Firmas digitales

Fernando Martínez

Departament de Matemàtiques • Universitat Politècnica de Catalunya

11 de noviembre de 2022

Índice

Firmas digitales

La criptografía de clave pública, además de confidenciabilidad, proporciona:

- Identificación.
- Integridad.
- No repudio.

Todos estos objetivos se consiguen utilizando firmas digitales, el equivalente de las firmas convencionales en el sentido de que son un añadido al final del mensaje conforme se está de acuerdo con lo que allí se dice.

Diferencias entre firma digital y convencional

- Una firma convencional está físicamente añadida al mensaje mediante el papel, la digital no. Si sólo consistiera en un simple añadido al final del mensaje sería muy fácil de manipular. La firma digital debe depender del firmante y del mensaje.
- Un documento convencional es fácil de distinguir de una copia, pero en el caso de documentos digitales es imposible. Hemos de tener cuidado de que un documento digital se utilize varias veces, por ejemplo para cobrar repetidamente la misma orden de pago.
- Para verificar la firma de un documento convencional es suficiente con comprobar que la firma coincida con la de un documento anterior. Para cada documento digital la firma es diferente por lo cual ha de haber un método para verificar cada firma y que además lo pueda utilizar cualquier persona.

Ejemplo: RSA

- $\mathrm{Sig}_k(m) = m^d \bmod n$,
- $\mathrm{Ver}_k(m,f) = \text{cierto} \Leftrightarrow m \equiv f^e \bmod n$.

– $\{e,n\}$ son públicos y d es privado.

– e y d pueden corresponder a los exponentes de cifrado aunque no es recomendable.

– Es conveniente que e sea pequeño ($e = 3, 17, 2^{16}+1$) porque un documento se firma una única vez pero se verifica muchas.

Firma electrónica en Europa

La Directiva 1999/93/CE fue la primera norma que regulaba los servicios de firma electrónica en la Unión Europea. Se actualizó con el Reglamento nº 910/2014 (Reglamento eIDAS). Ver FAQ En dichos documentos se define:

- **Firma electrónica** (firma electrónica simple): datos en forma electrónica anejos a otros datos electrónicos o asociados de manera lógica a ellos, utilizados como medio de autenticación. Ejemplos: garabato en una pantalla, *check box*,...
- **Firma electrónica avanzada:** firma electrónica que cumple
  - estar vinculada al firmante de manera única;
  - permitir la identificación del firmante;
  - haber sido creada utilizando medios que el firmante puede mantener bajo su exclusivo control;
  - estar vinculada a los datos a que se refiere de modo que cualquier cambio ulterior a los mismos sea detectable.
- **Firma electrónica cualificada:** firma electrónica avanzada que se crea mediante un dispositivo cualificado de creación de firmas electrónicas y que se basa en un certificado cualificado de firma electrónica.

Las firmas digitales que veremos se corresponderían con las definiciones legales de firma electrónica avanzada y firma electrónica cualificada.

Efectos jurídicos de las firmas electrónicas en Europa

Artículo 25 del Reglamento eIDAS:

- No se denegarán efectos jurídicos ni admisibilidad como prueba en procedimientos judiciales a una firma electrónica por el mero hecho de ser una firma electrónica o porque no cumpla los requisitos de la firma electrónica cualificada.
- Una firma electrónica cualificada tendrá un efecto jurídico equivalente al de una firma manuscrita.
- Una firma electrónica cualificada basada en un certificado cualificado emitido en un Estado miembro será reconocida como una firma electrónica cualificada en todos los demás Estados miembros.

Los tres tipos de firmas electrónicas tienen valor legal y son admisibles a juicio como pruebas legalmente válidas aunque varía la carga de la prueba:

- La autoría de la firma electrónica cualificada se presume como demostrada salvo que se aporten pruebas en contra por parte de quien sostenga lo contrario.
- En el caso de la firma electrónica simple o avanzada si se cuestiona la autoría de una firma corresponde demostrar la validez de la misma a quien sostenga que es auténtica.

Firmas: Observaciones

1. Los algoritmos asimétricos son muy lentos.
2. La firma tiene un tamaño igual o superior al mensaje.
3. En realidad se firma resumen o *hash* del mensaje.

Funciones Hash

- Función Hash: Función que asocia a una cadena de longitud arbitraria otra de longitud fija.
- Función Hash débilmente libre de colisiones (*second-preimage resistance*): Aquella que dado x es computacionalmente imposible hallar $x' \neq x$ tal que $h(x) = h(x')$.
- Función Hash fuertemente libre de colisiones (*collision resistance*): Aquella para la cual es computacionalmente imposible hallar x y x' tales que $h(x) = h(x')$.
- Función Hash unidireccional (*preimage resistance)*: Aquella que dado z es computacionalmente imposible hallar x tal que $h(x) = z$.

Toda función hash fuertemente libre de colisiones es unidireccional.

Funciones Hash

SHA 256 (224)

SHA 512 (384)

SHA1: NO usar en nuevas aplicaciones. Hay ataques más eficientes que los basados en la *paradoja del cumpleaños*.
Ya se han encontrado colisiones https://shattered.io/

MD5: NO usar en nuevas aplicaciones. Está completamente roto.
Ver por ejemplo:
http://www.mathstat.dal.ca/~selinger/md5collision

Arquitectura SHA1, SHA2

Merkle-Damgård hash function

- Preproceso:
  - Añadir *padding*.
  - Trocear el resultado en bloques, $M^{(1)}, M^{(2)}, \ldots, M^{(N)}$, de longitud fijada.
  - Inicializar los valores H_i del hash.
- Proceso: Procesar los bloques uno a uno:
  `Para` $t = 1...N$
  $(H_0, ..., H_k) = f(H_0, ..., H_k, M^{(t)})$
- Hash: $H_0 \| ... \| H_k$

Secure Hash Standard (SHS)

SHA-3 http://csrc.nist.gov/groups/ST/hash/sha-3/index.html

- 2/11/2007 el NIST hace un llamamiento público para desarrollar una nueva función hash criptográficamente segura.
- 31/10/2008: NIST recibe 64 propuestas,
- 10/12/2008: 51 son admitidas en la primera ronda.
- 24/07/2009: 14 pasan a la segunda ronda. Se abre un periodo de un año de exposición pública para presentar comentarios.
- 09/12/2010: NIST selecciona 5 finalistas: BLAKE, Grøstl, JH, Keccak, Skein
- El 2 de octubre de 2012 se anunció el ganador: **Keccak**.

Sha-3 standardization. *FIPS 202, SHA-3 Standard.*

SHA-3 Selection Announcement

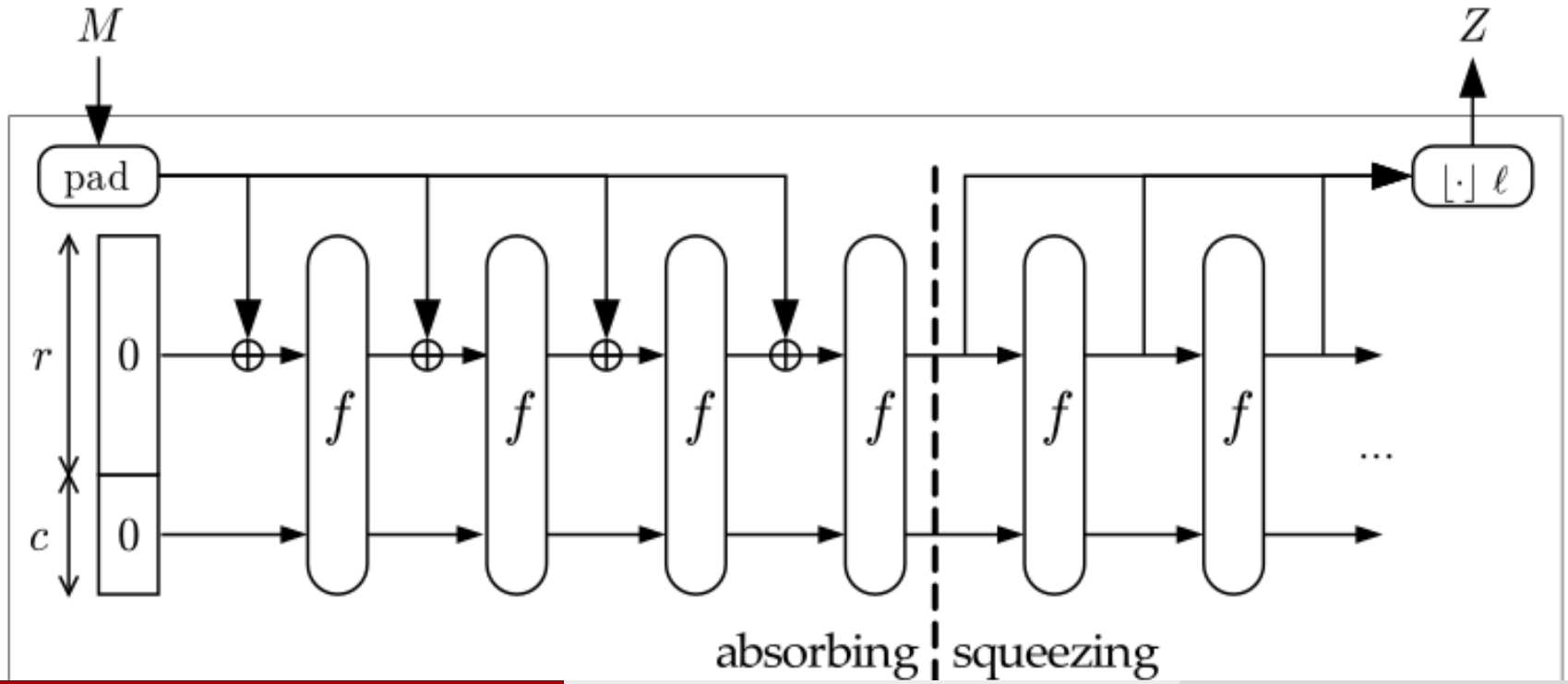
[...] NIST chose KECCAK [...] for its elegant design, large security margin, good general performance, excellent efficiency in hardware implementations, and for its flexibility. KECCAK uses a new ***sponge construction*** chaining mode, based on a fixed permutation, that can readily be adjusted to trade generic security strength for throughput, and can generate larger or smaller hash outputs as required. [...]

Additionally, KECCAK complements the existing SHA-2 family of hash algorithms well. NIST remains confident in the security of SHA-2 which is now widely implemented, and the SHA-2 hash algorithms will continue to be used for the foreseeable future, as indicated in the NIST hash policy statement. **One benefit that KECCAK offers as the SHA-3 winner is its difference in design and implementation properties from that of SHA-2.** It seems very unlikely that a single new cryptanalytic attack or approach could threaten both algorithms. Similarly, the very different implementation properties of the two algorithms will allow future application and protocol designers greater flexibility in finding one of the two hash algorithms that fits well with their requirements. [...]

The sponge construction

http://sponge.noekeon.org/

The sponge construction is a simple iterated construction for building a function F with variable-length input and arbitrary output length based on a fixed-length transformation f operating on a fixed number b of bits. b is called the width. The sponge construction operates on a state of $b = r + c$ bits. The value r is called the bitrate and the value c the capacity.



SHA-3 Family

- SHA3-224, SHA3-256, SHA3-384, SHA3-512: The SHA-3 hash function that produces 224, 256, 384, 512 bit digests.
  In each case, the capacity is double the digest length, i.e., $c = 2d$,

- SHAKE128, SHAKE256: The SHA-3 XOF *extendable-output function* that generally supports 128, 256 bits of security strength, if the output is sufficiently long ($d \geq 256/512$ minimun).
  The capacity is 256, 512.

Security strengths of SHA-1, SHA-2, SHA-3

	Size	Collision	Preimage	2nd Preimage
SHA-1	160	< 80	160	160-L(M)
SHA-224	224	112	224	min(224,256-L(M))
SHA-512/224	224	112	224	224
SHA-256	256	128	256	256-L(M)
SHA-512/256	256	128	256	256
SHA-384	384	192	384	384
SHA-512	512	256	512	512-L(M)
SHA3-224	224	112	224	224
SHA3-256	256	128	256	256
SHA3-384	384	192	384	384
SHA3-512	512	256	512	512
SHAKE128	d	min(d/2, 128)	$\geq$min(d, 128)	min(d, 128)
SHAKE256	d	min(d/2, 256)	$\geq$min(d, 256)	min(d, 256)

L(M)=$\lceil \log_2(len(M)/B) \rceil$, B is the block length of the function in bits, B=512 for SHA-1, SHA-224, SHA-256, and B=1024 for SHA-512.

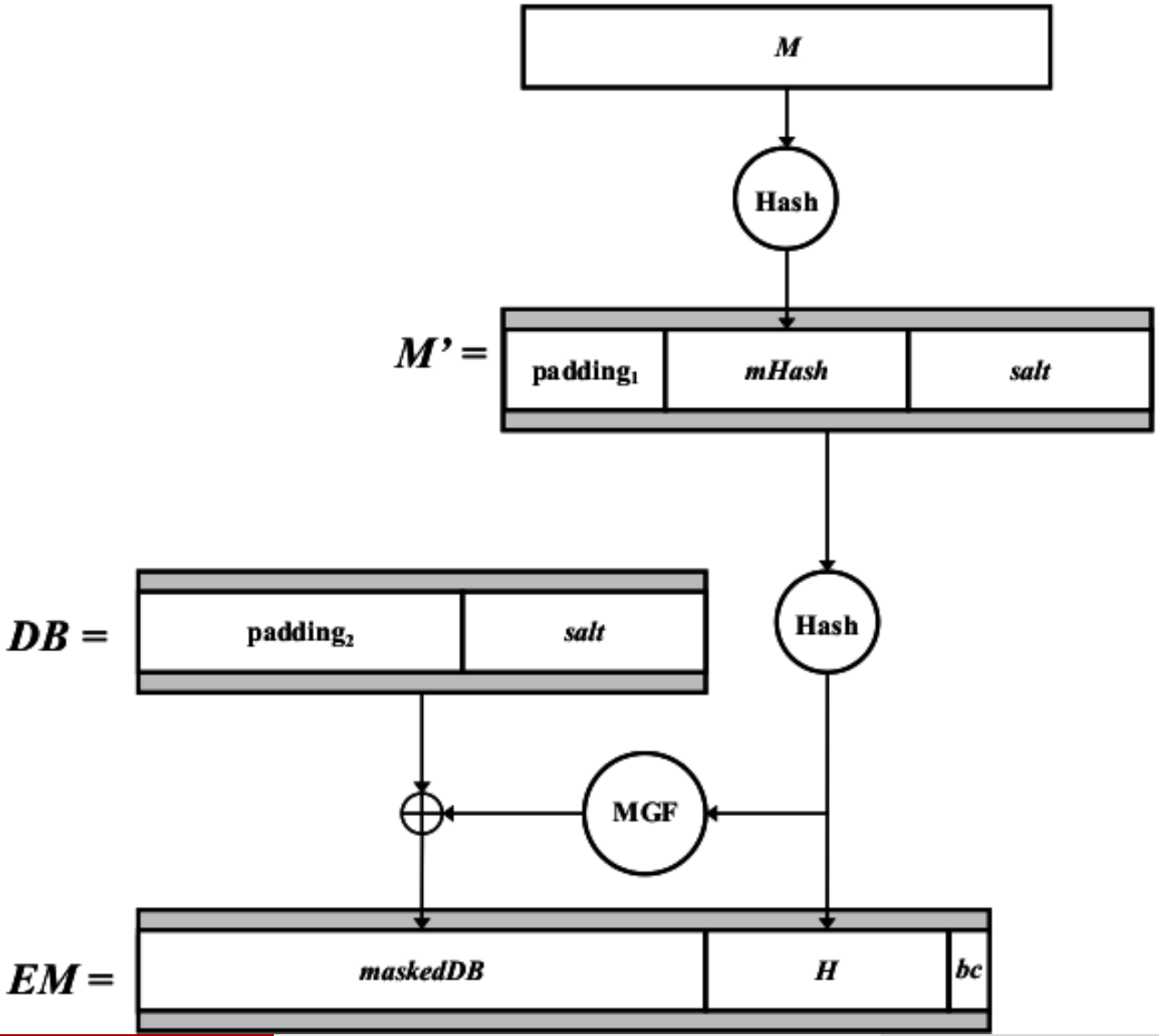
Digital Signature Standard (DSS)

https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

1. The RSA Digital Signature Algorithm
2. The Digital Signature Algorithm (DSA)
3. The Elliptic Curve Digital Signature Algorithm (ECDSA)

RSA-PSS

https://tools.ietf.org/html/rfc3447



DSA Digital Signature Algorithm: Generación de claves

Cada usuario realiza los siguientes pasos:

1. Elige q primo, $2^{N-1} < q < 2^N$ (N bits).
2. Elige p primo de L bits tal que $q|(p-1)$.
3. Elige $x \in \mathbb{Z}_p^*$ y calcula $g = x^{\frac{p-1}{q}} \bmod p$. Si $g = 1$ se toma otro x. g es el generador de un grupo cíclico.
4. Elige $r \in [2, q-1]$ aleatorio.
5. Calcula $u = g^r \bmod p$.
6. La clave pública es $\{p, q, g, u\}$, la clave privada es r.

L	~~1024~~	2048	2048	3072
N	~~160~~	224	256	256

DSA: Firma

Para firmar un mensaje m se realizan los siguientes pasos:

1. Se elige $k \in [2, q-1]$ aleatorio.
2. Se calcula $f_1 = (g^k \bmod p) \bmod q$.
3. Se calcula $k^{-1} \bmod q$.
4. Se calcula $f_2 = k^{-1}(SHA(m) + rf_1) \bmod q$. Si $f_2 = 0$ se toma otro k.
5. La firma es el par (f_1, f_2).

DSA: Verificación de la firma

Para verificar que la firma (f_1, f_2) del mensaje m es de A se realizan los pasos:

1. Se busca la clave pública de A, $\{p, q, g, u\}$.
2. Se calcula $w = f_2^{-1} \bmod q$.
3. Se calcula $\alpha_1 = SHA(m)w \bmod q$ y $\alpha_2 = f_1 w \bmod q$.
4. Se calcula $v = (g^{\alpha_1} u^{\alpha_2} \bmod p)$.
5. La firma es válida si $v \equiv f_1 \bmod q$.

Observación: La firma tiene longitud igual a $2N$ bits.

ECDSA Elliptic Curve Digital Signature Algorithm: Generación de claves

Cada usuario realiza los siguientes pasos:

1. Elige una curva elíptica sobre $\mathbb{Z}_p$, $E(\mathbb{Z}_p)$.
2. Elige $P \in E(\mathbb{Z}_p)$ de orden n grande.
3. Elige $d \in [2, n-1]$ aleatorio.
4. Calcula $Q = d\,P$.
5. La clave pública es $\{E(\mathbb{Z}_p), P, n, Q\}$, la clave privada es d.

ECDSS: Firma

Para firmar un mensaje m se realizan los siguientes pasos:

1. Elige $k \in [2, n-1]$ aleatorio.*
2. Calcula $kP = (x_1, y_1)$ y $f_1 = x_1 \bmod n$. Si $f_1 = 0$ se elige otro k.**
3. Calcula $k^{-1} \bmod n$.
4. Calcula $f_2 = k^{-1}(SHA(m) + f_1 d) \bmod n$. Si $f_2 = 0$ se toma otro k.
5. La firma es el par (f_1, f_2).

---

*Sony PS3 Security Broken. PS3 Epic Fail
https://events.ccc.de/congress/2010/Fahrplan/events/4087.en.html
https://events.ccc.de/congress/2010/Fahrplan/attachments/1780_27c3_console_hacking_2010.pdf

**https://arstechnica.com/information-technology/2022/04/major-crypto-blunder-in-java-enables-psychic-paper-forgeries

ECDSA: Verificación de la firma

Para verificar que la firma (f_1, f_2) del mensaje m es de A se realizan los siguientes pasos:

1. Se busca la clave pública de A, $\{E(\mathbb{Z}_p), P, n, Q\}$.
2. Calcula $w = f_2^{-1} \bmod n$.
3. Calcula $u_1 = SHA(m)\ w \bmod n$.
4. Calcula $u_2 = f_1\ w \bmod n$.
5. Calcula $u_1 P + u_2 Q = (x_0, y_0)$ y $v = x_0 \bmod n$.
6. La firma es válida si $v = f_1$.

ECDSA: $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$

Curva pseudo aleatoria $y^2 = x^3 - 3x + b \bmod p$:

- primo p=1157920892103562487626974469494075735...
  300861434152903141955336313088670978539 51,
- orden r=1157920892103562487626974469494075735...
  299969552241357603424222590610685120443 69,
- cofactor $h = 1$,
- semilla de 160 bits usada para generar el coeficiente b,
  semilla=c49d360886e704936a6678e1139d26b7819f7e90,
- parámetro c del algoritmo usado para generar el coeficiente b,
  c=7efba1662985be9403cb055c75d4f7e0ce8d84a9c5114abcaf3177680104fa0d,
- coeficiente b que satisface $b^2c \equiv -27 \bmod p$,
  b=5ac635d8aa3a93e7b3ebbd55769886bc651d06b0cc53b0f63bce3c3e27d2604b,
- punto $P = (P_x, P_y)$ de orden $n = r/h$
  P_x=6b17d1f2e12c4247f8bce6e563a440f277037d812deb33a0f4a13945d898c296,
  P_y=4fe342e2fe1a7f9b8ee7eb4a7c0f9e162bce33576b315ececbb6406837bf51f5.

p y r están en decimal, el resto en hexadecimal.

ECDSA: Generación de una curva aleatoria

Input p, primo impar.

Output cadena binaria *semilla* de 160 o más bits, $a,\ b \in \mathbb{Z}_p$ que definen una curva elíptica sobre $\mathbb{Z}_p$.

1. Sean $t = \lceil \log_2 p \rceil$, $s = \lfloor (t-1)/160 \rfloor$, $v = t - 160s$.
2. Elegir *semilla* aleatorio de longitud $g \geq 160$.
3. Calcular $H = \mathrm{SHA}(\textit{semilla})$ y sea c_0 la cadena de los v bits menos significativos de H (*v rightmost bits*).
4. Sea W_0 la cadena de v bits obtenidos haciendo 0 el bit más significativo (*leftmost*) de c_0. (Así nos aseguramos que $c < p$.)
5. Sea z el entero cuya expesión en binario es *semilla*.
6. Para i desde 1 hasta s:
   1. Sea s_i la expresión binaria del entero $(z+i) \bmod 2^g$.
   2. Calcular $W_i = \mathrm{SHA}(s_i)$.
7. Sea $W = W_0 \| W_1 \| \cdots \| W_s$.
8. Sea c entero que tiene por representación binaria W.
9. Si $c = 0$ o $4c + 27 \equiv 0 \bmod p$ volver a 2.
10. Elegir $a,\ b \in \mathbb{Z}_p^*$, tales que $b^2 c \equiv a^3 \bmod p$.
11. `Salida:` *semilla*, a y b.

ECDSA: Verificación de una curva

Dada una curva elíptica es conveniente comprobar que los parámetros son correctos.

1. Verificar que p es un primo impar.
2. Verificar que P no es el punto del infinito $\mathcal{O}$.
3. Verificar que a, b, P_x y P_y son enteros en el intervalo $[0, p-1]$.
4. Según el caso, verificar que la curva se ha obtenido aleatoriamente.
5. Verificar que a y b definen una curva elíptica sobre $\mathbb{Z}_p$, $4a^3 + 27b^2 \not\equiv 0 \bmod p$.
6. Verificar que P pertenece a la curva, $P_y^2 \equiv P_x^3 + a\,P_x + b \bmod p$
7. Verificar que $r = n\,h$.
8. Verificar que n es primo.
9. Verificar $n\,P = \mathcal{O}$.
10. Verificar que $n > 2^{160}$ y $n > 4\sqrt{p}$.
11. Comprobar que $h = \lfloor (\sqrt{p}+1)^2/n \rfloor$.
12. Verificar que n no divide $p^k - 1$, $1 \leq k \leq 20$.
13. Verificar que $n \neq p$.

9, 10 y 11 permiten comprobar que el número de puntos de la curva es el correcto.

ECDSA: Verificación de una clave ECDSA

Dada una clave ECDSA (un punto Q de la curva) es conveniente comprobar es correcta.

1. Verificar que Q no es el punto del infinito $\mathcal{O}$.
2. Verificar que Q_x y Q_y son enteros en el intervalo $[0, p-1]$.
3. Verificar que Q pertenece a la curva, $Q_y^2 \equiv Q_x^3 + a\,Q_x + b \bmod p$
4. Verificar $n\,Q = \mathcal{O}$.

Reducción módulos especiales***, p.e. $p = 2^{192} - 2^{64} - 1$

Si $x < p^2$, $x = x_5 2^{320} + x_4 2^{256} + x_3 2^{192} + x_2 2^{128} + x_1 2^{64} + x_0$, siendo $x_i < 2^{64}$.
Teniendo en cuenta

$$\begin{aligned} 2^{192} &\equiv 2^{64} + 1 \quad \text{mód } p \\ 2^{256} &\equiv 2^{128} + 2^{64} \quad \text{mód } p \\ 2^{320} &\equiv 2^{192} + 2^{128} \quad \text{mód } p \end{aligned}$$

tenemos

$$\begin{aligned} x &\equiv x_4 2^{256} + (x_5 + x_3)2^{192} + (x_5 + x_2)2^{128} + x_1 2^{64} + x_0 \quad \text{mód } p \\ x &\equiv (x_5 + x_3)2^{192} + (x_5 + x_4 + x_2)2^{128} + (x_4 + x_1)2^{64} + x_0 \quad \text{mód } p \\ x &\equiv (x_5 + x_4 + x_2)2^{128} + (x_5 + x_4 + x_3 + x_1)2^{64} + x_5 + x_3 + x_0 \quad \text{mód } p \end{aligned}$$

Si fuera necesario,

$\underline{x \equiv p - (x_5 + x_4 + x_2)2^{128} + (x_5 + x_4 + x_3 + x_1)2^{64} + x_5 + x_3 + x_0} \quad \text{mód } p$

*** Ver FIPS 186-4

Certificado digital

Certificado digital: documento electrónico que asocia una clave pública con su propietario.

- OpenPGP Message Format
- Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile

Aplicaciones de los certificados/firmas

- TLS (https, imaps, etc.)
- Correo electrónico
- Applets Java
- Atentificación de código
- Comercio electrónico.
- Secure boot.
- ...

Autoridades certificadoras (CA): Obtención de un certificado

Para la obtención de un certificado digital se siguen los siguientes pasos:

1. El usuario genera un par de claves pública-privada.
2. El usuario envía la clave pública a una autoridad certificadora CA. $\longrightarrow$ RA (autoridad de registro)
3. La CA comprueba la identidad del usuario. $\longrightarrow$ RA
4. La CA genera y firma un certificado para el usuario.
5. La CA envía el certificado al usuario.

Otros servicios de una CA

- Renovación de certificados.
- Búsqueda de certificados.
- Suspensión y revocación de certificados.
- Información sobre el estado del certificado.

Ejemplo X509v3: tbsCertificate

```
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 109 (0x6d)
    Signature Algorithm: ecdsa-with-SHA256
    Issuer: C = ES, ST = CATALUNYA, L = Barcelona, O = FIB, OU = MAT, CN = Criptografia FIB, email
    Validity
      Not Before: Nov 15 09:16:58 2021 GMT
      Not After : Jan 30 23:59:59 2022 GMT
    Subject: CN = Calvin, emailAddress = xxx.yyyr@estudiantat.upc.edu
    Subject Public Key Info:
      Public Key Algorithm: id-ecPublicKey
        Public-Key: (521 bit)
        pub:
          04:
          00:ea:af:d1:09:13:fd:de:3f:ab:8d:de:25:df:20:28:96:95:e0:3f:75:c4:0c:3f:62:28:10:28:cc:2
          d7:fe:a1:9a:3d:24:83:4c:ea:89:5a:c6:19:b5:db:0a:fd:e8:95:d5:26:43:8a:0a:88:ed:a2:e1:3e:c
          77:f4:10:dc:c1:63:00:1b:b8:e7:0f:7a:ec:b9:4d:b9:d0:d2:80:24:5c:98:fb:d3:c1:45:c4:73:80:1
          66:25:aa:60:7a:f2:47:24:7d:85:37:8b:07:75:86:22:74:94:a9:57:87:0c:0e:ca:31:5f:d2:87:97:b
          90:81:f3:76:bf:cd:fb:b7:75:2b:3c:cd
        ASN1 OID: secp521r1
        NIST CURVE: P-521
    X509v3 extensions:
      X509v3 Subject Alternative Name:
        email:xxx.yyyr@estudiantat.upc.edu
      X509v3 Basic Constraints: critical
        CA:FALSE
      X509v3 Authority Key Identifier:
        keyid:DC:E5:F0:C2:E2:87:E7:7B:EB:B0:D8:B4:65:69:FF:44:5F:E7:77:F2
     X509v3 Extended Key Usage:
      E-mail Protection
```

Ejemplo X509v3: SignatureValue

```
Signature Algorithm: ecdsa-with-SHA256
  30:
    81:87:
       02:
         41:
           41:94:db:d9:25:4b:10:3c:89:58:37:63:49:6f:6e:f0:
           36:bf:b5:04:d2:0b:e8:d1:a6:ab:13:b4:fe:3c:b3:de:
           17:81:2f:7c:cb:0b:7c:63:9e:48:0f:22:e3:18:fa:e7:
           b0:68:f0:cc:cc:0e:80:cd:f1:bd:10:cd:3f:a3:1c:e2:
           da:
       02:
         42:
           01:03:7b:47:cd:49:5c:89:12:d6:fa:7a:98:19:3b:ec:
           b6:5a:d8:5a:80:d8:ce:6f:fc:3f:a6:b3:a8:43:f6:27:
           ea:b9:5c:d3:ae:fb:da:63:b3:2a:d6:6e:34:63:90:71:
           82:42:1e:7e:a5:3e:90:d6:07:98:e9:df:e8:8e:af:e3:
           d7:02
```

X.509: Extensiones

- Authority Key Identifier
- Subject Key Identifier
- Key Usage: digitalSignature (0), nonRepudiation (1), keyEncipherment (2), dataEncipherment (3), keyAgreement (4), keyCertSign (5), cRLSign (6), encipherOnly (7), decipherOnly (8)
- Certificate Policies
- Basic Constraints
- CRL Distribution Points
- ...

Si una extensión es *crítica* y no es reconocida por la aplicación ha de darse un error. Si es *no-crítica* y no es reconocida por la aplicación puede ignorarse.

Certificate Revocation List

- La *CA* publica periódicamente listas firmadas de certificados emitidos por la propia *CA* que han sido revocados (CRL).
- En algunos casos, la *CA* puede delegar la publicación de CRL a una tercera parte de confianza (indirect CRL).
- Las CRL tienen un ámbito de aplicación. Por ejemplo, certificados emitidos por la *CA X* para empleados de la empresa *Y* destinados en *Z*.
- Para comprobar la validez de un certificado es necesario obtener la última CRL, verificar la firma de la *CA* y, si es válida, comprobar si el certificado está incluido o no en la lista.

Ejemplo CRL

```
Certificate Revocation List (CRL):
     Version 2 (0x1)
     Signature Algorithm: sha1WithRSAEncryption
     Issuer: /C=ES/O=Agencia Catalana de Certificacio (NIF Q-0801176-I)/L=Passatge de la Concepcio
     Last Update: Nov 17 03:59:31 2005 GMT
     Next Update: Nov 24 03:59:31 2005 GMT
     CRL extensions:
         X509v3 Authority Key Identifier:
         keyid:CD:92:C0:45:46:34:76:0D:D2:F4:5B:A2:74:1D:AB:CF:6C:B6:0B:B9
         DirName:/C=ES/O=Agencia Catalana de Certificacio (NIF Q-0801176-I)/OU=Serveis Publics de
         serial:70:40:40:D0:FD:CA:3C:19:3F:A2:3C:27:61:EA:70:CE
         X509v3 CRL Number: 1604
Revoked Certificates:
    Serial Number: 01
        Revocation Date: Nov 21 16:33:47 2003 GMT
    Serial Number: 15
        Revocation Date: May  7 08:37:14 2004 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code: Key Compromise
    Serial Number: 1A
        Revocation Date: May 27 09:48:11 2004 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code: Unspecified
    Serial Number: 01D3
        Revocation Date: Jan  7 12:25:08 2005 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code: Certificate Hold
            Hold Instruction Code:  Hold Instruction Reject
Signature Algorithm: sha1WithRSAEncryption
  a0:26:2f:38:01:65:64:ad:3f:14:92:28:9f:81:3e:06:e9:c6:b5:1f:71:f2:86:cb:f1:25:2a:59:5a:2f:23:49:
  b5:33:f5:93:0f:9d:11:58:15:94:c2:3c:fd:3f:eb:ef:d4:2a:db:e5:0e:2c:6f:2d:2a:58:35:6e:b7:f9:2f:7f:
```

Certificate Revocation List

Ventajas

- Simplicidad.

Inconvenientes

- Tamaño,
- entre publicación y publicación de la CRL se pueden aceptar como válidos certificados que están revocados (ventana de vulnerabilidad grande),
- coste de distribución alto,
- cada vez que se quiere comprobar un certificado es necesario consultar si se ha emitido una nueva CRL,
- ¿cómo se revoca el certificado de una *CA*?

Se está abandonando su uso.

Online Certificate Status Protocol (OCSP)

https://datatracker.ietf.org/doc/html/rfc6960

- Protocolo para determinar el estado de un certificado en cada momento.
- Se envía una petición para saber el estado de un certificado y se recibe una respuesta firmada por:
  - La *CA* que ha emitido el certificado,
  - una tercera parte de confianza del cliente o
  - un *Authorized Responder (AR)* que tiene un certificado especial emitido por la *CA* para firmar las respuesta OCSP.

Online Certificate Status Protocol (OCSP)

- La respuesta puede ser:
  - *good*: el certificado no ha sido revocado pero no implica que haya sido emitido o que sea válido en el tiempo en el momento de generar la respuesta,
  - *revoked*: el certificado ha sido revocado,
  - *unknown*: no se tiene información disponible sobre el certificado.
- La respuesta puede contener tres fechas:
  - thisUpdate: instante en el cual el estatus es conocido que es correcto.
  - nextUpdate: instante máximo en el que la información sobre el estatus será renovada
  - producedAt: instante en el que se firma la respuesta.

No es conveniente tener pre-calculadas las respuestas porque un atacante podría reenviar una respuesta *válida* aunque el certificado esté revocado.

Ejemplo OCSPRequest

```
OCSP Request Data:
  Version: 1 (0x0)
  Requestor List:
    Certificate ID:
      Hash Algorithm: sha1
      Issuer Name Hash: 1175295285B7738D52A8E3508FB390C5EEC7D46A
      Issuer Key Hash: 67FD8820142798C709D22519BBE9511163755062
    Serial Number: 08476C88FD62CE503FAAC820D9467380
  Request Extensions:
    OCSP Nonce:
    041011DBE69D6C294A05DFA717B611E5C1A7
```

Ejemplo OCSPResponse

```
OCSP Response Data:
  OCSP Response Status: successful (0x0)
  Response Type: Basic OCSP Response
  Version: 1 (0x0)
  Responder Id: 67FD8820142798C709D22519BBE9511163755062
  Produced At: Nov 17 18:36:34 2021 GMT  <---  Petición hecha Nov 18 14:20 --------
  Responses:
    Certificate ID:
    Hash Algorithm: sha1
    Issuer Name Hash: 1175295285B7738D52A8E3508FB390C5EEC7D46A
    Issuer Key Hash: 67FD8820142798C709D22519BBE9511163755062
    Serial Number: 08476C88FD62CE503FAAC820D9467380
  Cert Status: good
  This Update: Nov 17 18:21:01 2021 GMT
  Next Update: Nov 24 17:36:01 2021 GMT

  Signature Algorithm: sha256WithRSAEncryption
    0f:8a:96:1a:4f:9d:87:4a:0b:a4:fb:47:60:cb:fc:15:13:53:
    23:e5:90:09:7c:84:f7:b5:84:6d:0f:14:c2:35:0a:c2:49:ad:
    56:92:3f:af:b7:2b:d9:1b:77:3c:07:da:f4:2c:a9:1a:f6:b0:
    ...
    13:5d:5b:12:94:76:a9:e2:86:27:e3:1a:d8:88:54:d5:08:65:
    de:87:b8:07
WARNING: no nonce in response
Response verify OK
www.fib.upc.edu.crt: good
    This Update: Nov 17 18:21:01 2021 GMT <---------------
    Next Update: Nov 24 17:36:01 2021 GMT <---------------
```

OCSP

Ventajas

- No es necerario procesar gran cantidad de información.
- La ventana de vulnerabilidad puede reducirse.
- Con un único servidor OCSP puede ser suficiente para comprobar el estatus de certificados emitidos por diferentes *CA*.

Inconvenientes

- Es necesario confiar en una nueva entidad.
- La respuesta puede ser confusa.
- If you reject certificates when you can't get an answer, that's called *hard-fail*. If you accept certificates when you can't get an answer that's called *soft-fail*.

Algunos incidentes

- *Microsoft Security Bulletin MS01-017 (22/3/2001):* In mid-March 2001, VeriSign, Inc., advised Microsoft that on January 29 and 30, 2001, it issued two VeriSign Class 3 code-signing digital certificates to an individual who fraudulently claimed to be a Microsoft employee.

- *Comodo SSL Affiliate The Recent RA Compromise (23/3/2011)* On March 15th 2011, a Comodo affiliate RA was compromised resulting in the fraudulent issue of 9 SSL certificates to sites in 7 domains. Although the compromise was detected within hours and the certificates revoked immediately, the attack and the suspected motivation require urgent attention of the entire security field. 9 certificates were issued:
  - NOT seen live on the internet: mail.google.com, www.google.com, login.yahoo.com (2), login.skype.com, addons.mozilla.org, login.live.com, global trustee.
  - Seen live on the internet: login.yahoo.com,

The incident report

- *DigiNotar reports security incident (30/8/2011)* On July 19th 2011, DigiNotar detected an intrusion into its Certificate Authority (CA) infrastructure, which resulted in the fraudulent issuance of public key certificate requests for a number of domains, including Google.com.

  An update on attempted man-in-the-middle attacks

- *EFF to Verizon: Etisalat Certificate Authority Threatens Web Security (13/8/2010)* We are writing to request that Verizon investigate the security and privacy implications of the SSL CA certificate (serial number 0x40003f1) that Cybertrust (now a division of Verizon) issued to Etisalat on the 19th of December, 2005, and evaluate whether this certificate should be revoked.[...]
  These events clearly demonstrate that Etisalat and the UAE regulatory environment within which it operates are institutionally hostile to the existence and use of secure cryptosystems. It is therefore of great concern to us that Etisalat is in possession of a trusted SSL CA certificate and the accompanying private key, which effectively functions as a master key for the encrypted portion of the World Wide Web. Etisalat could use this key to issue itself valid HTTPS certificates for verizon.com, eff.org, google.com, microsoft.com, or indeed any other website. Etisalat could use those certificates to conduct virtually undetectable surveillance and attacks against those sites. Etisalat's keys could also possibly be used to obtain access to some corporate VPNs.

- *Win32/Stuxnet Signed Binaries (19/7/2010)* On July 17th, ESET identified a new malicious file related to the Win32/Stuxnet worm. This new driver is a significant discovery because the file was signed with a certificate from a company called `JMicron Technology Corp`. This is different from the previous drivers which were signed with the certificate from `Realtek Semiconductor Corp`. It is interesting to note that both companies whose code signing certificates were used have offices in Hsinchu Science Park, Taiwan.
- *Adobe Reader zero-day attack - now with stolen certificate (8/9/2010)* While most malicious PDFs download their payload, this time the PDF has malicious content embedded. The PDF drops an executable into the `temp` directory and tries to execute it.
  The file it drops is digitally signed with a valid signature from a `US-based Credit Union`!

- *Sony attackers also stole certificates to sign malware (9/12/2014)*
  Security firm Kaspersky Labs reports that a new sample of the Destover malware–the malware family used in the recent attack on the networks of Sony Pictures–has been found bearing a valid digital signature that could help it sneak past security screening on some Windows systems. And that digital signature is courtesy of a certificate stolen from Sony Pictures.

- *How a 2011 Hack You've Never Heard of Changed the Internet's Infrastructure.* It all started with an internet user in Iran who couldn't get into his Gmail account.

Chrome's Plan to Distrust Symantec Certificates

At the end of July 2017, the Chrome team and the PKI community converged upon a plan to reduce, and ultimately remove, trust in Symantec's infrastructure in order to uphold users security and privacy when browsing the web. This plan, arrived at after significant debate on the blink-dev forum, would allow reasonable time for a transition to new, independently-operated Managed Partner Infrastructure while Symantec modernizes and redesigns its infrastructure to adhere to industry standards. This post reiterates this plan and includes a timeline detailing when site operators may need to obtain new certificates.

Algunas soluciones

- OCSP stapling fixes certain performance and privacy issues with OCSP revocation checking.
- *Certificate Transparency* makes it possible to detect SSL certificates that have been mistakenly issued by a certificate authority or maliciously acquired from an otherwise unimpeachable certificate authority. It also makes it possible to identify certificate authorities that have gone rogue and are maliciously issuing certificates.
  `https://crt.sh/`

Firmas basadas en funciones hash (*hash-based signatures*)

No están basadas en los mismos principios que la criptografía de clave pública sino que hacen uso de funciones *hash* para realizar la firma en sí.

Su principal inconveniente es que claves y firmas son muy grandes comparadas con las de la criptografía de clave pública.

Su ventaja, de cara al futuro, es su *resistencia* a los ordenadores cuánticos.

Lamport One Time Signature (LOTS) (I)

Una clave sólo se puede usar para una firma.

- **Clave privada**

$$(x_1, y_1), \ldots, (x_k, y_k)$$

siendo k el número de bits de la salida de la función HASH usada.

- **Clave pública**

$$(Pub_{x_1}, Pub_{y_1}), \ldots, (Pub_{x_k}, Pub_{y_k})$$

siendo $Pub_{x_i} = HASH(x_i)$, $Pub_{y_i} = HASH(y_i)$.

Si se usa SHA256, $k = 256$ y el tamaño de la clave privada y de la pública es 16Kbytes (ya que x_i e y_i tienen 256 bits cada uno, cómo mínimo, y necesitamos 256 pares.

Lamport One Time Signature (LOTS) (II)

- **Firma** del documento M.
  Sea $h = HASH(M)$, $h = h_1 h_2 \dots h_k$, siendo h_i los bits de h.
  Definimos:

$$S_i = \begin{cases} x_i & \text{si } h_i = 0, \\ y_i & \text{si } h_i = 1, \end{cases}$$

  La firma del documento M es $S = S_1, S_2, \dots, S_k$

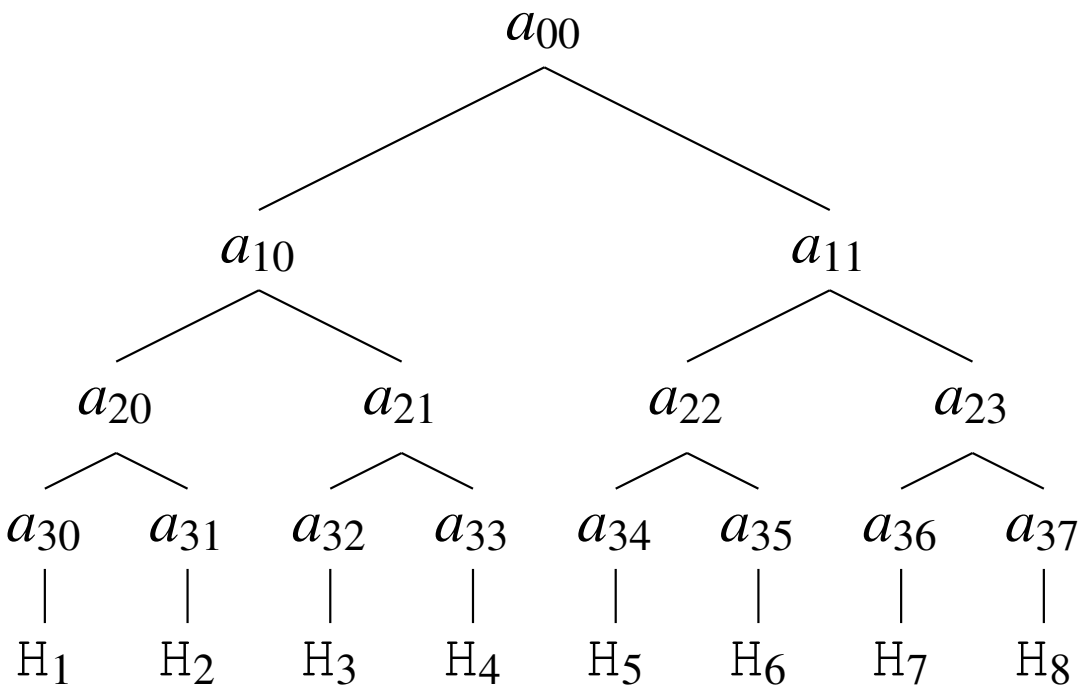
- **Verificación** de la firma $S = S_1, S_2, \dots, S_k$ del documento M con la clave $(Pub_{x_1}, Pub_{y_1}), \dots, (Pub_{x_k}, Pub_{y_k})$.
  Sea $h = HASH(M) = h_1 h_2 \dots h_k$

$$\begin{cases} \text{si } h_i = 0 \text{ comprueba que } HASH(S_i) = Pub_{x_i} \\ \text{si } h_i = 1 \text{ comprueba que } HASH(S_i) = Pub_{y_i}, \end{cases}$$

Merkle Signature Scheme (I)

Generación de claves (Una clave permite firmar $N = 2^k$ mensajes.)

- Se generan N pares de claves LOTS pública-privada $(Pub_i, Priv_i)$.
- Se calcula $\mathtt{H}_i = HASH(Pub_i)$
- *hash-tree*: Con $\mathtt{H}_i$ como hojas cada padre es el hash de sus hijos.



La **clave pública** es la raíz a_{00} del *hash-tree* (256 bits si se usa SHA256).

La **clave privada** son los pares $(Pub_i, Priv_i), i = 1, \ldots, N$

Merkle Signature Scheme (II)

Firma: Para firmar M elegimos el primer par $(Pub_i, Priv_i)$ no usado y lo firmamos con Lamport One Time Signature, siendo el resultado $\widetilde{Sig}$.

Para demostrar que la clave usada forma parte de la clave pública se revela Pub_i y los nodos necesarios del *hash-tree* que permiten recalcular la raíz. Por ejemplo, si $(Pub_5, Priv_5)$ es la clave usada para firmar publicamos $a_{10}, Pub_5, \mathtt{H}_6, a_{23}$; con Pub_5 calculamos $\mathtt{H}_5$, con $\mathtt{H}_5, \mathtt{H}_6$ calculamos a_{22}; con a_{22}, a_{23} calculamos a_{11}; y con a_{10}, a_{11} calculamos la raíz a_{00}. (Hay que publicar k nodos –profundidad del árbol– y la Pub_i.)

La firma es $Sig = (\widetilde{Sig}, Pub_i, k \text{ nodos})$.

Verificación de la firma:

- Con $\widetilde{Sig}$ y Pub_i se verifica la LOTS.
- Con los k nodos y Pub_i se verifica que se ha usado la clave pública a_{00}.

Recommendation for Stateful Hash-Based Signature Schemes

Recommendation for Stateful Hash-Based Signature Schemes

David A. Cooper, Daniel C. Apon, Quynh H. Dang, Michael S. Davidson, Morris J. Dworkin, Carl A. Miller

Abstract: This recommendation specifies two algorithms that can be used to generate a digital signature, both of which are stateful hash-based signature schemes: the Leighton-Micali Signature (LMS) system and the eXtended Merkle Signature Scheme (XMSS), along with their multi-tree variants, the Hierarchical Signature System (HSS) and multi-tree XMSS (XMSSMT).

https://doi.org/10.6028/NIST.SP.800-208