en disco cifrada, a su vez, por una clave simétrica derivada de una contraseña. Esta es la razón por la que, en los ejemplos de comandos reales que hemos anteriormente, el software te solicita una contraseña cuando el proceso de creación del par clave pública/privada.

¿Cómo se transforma una contraseña, que son una serie de caracteres ASCII, en una clave simétrica criptográfica, que será 128 o 256 bits completamente aleatorios? Existen diversos métodos, pero uno muy sencillo puede ser calcular el valor *hash* de la contraseña y utilizar esta salida como clave criptográfica.

Un sistema preferido sobre guardar la clave en el propio disco es utilizar hardware criptográfico específico, como una *smartcard*. Estas tarjetas contienen un chip especialmente protegido con medidas anti-manipulación de forma que pueden guardar información secreta, como el PIN de una tarjeta de crédito o una clave privada.

Un ejemplo de este tipo de tarjetas es el **DNI electrónico español** (véase la **Figura 9**), que almacena en su interior un certificado electrónico X.509 y la correspondiente clave privada (además de otros datos como una fotografía, una versión digital de la huella dactilar, entre otros). El certificado personal del **DNI electrónico** está firmado por la CA de la Dirección General de la Policía, de forma que pueda ser verificado por cualquier otra parte que así lo desee.

Figura 9. Elementos de seguridad de la tarjeta soporte del DNIe

De hecho ya pueden realizarse muchos trámites administrativos de forma telemática en España con este dispositivo, gracias al apoyo de la **Ley de Firma electrónica**. De esta forma, el DNIe puede utilizarse para «firmar digitalmente documentos electrónicos, otorgándoles una validez jurídica equivalente a la que les proporciona la firma manuscrita».

Tiempo de vida de claves criptográficas

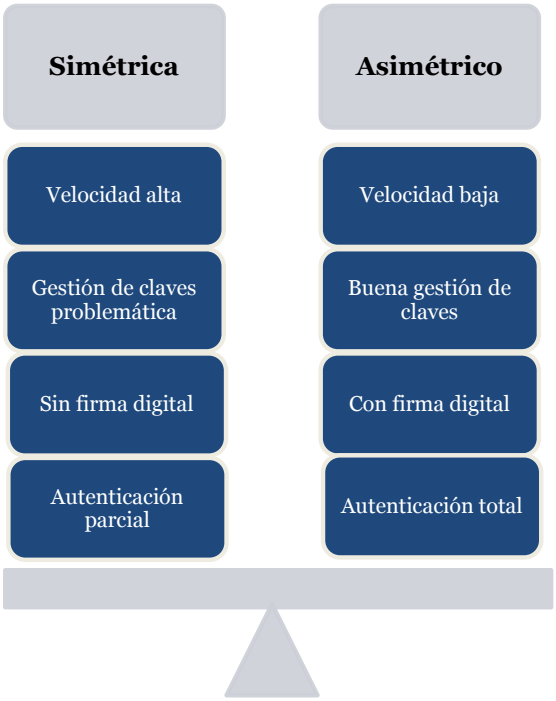
Resulta lógico pensar que ninguna clave criptográfica (ni contraseña de acceso) debería utilizarse durante un periodo ilimitado de tiempo. Las razones para esto son múltiples:

- **Si una clave o contraseña es comprometida sin que nadie se perciba, el atacante podrá hacer uso de la clave hasta que sea descubierto.** Por el contrario, si una contraseña caduca cada mes, por ejemplo, el atacante sólo dispondrá de un mes como máximo hasta que ésta deje de funcionar.
- **Cuanto más sea utilizada una clave, más posibilidades existen de que sea comprometida.** Desgraciadamente la gente pierdes las claves, o las escribe directamente en un post-it pegado en su monitor (sí, de verdad). Además, disponer de una gran cantidad de texto cifrado facilita la tarea de criptoanálisis para un atacante.
- **Cuanto más sea utilizada una clave, mayor es el daño en caso de compromiso. Esto resulta lógico, pues cuanta más información se haya cifrado con un clave, mayor será la pérdida en caso de que este clave sea vulnerada.** Por otro lado, si un atacante sabe que una clave no será cambiada nunca (o cada muchos años), mayor será su incentivo para atacarla, incluso si su única opción es la fuerza bruta.

Por todos estos motivos, cualquier aplicación que haga uso de criptografía debería disponer de una política clara que determine, entre otras cosas, el periodo de vida de las claves criptográficas. Es permisible, sin embargo, que existan claves con diferentes periodos de vida, en función de su longitud, el uso que vayan a recibir y su criticidad.

En general, **cuanto más vaya a ser usada una clave, menor debería ser su periodo de vida**. Así, la clave utilizada en un enlace de alto ancho de banda (gigabits por segundo, por ejemplo) debería ser cambiada más a menudo que la de un módem.

4.7. Resumen


Simétrica
Asimétrico
Velocidad alta
Velocidad baja
Gestión de claves problemática
Buena gestión de claves
Sin firma digital
Con firma digital
Autenticación parcial
Autenticación total

Lo + recomendado

No dejes de leer…

PKI de código abierto

Poner en marcha una PKI es un proceso muy costoso y complejo. Sin embargo, existe una alternativa libre de código abierto, EJBCA, muy conocida y utilizada. Visita su sitio Web, revisa la estructura general del *software* y de sus partes y sobre todo, entiende todos los conceptos que se explican en el segundo enlace.

Accede a los documentos a través del aula virtual o desde las siguientes direcciones web:

http://www.ejbca.org/

http://www.ejbca.org/docs/concepts.html

No dejes de ver…

DNI electrónico

No te pierdas este interesante video realizado por el profesor Daniel Sánchez de la Universidad de Murcia, en el que analiza los principales aspectos del DNI electrónico español. Se estudian su funcionalidad, sus principales características físicas y sus usos más comunes.

Accede al video a través del aula virtual o desde la siguiente dirección web:

https://www.youtube.com/watch?v=PMv_-qkUuFo

Lección de la Intypedia sobre clave pública

En esta lección de la *Intypedia* (interesante iniciativa para la enseñanza y difusión de la criptografía, liderada por el profesor Jorge Ramió de la UPM) se presenta una introducción muy asequible a los principios de la clave pública. No dejes de verla como repaso, o para comprobar que realmente has entendido los principales conceptos estudiados a lo largo de este tema.

Accede al video a través del aula virtual o desde la siguiente dirección web:
https://www.youtube.com/watch?v=On1clzor4x4

Criptografía asimétrica con PGP

Una de las aplicaciones de encriptación asimétrica más populares de todos los tiempos es Pretty Good Privacy, y Gnu Privacy Guard compatible con OpenPGP es una implementación de uso común. En este video, aprenderás sobre ambos.

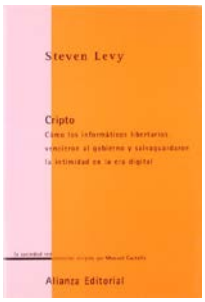
Accede al video a través del aula virtual o desde la siguiente dirección web:
https://www.youtube.com/watch?v=7_QnaPSH5bY

+ Información

A fondo

Cripto: Cómo los informáticos libertarios vencieron al gobierno y salvaguardaron la intimidad en la era digital

Levy, S. (2002). *Cripto: Cómo los informáticos libertarios vencieron al gobierno y salvaguardaron la intimidad en la era digital*. Edición Alianza. ISBN: 8420691089.

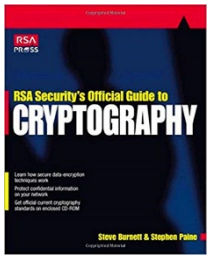


Este libro, escrito casi en forma de novela, cuenta la interesantísima historia del nacimiento de la criptografía de clave pública, que supuso una lucha sin precedentes por la libertad entre el gobierno (la NSA particularmente) y el mundo civil. Muchos de los ecos de esa lucha podemos encontrarlos hoy en día con los programas de espionaje masivos de la agencia de espionaje norteamericana.

RSA Security's Official Guide to Cryptography

Burnet, S., Paine, S. (2001). *RSA Security's Official Guide to Cryptography*. Edición McGraw-Hill. ISBN: 978-0072131390.

Escrito por personal de RSA Security, este recurso explica las diferencias entre la clave simétrica y la criptografía de clave pública, cómo PKI y X.509 afectan la seguridad y cómo funciona el algoritmo RSA dentro de los protocolos.



Enlaces relacionados

Fábrica Nacional de la Moneda y Timbre (FNMT)

En la página Web indicada más abajo, podrás encontrar el sitio oficial de la FNTM española, una de las CA más importantes del país. Navega por los distintos apartados y aprende sobre sus prácticas de certificación, los distintos certificados que emite y cómo es el proceso de obtención de los mismos.

Accede a la página web a través del aula virtual o desde la siguiente dirección:
https://www.sede.fnmt.gob.es/certificados

Recursos externos

OpenSSL

Kit criptográfica.

Accede a la página través del aula virtual o desde la siguiente dirección web:
http://www.openssl.org
Accede al manual a través del aula virtual o desde la siguiente dirección web:
http://www.iit.upcomillas.es/palacios/seguridad/openssl.pdf

Test

1. En la criptografía asimétrica o de clave pública, las claves pública y privada están relacionadas de forma que:

A. Sólo la clave privada puede utilizarse para cifrar mensajes
B. Lo que se cifra con una clave únicamente puede ser descifrado con la otra
C. Para enviar un mensaje, debemos usar la clave pública del receptor
D. Para descifrar un mensaje recibido, se debe utilizar nuestra propia clave privada

2. En su esquema más básico, para enviar un mensaje cifrado (no firmado) a un receptor llamado Bob, Alicia, el emisor, deberá:

A. Cifrar el mensaje con su propia clave pública, es decir, `E[Kpub(A), X]`
B. Cifrar el mensaje con su propia clave privada, es decir, `E[Kpriv(A), X]`
C. Cifrar el mensaje con la clave pública de Bob, es decir, `E[Kpub(B), X]`
D. Cifrar el mensaje con la clave privada de Bob, es decir, $E[K_{priv}(B), X]$

3. ¿Qué característica obtenemos cuando ciframos un mensaje con nuestra clave privada?

A. Autenticación, a través de la firma digital
B. Confidencialidad, a través del cifrado
C. Integridad
D. Confidencialidad e integridad

4. ¿Para qué sirve, en esencia, una función *hash*?

A. Para firmar digitalmente un mensaje, de forma que no pueda ser modificado
B. Para calcular un valor identificativo único de una cadena de entrada
C. Es un tipo de algoritmo de cifrado (se utiliza, por ejemplo, para guardar las contraseñas en las BBDD)
D. Genera un valor identificativo de longitud variable (128, 256 o 512 bits) en función de la longitud de la cadena de entrada

5. Una colisión en una función *hash* se produce cuando:

A. Dos textos de entrada distintos producen el mismo valor *hash*
B. Dos textos de entrada iguales producen el mismo valor *hash*
C. Dos textos de entrada distintos producen diferentes valores *hash*
D. Dos textos de entrada iguales producen diferentes valores *hash*

6. Cuando se genera un sobre digital, que proporciona confidencialidad y autenticación al mensaje, siempre se debe respetar el principio de:

A. Primero firmar, luego cifrar
B. Primero cifrar, luego firmar
C. Como se trata de una firma digital, el mensaje no se cifra
D. Las operaciones de cifrado y firma se realizan simultáneamente sobre el mismo texto de entrada

7. RSA es un algoritmo de…

A. Clave pública, basado en el problema de factorización de números
B. Clave secreta, basado en el problema de factorización de números
C. Firma digital, basado en el problema de factorización de números
D. Establecimiento de claves, basado en el problema de factorización de números

8. La longitud mínima recomendable de una clave pública, teniendo en cuenta los últimos avances en los algoritmos de factorización, sería de:

A. 768 bits
B. 1024 bits
C. 2048 bits
D. 4096 bits

9. Un certificado digital sirve esencialmente para:

A. Crear un vínculo criptográfico entre una clave privada y una identidad
B. Certificar que una persona es realmente quien dice ser, a través de una Autoridad de Certificación, o CA
C. Crear un vínculo criptográfico entre una clave pública y una identidad
D. Certificar que una persona o máquina (en general, una entidad) es realmente quien dice ser, a través de una Autoridad de Certificación, o CA

10. La duración de la vida de una clave criptográfica debería ser, en general:

A. Más corta cuanto mayor sea su uso
B. Más larga cuanto mayor sea su uso
C. Más corta cuanto menor sea su uso
D. Más larga cuanto menor sea su uso