
Criptografia de clau pública

PID_00235163

Cristina Pérez Solà
Jordi Herrera Joancomartí

Temps mínim de dedicació recomanat: 3 hores



**Cristina Pérez Solà**

Doctora en Informàtica per la Universitat Autònoma de Barcelona i la Universitat Catòlica de Lovaina. Actualment és professora dels Estudis d'Informàtica, Multimèdia i Telecomunicacions de la Universitat Oberta de Catalunya. Els seus àmbits de recerca són les criptomonedes basades en *blockchain* i, en especial, els aspectes relacionats amb la seguretat i la privadesa d'aquestes. També està interessada en els problemes de privacitat que sorgeixen arran de l'ús de les xarxes socials i en l'adaptació de tècniques de mineria de dades a la naturalesa específica d'aquest tipus de xarxes.

**Jordi Herrera Joancomartí**

Llicenciat en Matemàtiques per la Universitat Autònoma de Barcelona i doctor per la Universitat Politècnica de Catalunya. Els seus àmbits de recerca són la criptografia, les criptomonedes i la tecnologia *blockchain*. Ha publicat nombrosos textos docents i més de cent articles de recerca en revistes i congressos nacionals i internacionals. Ha dirigit nou tesis doctorals i ha estat investigador principal de diversos projectes de recerca nacionals. Ha participat com a avaluador per a agències de recerca de diversos països europeus i també per a la Comissió Europea. Actualment és professor agregat del departament d'Enginyeria de la Informació i les Comunicacions a la Universitat Autònoma de Barcelona.

L'encàrrec i la creació d'aquest recurs d'aprenentatge UOC han estat coordinats per la professora: Helena Rifà Pous

Segona edició: febrer 2021
© d'aquesta edició, FUOC, 2021
Av. Tibidabo, 39-43, 08035 Barcelona
Autoria: Jordi Herrera Joancomartí, Cristina Pérez Solà
Producció: FUOC
Tots els drets reservats

Cap part d'aquesta publicació, incloent-hi el disseny general i la coberta, no pot ser copiada, reproduïda, emmagatzemada o transmesa de cap manera ni per cap mitjà, tant si és elèctric com mecànic, òptic, de gravació, de fotocòpia o per altres mètodes, sense l'autorització prèvia per escrit del titular dels drets.

Índex

Introducció	5
Objectius	6
1. L'origen de la criptografia de clau pública	7
2. Intercanvi de claus de Diffie-Hellman	9
3. Xifres de clau pública	12
3.1. Xifratge basat en la factorització d'enters: RSA	12
3.2. Xifratge basat en el logaritme discret: ElGamal	15
4. Signatures digitals	17
4.1. Signatures basades en la factorització d'enters: RSA	18
4.2. Signatures basades en el logaritme discret: ElGamal	19
4.2.1. L'estàndard DSA	21
4.3. Atacs als esquemes de signatura digital	22
4.3.1. Falsificació existencial de signatures RSA	23
4.3.2. Falsificació existencial de signatures ElGamal	23
4.3.3. Vulnerabilitat en la reutilització de valors ElGamal ...	24
5. Criptografia simètrica i asimètrica	26
6. Implementació dels algorismes de clau pública	28
6.1. Optimització del xifrat RSA	28
6.2. Optimització del desxifrat RSA	29
6.3. Optimització del xifrat ElGamal	31
6.4. Optimització del desxifrat ElGamal	31
7. Criptografia postquàntica	33
Resum	34
Exercicis d'autoavaluació	35
Solucionari	37
Glossari	39
Bibliografia	40

Introducció

En aquest mòdul didàctic ens introduïrem en el món de la criptografia de clau pública. En primer lloc, descriurem el concepte de criptografia de clau pública o asimètrica, així com les característiques més destacades d'aquest tipus d'algorismes. També veurem dos dels algorismes de xifrat de clau pública més utilitzats avui en dia: l'RSA i l'ElGamal.

Seguidament presentarem el concepte de signatura digital i descriurem tres dels algorismes més populars de signatura digital: l'esquema de signatura RSA, el d'ElGamal, i el DSA.

A continuació, compararem els algorismes de clau pública presentats amb els algorismes de criptografia simètrica que havíem vist en mòduls anteriors, destacant-ne les seves fortaleses i debilitats. També descriurem alguns detalls que cal tenir en compte a l'hora d'implementar els algorismes de clau pública descrits.

Finalment, presentarem altres famílies de criptografia de clau pública que, a diferència de les presentades amb detall en aquest mòdul, no estan basades en els problemes de factorització d'enters ni en el càlcul del logaritme discret.

Objectius

Els objectius que es pretén que l'estudiant assoleixi amb aquest mòdul didàctic són els següents:

- 1.** Familiaritzar-se amb el concepte de criptografia de clau pública.
- 2.** Conèixer els criptosistemes de clau pública més populars.
- 3.** Aprendre el concepte de signatura digital i les propietats que ens ofereix.
- 4.** Conèixer els esquemes de signatura digital més populars.
- 5.** Veure algunes de les optimitzacions que es fan servir habitualment a l'hora d'implementar algorismes de criptografia de clau pública.
- 6.** Descobrir com es combina la criptografia de clau pública amb altres primitives criptogràfiques.

1. L'origen de la criptografia de clau pública

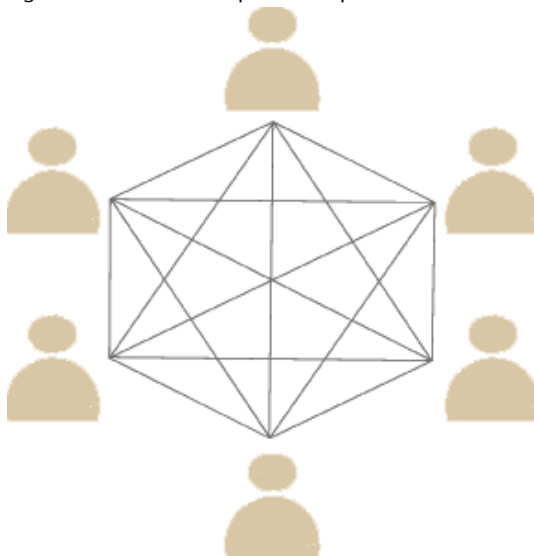
La criptografia de clau simètrica es caracteritza per fer servir una mateixa clau tant per a xifrar com per a desxifrar. És a dir, en criptografia simètrica, tant l'emissor d'un missatge (que el xifrarà abans d'enviar-lo) com el receptor (que l'haurà de desxifrar per poder-lo interpretar) comparteixen una única clau.

La criptografia de clau simètrica presenta algunes limitacions:

1) La **distribució de claus** s'ha de realitzar sobre un canal segur. Com que els dos usuaris comparteixen la mateixa clau i aquesta clau pot ser utilitzada directament per a xifrar i desxifrar missatges, la clau no pot ser transmesa per un canal insegur, ja que aleshores un atacant que estigués escoltant el canal podria capturar-la i utilitzar-la. Per tant, per a aconseguir que dos usuaris, l'Alice i en Bob, aconseguixin tenir una clau simètrica amb la qual comunicar-se, caldrà que aquests dos usuaris hagin tingut un canal segur pel qual transmetre-la amb anterioritat. Per exemple, caldrà que l'Alice i en Bob s'hagin trobat presencialment o bé que ja disposin d'un canal segur per on transmetre la clau.

2) La **gestió de claus** es complica quan el número d'usuaris creix. Si hi ha n usuaris i cada parell d'usuaris necessita compartir una clau, caldrà gestionar $n(n-1)/2$ claus, és a dir, el número de claus d'un sistema creix quadràticament amb el número d'usuaris d'aquest. Així doncs, apareixen problemes d'escalabilitat a l'hora de gestionar totes aquestes claus. La figura 1 mostra un exemple de sis usuaris que comparteixen claus 2 a 2; necessiten, per tant, l'existència de quinze claus.

Figura 1. Sis usuaris comparteixen quinze claus diferents.



3) No es disposa de la propietat de **no-repudi**. Diem que un criptosistema ofereix la propietat de no-repudi si l'autor d'un missatge no en podrà negar posteriorment l'autoria. En criptografia simètrica, com que hi ha més d'un usuari que comparteix una mateixa clau, no es pot garantir que un usuari en concret hagi realitzat una acció donada, ja que sempre hi haurà algun altre usuari que la podria haver realitzat.

La **criptografia de clau pública** o asimètrica permet superar aquestes limitacions. D'una banda, la criptografia de clau pública proporciona mètodes d'establiment de clau, és a dir, mètodes per aconseguir que dos usuaris que es comuniquen per un canal insegur puguin crear claus que els permetin comunicar-se de manera segura. D'altra banda, la criptografia de clau pública permet que un conjunt d'usuaris es comuniquin dos a dos de manera segura fent servir únicament un parell de claus per a cada usuari. Així, el nombre de claus d'un sistema de clau pública creix linealment amb el número d'usuaris d'aquest sistema (en contraposició al creixement quadràtic que presenten els esquemes de criptografia simètrica). Finalment, per mitjà de les signatures digitals, la criptografia de clau pública ens ofereix la propietat de no-repudi.

L'origen de la criptografia de clau pública és una mica discutit. L'article "New directions in cryptography" (1976), de Whitfield Diffie i Martin Hellman, va donar a conèixer la criptografia de clau pública a la comunitat científica i va suposar l'inici d'un canvi de paradigma en la seguretat de la informació. Posteriorment, però, amb la desclassificació de documents confidencials del govern Britànic l'any 1997, es va saber que James Ellis, Clifford Cocks i Graham Williamson havien descobert el principi de clau pública uns anys abans que Diffie i Hellman, però que el descobriment no s'havia fet públic perquè era confidencial. Tot i això, es creu que els criptògrafs del govern britànic no eren conscients en aquell moment del potencial del que havien descobert.

2. Intercanvi de claus de Diffie-Hellman

L'algorisme d'intercanvi de claus de Diffie-Hellman (DHKE, de l'anglès *Diffie-Hellman key exchange*) és un algorisme d'intercanvi de claus basat en el problema del logaritme discret. L'algorisme va ser proposat per Whitfield Diffie i Martin Hellman el 1976 i està inspirat en la feina de Ralph Merkle. Va ser el primer algorisme de criptografia asimètrica.

L'algorisme d'intercanvi de claus de Diffie-Hellman permet que dos usuaris que es comuniquen per un canal insegur puguin aconseguir derivar una clau compartida de manera segura. D'aquesta manera, encara que un atacant estigui escoltant el canal, no pot aconseguir conèixer la clau derivada pels usuaris. Tot i així, l'esquema no és segur davant d'atacants que puguin modificar la informació que viatja pel canal.

L'algorisme requereix d'un procés d'inicialització en què es trien dos valors, p i α , que es fan públics:

Ús del DHKE

L'algorisme d'intercanvi de claus de Diffie-Hellman es fa servir actualment a internet en els protocols TLS, SSH o IPSec.

Algorisme d'intercanvi de claus de Diffie-Hellman

L'algorisme d'intercanvi de claus de Diffie-Hellman entre dos usuaris A i B consta dels passos següents:

- 1) Es tria un nombre primer aleatori p i un enter $\alpha \in [2, \dots, p-2]$ primitiu. Es fan públics els valors p i α .
- 2) A tria un valor aleatori $a = k_{privA} \in [2, \dots, p-2]$ i calcula $k_{pubA} = \alpha^a \text{ mod } p$.
- 3) B tria un valor aleatori $b = k_{privB} \in [2, \dots, p-2]$ i calcula $k_{pubB} = \alpha^b \text{ mod } p$.
- 4) A i B intercanvien els seus valors k_{pub} , és a dir, A envia a B el valor k_{pubA} i B envia a A el valor k_{pubB} .
- 5) A deriva la clau compartida $k_{AB} = k_{pubB}^a \text{ mod } p$.
- 6) B deriva la clau compartida $k_{AB} = k_{pubA}^b \text{ mod } p$.

Així doncs, efectivament, el valor k_{AB} derivat per les dues parts participants en el protocol és el mateix, ja que, d'una banda, l'usuari A calcula:

$$k_{AB} = k_{pubB}^a \text{ mod } p = (\alpha^b)^a \text{ mod } p$$

D'altra banda, l'usuari B calcula:

$$k_{AB} = k_{pubA}^b \pmod p = (\alpha^a)^b \pmod p = (\alpha^b)^a \pmod p$$

L'esquema descrit fa servir un element α primitiu a \mathbb{Z}_p^* en què p és un primer. Aquesta descripció correspon a la implementació original de l'algorisme. Tot i això, l'algorisme pot ser generalitzat per a fer servir qualsevol grup cíclic finit G d'ordre n i un element α generador a G . En aquest cas, direm que el protocol es basa en el problema de Diffie-Hellman generalitzat.

Pel que fa a la seguretat davant d'un atacant que escolti el canal, l'atacant coneixerà els dos valors intercanviats pel canal, k_{pubA} i k_{pubB} , a més dels paràmetres públics p i α , però calcular k_{AB} a partir d'aquests valors és un procés computacionalment difícil.

Logaritme discret en els enters mòdul p

El 2005 es va aconseguir calcular el logaritme discret mòdul un primer fort de 431 bits. El 2007 es va anunciar el càlcul d'un logaritme discret mòdul un primer segur de 530 bits. El 2014 es va aconseguir per a un primer segur de 596 bits, i el 2016, per a un de 768 bits.

Exemple d'intercanvi de claus de Diffie-Hellman

Els usuaris A i B disposen d'un canal insegur amb el qual comunicar-se i volen aconseguir crear una clau compartida k_{AB} :

- 1) Se seleccionen els valors públics $p = 508107251$ i $\alpha = 5203822$.
- 2) A tria el valor aleatori $a = 385641303$ i calcula:
 $k_{pubA} = \alpha^a \pmod p = 5203822^{385641303} \pmod{508107251} = 421539355$.
- 3) B tria un valor aleatori $b = 467804164$ i calcula:
 $k_{pubB} = \alpha^b \pmod p = 5203822^{467804164} \pmod{508107251} = 230346411$.
- 4) A i B intercanvien els seus valors k_{pub} .
- 5) A deriva la clau compartida, calculant:
 $k_{AB} = (k_{pubB})^a \pmod p = 230346411^{385641303} \pmod{508107251} = 45453571$.
- 6) B deriva la clau compartida, calculant:
 $k_{AB} = (k_{pubA})^b \pmod p = 421539355^{467804164} \pmod{508107251} = 45453571$.

El protocol finalitza amb A i B compartint el mateix valor secret k_{AB} .

Com hem comentat anteriorment, el protocol d'intercanvi de claus de Diffie-Hellman no és segur davant d'atacants que puguin situar-se enmig de la comunicació entre els dos usuaris i puguin modificar la informació que viatja entre ells.

Exemple de MiM en l'intercanvi de claus de Diffie-Hellman

Suposem que els dos usuaris A i B de l'exemple anterior es comuniquen per un canal insegur. L'usuari M pot interceptar les seves comunicacions i modificar els missatges que viatgen pel canal. De nou, A i B intenten establir una clau compartida k_{AB} .

- 1) Els passos de l'1 al 3 es realitzen exactament igual que a l'exemple anterior. Per simplicitat, farem servir els mateixos valors. Així doncs, en finalitzar el pas 3, A haurà calculat $k_{pubA} = 421539355$ i B haurà calculat $k_{pubB} = 230346411$ ($p = 508107251$ i $\alpha = 5203822$).
- 2) A envia k_{pubA} pel canal que el comunica amb B . Alhora, B envia k_{pubB} pel mateix canal.
- 3) M intercepta els missatges k_{pubA} i k_{pubB} , i procedeix a fer el següent:
 - a) Triar un valor aleatori $ma = 361369039$ i calcular:
 $k_{pubMA} = \alpha^{ma} \pmod p = 5203822^{361369039} \pmod{508107251} = 176105595$

b) Triar un valor aleatori $mb = 504619741$ i calcular:

$$k_{pubMB} = \alpha^{mb} \pmod p = 5203822^{504619741} \pmod{508107251} = 342944530$$

c) Enviar el valor k_{pubMA} a A i el valor k_{pubMB} a B. Noteu que, a partir d'aquest moment, A espera rebre k_{pubB} però rebrà k_{pubMA} i B espera rebre k_{pubA} però rebrà k_{pubMB} .

4) A deriva la clau compartida, calculant:

$$k'_{AB} = (k_{pubMA})^a \pmod p = 176105595^{385641303} \pmod{508107251} = 36322887$$

A creu que el valor k'_{AB} és una clau compartida amb B, però en realitat correspon a una clau compartida amb M.

5) B deriva la clau compartida, calculant:

$$k''_{AB} = (k_{pubMB})^b \pmod p = 342944530^{467804164} \pmod{508107251} = 461525945$$

B creu que el valor k''_{AB} és una clau compartida amb A, però en realitat correspon a una clau compartida amb M. Noteu que els valors k'_{AB} i k''_{AB} difereixen.

6) M deriva les claus compartides amb A i B, calculant:

$$k_{MA} = (k_{pubA})^{ma} \pmod p = 421539355^{361369039} \pmod{508107251} = 36322887$$

$$k_{MB} = (k_{pubB})^{mb} \pmod p = 230346411^{504619741} \pmod{508107251} = 461525945$$

A partir d'aquest moment, M comparteix una clau amb A i una amb B, i és capaç de llegir i modificar tots els missatges que s'intercanvien pel canal.

Noteu que aquest atac és possible ja que no hi ha autenticació de les parts que participen en el protocol.

3. Xifres de clau pública

Habitualment els algorismes de xifrat de clau pública comprenen tres funcions bàsiques: la generació de claus, el xifrat i el desxifrat. L'algorisme de generació de claus retorna un parell de claus de criptografia asimètrica, $[k_{pub}, k_{priv}]$. L'algorisme de xifrat rep un missatge m i una clau pública k_{pub} i genera el missatge xifrat c , i l'algorisme de desxifrat rep un missatge xifrat c i una clau privada k_{priv} i permet recuperar el missatge original m .

Clau pública

Les funcions bàsiques d'un esquema de xifrat amb clau pública són:

$$[k_{pub}, k_{priv}] = \text{generació_claus}()$$
$$c = E(k_{pub}, m)$$
$$m = D(c, k_{priv})$$

3.1. Xifratge basat en la factorització d'enters: RSA

L'RSA és un criptosistema de clau pública basat en el problema de la factorització d'enters. Va ser el primer criptosistema de clau pública proposat, presentat el 1977, només un any més tard que el concepte de criptografia de clau pública es fes públic. RSA són les sigles formades a partir de les inicials dels cognoms dels seus creadors, Ron Rivest, Adi Shamir i Leonard Adleman. Avui en dia, l'RSA encara és un dels criptosistemes de clau pública més utilitzats, tot i que els criptosistemes basats en corbes el·líptiques cada vegada van guanyant més terreny.

L'RSA es fa servir, principalment, en dos contextos: per a xifrar dades de poca mida (normalment, per a xifrar claus criptogràfiques) i en signatures digitals. A l'apartat 5 veurem una de les construccions més utilitzades per a transmetre grans quantitats de dades fent servir l'RSA combinat amb un criptosistema de clau simètrica, obtenint així els beneficis dels dos criptosistemes.

Algorisme de generació de claus RSA

Els passos que cal seguir per tal de generar un parell de claus RSA són els següents:

- 1) Es trien dos primers aleatoris p i q .
- 2) Es calcula $n = p \cdot q$.
- 3) Es calcula $\phi(n) = (p-1)(q-1)$.
- 4) Se selecciona un exponent públic $e \in (1, \phi(n))$ tal que $\text{mcd}(e, \phi(n)) = 1$.
- 5) Es calcula l'exponent privat d tal que $d \cdot e = 1 \pmod{\phi(n)}$. És a dir, es calcula $d = e^{-1} \pmod{\phi(n)}$.
- 6) La clau pública k_{pub} és el parell (n, e) , mentre que la clau privada k_{priv} és el valor (d) . Els valors $\phi(n)$, p i q també són valors secrets que només coneix el propietari de la clau privada.

Funció $\phi(n)$

La funció totient d'Euler $\phi(n)$ (descrita al mòdul sobre fonaments matemàtics) compta el número d'enters positius més petits que n que són coprimers amb n .

Noteu que el valor d sempre existirà, ja que amb la condició $\text{mcd}(e, \phi(n)) = 1$ assegurem que e té invers mòdul $\phi(n)$.

Quan parlem de la longitud de la clau RSA, parlem de la mida del mòdul n , normalment expressada en bits.

Mida d' n

Amb data de febrer de 2016, l'Institut Nacional d'Estàndards i Tecnologia (NIST) recomanava fer servir claus d'almenys 2048 bits i augmentar la mida a 3072 bits per a algunes claus d'ús més crític.

Exemple de generació de claus RSA

Procedim a generar una clau RSA de 32 bits seguint els passos detallats a l'algorisme de generació de claus:

- 1) Seleccionem dos primers, per exemple, $p = 5879$ i $q = 484487$.
- 2) Calculem n :

$$n = pq = 5879 \cdot 484487 = 2848299073$$

Podem comprovar que, efectivament, la clau generada és de 32 bits, ja que:

$$2^{31} \leq 2848299073 < 2^{32}$$

- 3) Calculem $\phi(n)$:

$$\phi(n) = (p-1)(q-1) = (5879-1)(484487-1) = 2847808708$$

- 4) Seleccionem $e = 1535231195$.
Comprovem que, efectivament:

$$\text{mcd}(e, \phi(n)) = \text{mcd}(1535231195, 2847808708) = 1$$

- 5) Calculem d :

$$d = e^{-1} \pmod{\phi(n)} = 1437751395$$

Podem comprovar que, efectivament:

$$de = 1535231195 \cdot 1437751395 = 1 \pmod{2847808708}$$

- 6) Obtenim:

$$k_{pub} = (n, e) = (2848299073, 1535231195)$$

$$k_{priv} = (d) = (1437751395)$$

Per tal de xifrar un missatge amb RSA, s'aplicarà l'algorisme de xifrat, que fa servir la clau pública del destinatari:

Algorisme de xifratge RSA

A partir d'un missatge en clar m i la clau pública del destinatari $k_{pub} = (n, e)$, es calcula el missatge xifrat c :

$$c = m^e \pmod n$$

Factorització d'enters

L'empresa RSA Laboratories va esponsoritzar durant uns anys una sèrie de reptes de factorització de mòduls RSA. Dins d'aquests reptes, un mòdul de 512 bits va ser factoritzat amb èxit el 1999, un de 704 bits el 2012, i un de 729 el maig de 2016.

Quan el destinatari rebí el missatge xifrat c , podrà desxifrar-lo fent servir la seva clau privada (que només ell coneix):

Algorisme de desxifratge RSA

A partir d'un missatge en xifrat c i la clau privada del destinatari $k_{priv} = (d)$, es recupera el text en clar m :

$$m = c^d \pmod n$$

Noteu que, per tal de poder desxifrar correctament un missatge, caldrà que el missatge en clar original m sigui menor que el mòdul n .

Podem veure com, efectivament, desxifrar un missatge que ha estat prèviament xifrat resulta en l'obtenció del missatge original m :

$$c^d \pmod n = (m^e)^d \pmod n = m^{de} \pmod n = m$$

Per a realitzar l'últim pas, recordem, d'una banda, que seguint l'algorisme de generació de claus assegurem que:

$$d \cdot e = 1 \pmod{\phi(n)}$$

D'altra banda, el teorema d'Euler estableix que si x i n són coprimers, aleshores:

$$x^{\phi(n)} = 1 \pmod n$$

I, per tant:

$$x^{de} \pmod n = x^{1+t\phi(n)} \pmod n = x^1 \cdot x^{t\phi(n)} \pmod n = x \cdot x^{\phi(n)t} \pmod n = x \cdot 1 = x \pmod n$$

Exemple de xifratge i desxifratge amb RSA

L'usuari Bob és el propietari del parell de claus RSA generats a l'exemple anterior. Si l'Alice vol enviar un missatge xifrat $m = 424242$ a en Bob, procedirà de la manera següent:

$$\begin{aligned} c &= m^e \pmod n \\ &= 424242^{1535231195} \pmod{2848299073} \\ &= 1914597261 \end{aligned}$$

En Bob, per a desxifrar c i obtenir el missatge en clar original, procedirà a calcular:

$$\begin{aligned} m &= c^d \pmod n \\ &= 1914597261^{1437751395} \pmod{2848299073} \\ &= 424242 \end{aligned}$$

3.2. Xifratge basat en el logaritme discret: ElGamal

ElGamal és un criptosistema de clau pública basat en el problema del logaritme discret. En concret, ElGamal està basat en l'algorisme d'intercanvi de claus de Diffie-Hellman que s'ha presentat anteriorment a l'apartat 2. El criptosistema deu el seu nom al seu creador, Taher ElGamal, que el va descriure el 1985.

Algorisme de generació de claus ElGamal

L'algorisme de generació de claus ElGamal consta dels passos següents:

- 1) Es tria un primer p i un element α d'ordre q .
- 2) Es tria un valor aleatori $d = k_{priv} \in [2, \dots, p-2]$ i es calcula $\beta = \alpha^d \pmod p$.
- 3) La clau pública és $k_{pub} = (p, \alpha, \beta)$ i la clau privada és el valor $k_{priv} = d$.

Element α

No és necessari que α sigui un element primitiu de \mathbb{Z}_p^* , pot ser-ho d'un subgrup de \mathbb{Z}_p^* d'ordre q .

Per tal de xifrar un missatge amb ElGamal, es procedirà a aplicar l'algorisme de xifrat.

Algorisme de xifratge ElGamal

A partir d'un missatge en clar m i la clau pública del destinatari $k_{pub} = (p, \alpha, \beta)$, es calcula el missatge xifrat c :

- 1) Es tria un nombre aleatori v i es calcula $c_1 = \alpha^v \pmod p$.
- 2) Es recupera la clau pública del receptor i es calcula $c_2 = m \cdot \beta^v \pmod p$.
- 3) S'envia el missatge xifrat (c_1, c_2) .

Noteu que el xifratge amb el criptosistema ElGamal és probabilístic, ja que per a una mateixa clau pública i un mateix missatge en clar, es poden generar múltiples textos xifrats, triant diferents valors aleatoris v durant el procés de xifrat.

D'altra banda, el xifrat amb ElGamal és expansiu, ja que per a un missatge de mida m es generen textos xifrats de mida $2m$.

Quan un receptor rep el parell de valors que conformen un missatge xifrat, procedirà a desxifrar-lo amb l'algorisme de desxifrat.

Algorisme de desxifratge ElGamal

A partir d'un missatge xifrat (c_1, c_2) i la clau privada del destinatari k_{priv} , es recupera el text en clar m :

$$m = \frac{c_2}{c_1^d} \pmod{p}$$

Exemple de generació de claus ElGamal

Un usuari vol generar un parell de claus ElGamal i procedeix a executar l'algorisme de generació de claus:

- 1) Es tria un primer $p = 3725468627$ i un element $\alpha = 150083912$.
- 2) Es tria un valor aleatori $d = 807878087$ i es calcula:

$$\beta = \alpha^d \pmod{p} = 150083912^{807878087} \pmod{3725468627} = 3398986020$$
- 3) La clau pública és $k_{pub} = (3725468627, 150083912, 3398986020)$ i la clau privada és el valor $k_{priv} = 807878087$.

Exemple de xifratge i desxifratge amb ElGamal

L'usuari Bob és el propietari del parell de claus ElGamal generats en l'exemple anterior. Si l'Alice vol enviar un missatge xifrat $m = 424242$ a en Bob, procedirà de la manera següent:

- 1) Es tria un nombre aleatori $v = 1052400195$ i es calcula:

$$c_1 = \alpha^v \pmod{p} = 150083912^{1052400195} \pmod{3725468627} = 434020969$$
- 2) Es calcula:

$$c_2 = m \cdot \beta^v \pmod{p} = 424242 \cdot 3398986020^{1052400195} \pmod{3725468627} = 2787237740$$
- 3) S'envia el missatge xifrat:

$$(c_1, c_2) = (434020969, 2787237740)$$

En Bob, per a desxifrar (c_1, c_2) i obtenir el missatge en clar original, procedirà a calcular:

$$\begin{aligned} m &= \frac{c_2}{c_1^d} \pmod{p} = \frac{2787237740}{434020969^{807878087}} \pmod{3725468627} \\ &= \frac{2787237740}{1565777894} \pmod{3725468627} \\ &= 2787237740 \cdot 1602291419 \pmod{3725468627} \\ &= 424242 \end{aligned}$$

Desxifrar amb ElGamal

Per tal de calcular $\frac{c_2}{c_1^d} \pmod{p}$, es pot calcular l'invers modular de c_1^d i multiplicar el resultat per c_2 .

4. Signatures digitals

Més enllà d'oferir una solució al problema de la distribució de claus, la criptografia de clau pública ens permet realitzar signatures digitals.

Tradicionalment, una signatura analògica, realitzada en bolígraf sobre un paper, permet demostrar que una persona concreta l'ha generada. Així, per exemple, les signatures permeten donar validesa a contractes legals, autoritzar compres amb targetes sense pin o emetre txecs. D'una manera anàloga, les signatures digitals ens permeten demostrar que el propietari d'una determinada clau privada ha realitzat una signatura sobre un document, de manera que només el propietari és capaç de generar una signatura vàlida per a aquell document i que qualsevol que conegui la clau pública associada és capaç de validar-la.

Generalment, els algorismes de signatures digitals comprenen tres funcions bàsiques: la generació de claus, la generació de signatures i la validació de signatures. L'algorisme de generació de claus retorna un parell de claus de criptografia asimètrica, $[k_{pub}, k_{priv}]$; l'algorisme de signatura rep un missatge m i una clau privada k_{priv} i genera una signatura digital s del missatge m ; i l'algorisme de verificació rep una signatura s , un missatge m i una clau pública k_{pub} i valida la correctesa de la signatura.

Signatura digital

Les funcions bàsiques d'un esquema de signatura digital són:

```
[ $k_{pub}$ ,  $k_{priv}$ ] = generació_claus()  
 $s$  = signatura( $k_{priv}$ ,  $m$ )  
 $v$  = validació( $s$ ,  $m$ ,  $k_{pub}$ )
```

Atès que només el propietari de la clau privada és capaç de generar signatures amb aquella clau, diem que les signatures digitals ofereixen la propietat de no-repudi.

Propietat de no-repudi

La propietat de **no-repudi** s'aconsegueix quan un usuari que realitza una acció (per exemple, signar un contracte) no pot negar posteriorment que l'acció ha estat realitzada per ell.

Així doncs, com que el propietari d'una clau privada n'és l'únic coneixedor, no podrà negar que ha signat algun document, ja que és l'únic capaç de realitzar aquella signatura. Així mateix, cal remarcar que per a validar una signatura digital només cal conèixer la clau pública i el missatge signat, és a dir, la validació d'una signatura digital es pot dur a terme per a qualsevol part que conegui la clau pública.

Finalment, les signatures digitals ofereixen integritat sobre els documents signats. En efecte, si un atacant modifica el contingut del document signat, la signatura s'invalida, i el receptor pot detectar aquesta modificació.

4.1. Signatures basades en la factorització d'enters: RSA

L'esquema de signatura RSA està basat en l'algorisme de xifra RSA que s'ha descrit al subapartat 3.1. De la mateixa manera, la seguretat de l'algorisme de signatura RSA recau en la dificultat de factoritzar productes de dos primer grans.

Algorisme de signatura RSA

A partir d'un missatge en clar m i la clau privada de l'emissor $k_{priv} = (d)$, es calcula la signatura digital del missatge s :

$$s = m^d \pmod n$$

Quan el destinatari rebí el missatge m i la seva signatura s , podrà verificar la signatura fent servir la clau pública de l'emissor (que és de domini públic):

Algorisme de validació de signatura RSA

A partir d'un missatge m , la seva signatura s , i la clau pública del destinatari $k_{pub} = (n, e)$, es valida la signatura calculant m' :

$$m' = s^e \pmod n$$

També cal validar que $m' == m$. Si $m' = m$, aleshores la signatura digital és vàlida, mentre que en cas contrari la signatura digital és invàlida.

Efectivament, si la signatura s no s'ha modificat, aleshores:

$$m' = s^e \pmod n = (m^d)^e = m^{de} = m \pmod n$$

Exemple de signatura i validació amb RSA

L'usuari Bob és el propietari del parell de claus RSA que s'ha fet servir en els exemples sobre generació de claus i xifratge amb RSA:

$$\begin{aligned}k_{pub} = (n, e) &= (2848299073, 1535231195) \\k_{priv} = (d) &= (1437751395)\end{aligned}$$

En Bob vol enviar el missatge $m = 424242$ signat a l'Alice. Per a fer-ho, procedirà de la manera següent:

$$\begin{aligned}s &= m^d \pmod n \\&= 424242^{1437751395} \pmod{2848299073} \\&= 2060449075\end{aligned}$$

Quan el destinatari, en aquest cas l'Alice, rebí el missatge i la signatura, podrà validar la correctesa de la signatura a partir de la clau pública de Bob, calculant:

$$\begin{aligned}m' &= s^e \pmod n \\&= 2060449075^{1535231195} \pmod{2848299073} \\&= 424242\end{aligned}$$

I comprovant que el valor m' obtingut és igual al missatge m .

4.2. Signatures basades en el logaritme discret: ElGamal

L'algorisme de signatura d'ElGamal va ser proposat el 1985 i es basa en la dificultat de calcular el logaritme discret.

A diferència de l'RSA, en què els esquemes de xifratge i signatura són molt similars, l'algorisme de signatura d'ElGamal presenta diferències notables amb l'algorisme de xifratge.

Algorisme de signatura ElGamal

A partir d'un missatge en clar m i la clau privada de l'emissor $k_{priv} = d$, es calcula la signatura digital del missatge s :

- 1) Es tria un valor aleatori $h \in [2, p - 2]$ coprimer amb $p - 1$ (és a dir, $\text{mcd}(h, p - 1) = 1$).
- 2) Es calcula el valor $r = \alpha^h \pmod p$.
- 3) Es troba el valor s tal que $m = dr + hs \pmod{p - 1}$. El valor s es pot trobar calculant: $s = (m - d \cdot r) \cdot h^{-1} \pmod{p - 1}$.
- 4) La signatura correspon al parell de valors (r, s) .

Quan el destinatari rebí el missatge m i la seva signatura (r,s) , podrà verificar la signatura fent servir la clau pública de l'emissor (que és de domini públic):

Càlcul del valor s

Noteu que sempre podem calcular $h^{-1} \pmod{p-1}$, ja que en triar h hem assegurat que $\text{mcd}(h, p-1) = 1$.

Algorisme de validació de signatura ElGamal

A partir d'un missatge m , la seva signatura (r,s) , i la clau pública de l'emissor $k_{pub} = (p, \alpha, \beta)$, es valida la signatura calculant:

$$t = \beta^r r^s \pmod{p}$$

i comprovant si:

$$t == \alpha^m \pmod{p}$$

Si la igualtat es compleix, la signatura és correcta. En cas contrari, la signatura és incorrecta.

Noteu que l'esquema de signatura ElGamal, de la mateixa manera que amb l'esquema de xifratge, és probabilístic: per a un mateix missatge i una mateixa clau privada es poden generar múltiples signatures vàlides, variant el paràmetre h .

Exemple de signatura i validació amb ElGamal

L'usuari Bob és el propietari del parell de claus ElGamal que s'han fet servir en els exemples sobre generació de claus i xifratge amb ElGamal:

$$k_{pub} = (p, \alpha, \beta) = (3725468627, 150083912, 3398986020) \\ k_{priv} = 807878087$$

En Bob vol enviar el missatge $m = 424242$ signat a l'Alice. Per a fer-ho, procedirà de la manera següent:

- 1) Tria un valor aleatori $h = 249$ (comprovant que $\text{mcd}(249, 3725468627 - 1) = 1$).
- 2) Calcula el valor r :

$$r = \alpha^h \pmod{p} = 150083912^{249} \pmod{3725468627} = 1675101370$$

- 3) Troba el valor s tal que $m = dr + hs \pmod{p-1}$.

$$s = (m - dr) \cdot h^{-1} \pmod{p-1} = (424242 - 807878087 \cdot 1675101370) \cdot 249^{-1} \\ \pmod{3725468626} = 1431688902$$

- 4) La signatura correspon al parell de valors (r,s) :

$$(r,s) = (1675101370, 1431688902)$$

Quan el destinatari, en aquest cas l'Alice, rebí el missatge i la signatura, podrà validar la correctesa de la signatura a partir de la clau pública de Bob, calculant:

$$t = \beta^r r^s \pmod{p} = 3398986020^{1675101370} \cdot 1675101370^{1431688902} \\ \pmod{3725468627} = 1954079850 \\ \alpha^m \pmod{p} = 150083912^{424242} \pmod{3725468627} = 1954079850$$

I comprovant que, efectivament, el resultat és igual al valor t . Així la signatura es dona per vàlida.

Càlcul del valor s

Noteu que el càlcul del valor s es realitza mòdul $p-1$ i no pas mòdul p .

4.2.1. L'estàndard DSA

El DSA (per les seves sigles en anglès, *digital signature algorithm*) és una variant molt popular de l'algorisme de signatura d'ElGamal, que va ser proposada el 1991. Aquesta popularitat és deu, en part, al fet que el 1994 es va considerar un estàndard per a signatures digitals (DSS) del FIPS (de l'anglès, *federal information processing standards*), un conjunt d'estàndards públics que desenvolupa el Govern Federal dels Estats Units.

Algorisme de generació de claus DSA

Els passos que s'han de seguir en l'algorisme de generació de claus DSA són els següents:

- 1) Es tria un primer p tal que $2^{L-1} < p < 2^L$.
- 2) Es busca un divisor q de $p-1$ tal que q sigui primer i $2^{N-1} < q < 2^N$.
- 3) Es busca un element g d'ordre q a \mathbb{Z}_p ($1 < g < p$), és a dir, g és un generador del subgrup de q elements.
- 4) Es tria un valor aleatori $x = k_{priv}$ tal que $0 < x < q$.
- 5) Es calcula la clau pública $y = g^x \pmod p$.
- 6) La clau pública és $k_{pub} = (p, q, g, y)$ i la clau privada és el valor $k_{priv} = (x)$.

L'estàndard especifica quatre possibles alternatives per a l'elecció de la mida (en bits) dels valors p i q (respectivament, els valors L i N).

Taula 1. Valors per a L i N detallats a l'estàndard

L	N
1024	160
2048	224
2048	256
3072	256

Algorisme de signatura DSA

A partir d'un missatge en clar m i la clau privada de l'emissor $k_{priv} = (x)$, es calcula la signatura digital del missatge (r, s) :

- 1) Es genera un valor aleatori k tal que $0 < k < q$.
- 2) Es calcula el valor $r = (g^k \pmod p) \pmod q$.
- 3) Es troba el valor s tal que $m = k \cdot s - x \cdot r \pmod q$. El valor s es pot trobar calculant: $s = k^{-1}(m + x \cdot r) \pmod q$.
- 4) La signatura correspon al parell de valors (r, s) .

Si durant el procés de generació de signatura es donés el cas que s o r fossin 0, aleshores es repetiria el procés triant un nou valor k . D'aquesta manera, s'assegura que la signatura generada mai tingui un 0 en cap de les dues parts que la formen.

Algorisme de validació de signatura DSA

A partir d'un missatge m , la seva signatura (r,s) , i la clau pública de l'emissor $k_{pub} = (p,q,g,\gamma)$, es valida la signatura fent els càlculs següents:

1) Es comprova que $s \neq 0$ i $r \neq 0$.

2) Es calcula:

$$\begin{aligned}w &= s^{-1} \pmod q \\u_1 &= mw \pmod q \\u_2 &= rw \pmod q \\v &= (g^{u_1} \gamma^{u_2} \pmod p) \pmod q\end{aligned}$$

3) Es valida que $v == r \pmod q$.

Si la igualtat es compleix, la signatura és correcta. En cas contrari, la signatura és incorrecta.

4.3. Atacs als esquemes de signatura digital

En el context de les signatures digitals, existeixen tres tipus d'atacs de falsificació:

Falsificació existencial

Direm que un adversari realitza un atac de **falsificació existencial** quan aquest és capaç de crear almenys una signatura s corresponent a un missatge m .

En aquest tipus d'atacs, l'adversari no té cap control sobre el valor m , és a dir, m pot prendre qualsevol valor (el contingut del missatge no és important). L'atacant aconsegueix el seu objectiu només pel fet d'aconseguir una signatura vàlida s . Els atacs de falsificació existencial són els més senzills de realitzar.

Falsificació selectiva

Un atac de **falsificació selectiva** és un atac de falsificació en què l'adversari té com a objectiu crear una signatura vàlida per a un missatge m que ha triat el mateix adversari amb anterioritat a l'inici de l'atac.

Així, si un atacant és capaç de dur a terme un atac de falsificació selectiva, això implica que és capaç també de dur a terme un atac de falsificació existencial.

Falsificació universal

Direm que un adversari pot dur a terme un atac de **falsificació universal** si és capaç de crear una signatura s vàlida per a qualsevol missatge donat m .

Un adversari capaç de realitzar falsificació universal pot, per tant, crear signatures vàlides per a missatges triats a l'atzar, seleccionats per ell mateix, seleccionats per una tercera part, etc.

4.3.1. Falsificació existencial de signatures RSA

L'algorisme de signatura RSA explicat anteriorment és susceptible a atacs de falsificació existencial de signatures.

En efecte, donada una clau pública $k_{pub} = (n, e)$, un atacant pot procedir de la manera següent:

- 1) L'atacant tria una signatura $s \in \mathbb{Z}_n$.
- 2) L'atacant calcula el missatge $m = s^e \pmod n$.
- 3) L'atacant obté una signatura vàlida s per al missatge m .
- 4) La signatura és vàlida, ja que $s^e = m \pmod n$.

Noteu que, en aquest cas, l'atacant és capaç de construir signatures vàlides, però no té cap mena de control sobre els missatges que està signant.

4.3.2. Falsificació existencial de signatures ElGamal

L'algorisme de signatura ElGamal explicat anteriorment també és susceptible de rebre atacs de falsificació existencial.

En efecte, donada una clau pública $k_{pub} = (p, \alpha, \beta)$, un atacant pot procedir de la manera següent:

1) Tria dos enters i i j tals que $\text{mcd}(j, p-1) = 1$.

2) Calcula la signatura:

$$\begin{aligned} r &= \alpha^i \beta^j \pmod{p} \\ s &= -rj^{-1} \pmod{p-1} \end{aligned}$$

3) Calcula el missatge:

$$m = si \pmod{p-1}$$

4) L'atacant obté una signatura vàlida (r, s) per al missatge m .

5) La signatura és vàlida, ja que la igualtat $\beta^r r^s = \alpha^m$ es manté.

Vegem perquè, efectivament, la validació de la signatura és correcta:

$$\begin{aligned} \beta^r r^s \pmod{p} &= \alpha^{dr} r^s \pmod{p} \\ &= \alpha^{dr} \alpha^{(i+dj)s} \pmod{p} \\ &= \alpha^{dr} \alpha^{(i+dj)(-rj^{-1})} \pmod{p} \\ &= \alpha^{dr-dr} \alpha^{(-rij^{-1})} \pmod{p} \\ &= \alpha^{si} \pmod{p} \\ &= \alpha^m \pmod{p} \end{aligned}$$

De la mateixa manera que amb els atacs de falsificació existencial de signatures RSA, en aquest cas l'atacant tampoc té cap control sobre els missatges signats.

4.3.3. Vulnerabilitat en la reutilització de valors ElGamal

L'algorisme de signatura ElGamal fa servir un valor aleatori h a l'hora de signar els missatges. Aquest valor és el que permet que l'algorisme sigui probabilístic i que existeixin múltiples signatures vàlides per a un únic missatge i una clau determinada. L'aleatorietat en la selecció del paràmetre h és, però, crucial per a la seguretat del sistema. De fet, la seva reutilització permet a un atacant descobrir la clau privada feta servir per a crear les signatures digitals.

En efecte, l'algorisme de signatura ElGamal, així com algunes de les seves variants, té una vulnerabilitat per la qual un atacant que obté dues signatures diferents, sig_1 i sig_2 , realitzades amb la mateixa clau privada d i fent servir el mateix valor h , pot recuperar la clau privada feta servir per a signar:

$$\begin{aligned} sig_1 &= (r, s_1) \\ sig_2 &= (r, s_2) \\ s_1 \cdot h &= (m_1 - d \cdot r) \pmod{p-1} \\ s_2 \cdot h &= (m_2 - d \cdot r) \pmod{p-1} \\ (s_2 - s_1) \cdot h &= m_2 - m_1 \pmod{p-1} \\ h &= \frac{m_2 - m_1}{s_2 - s_1} \pmod{p-1} \\ d &= (m_1 - hs_1) \cdot r^{-1} \pmod{p-1} \end{aligned}$$

Noteu que el càlcul descrit només pot realitzar-se si $(s_2 - s_1)$ és invertible, és a dir, si $\text{mcd}(s_2 - s_1, p - 1) = 1$. En cas contrari, es pot realitzar el càlcul descomposant $p - 1$ i tractant els casos concrets individualment. Per tant, és important que el valor h sigui únic per a cada signatura.

Aconseguir una font d'aleatorietat prou bona com per garantir que el valor h serà únic per a cada una de les signatures realitzades per un dispositiu pot ser problemàtic en segons quins entorns (per exemple, en telèfons mòbils). Per aquest motiu, existeix una variant del DSA determinista, on el valor h queda determinat de manera única per la clau privada i el missatge que es vol signar. L'algorisme que genera el valor h actua de manera similar a un generador pseudoaleatori, fent servir la clau privada i el hash del missatge com a llavors del generador. Noteu que, amb aquesta versió del DSA, amb una mateixa clau només es podrà generar una única signatura per a un missatge concret. Addicionalment, és important notar que la variant determinista del DSA no modifica l'algorisme de generació de claus ni la validació de signatures, de manera que és compatible amb sistemes que implementen el DSA probabilístic.

5. Criptografia simètrica i asimètrica

Més enllà de les propietats que ens ofereix la criptografia de clau pública, aquesta també difereix de la criptografia simètrica en la mida de les claus i les necessitats computacionals dels seus algorismes. Pel que fa a la mida de les claus, normalment es fa servir el concepte de nivell de seguretat per a avaluar la fortalesa d'un algorisme criptogràfic amb una mida de claus concreta, i poder així comparar la seguretat oferta pels diferents algorismes.

Nivell de seguretat d'un algorisme criptogràfic

El **nivell de seguretat** d'un algorisme criptogràfic és el número de passos que necessita el millor atac conegut per a descobrir la clau. Direm que un algorisme proporciona una seguretat d' n bits quan el millor atac necessita 2^n passos.

És important notar que el nivell de seguretat d'un algorisme pot variar amb el descobriment de nous atacs que milloren l'eficiència dels ja coneguts.

Tenint en compte la definició de nivell de seguretat, és fàcil veure que un algorisme de criptografia simètrica amb mida de clau n bits ens proporciona una seguretat d' n bits. En canvi, calcular el nivell de seguretat d'un algorisme de clau pública no és tan directe. La taula 2 ens descriu els nivells de seguretat que s'assumeixen avui en dia per als algorismes de clau simètrica i asimètrica més populars.

Taula 2. Nivell de seguretat segons la mida de la clau

Algorisme	Nivell de seguretat			
	80	128	192	256
AES	80	128	192	256
RSA	1024	3072	7680	15360
ElGamal	1024	3072	7680	15360
DSA	1024	3072	7680	15360

Com es pot apreciar, com més gran és la mida de la clau utilitzada, més gran és la seguretat que ens ofereix. A més, per tal d'obtenir un mateix nivell de seguretat en un criptosistema de clau pública que en un de clau simètrica, caldrà que la clau del primer sigui molt més gran que la del segon. Això, unit al fet que els algorismes de clau pública requereixen càlculs computacionalment intensius, fa que en general els algorismes de clau pública siguin més lents que els de clau simètrica. Això no és un problema per als ordinadors actuals, però

Criptografia de corbes el·líptiques

Els criptosistemes de clau pública basats en corbes el·líptiques (que queden fora de l'abast d'aquest document) permeten obtenir el mateix nivell de seguretat que els algorismes de clau simètrica amb mides de clau superiors però molt més similars. Així, per exemple, caldrà una clau ECDSA de 160 bits per a aconseguir 80 bits de seguretat.

sí que pot ser-ho en dispositius amb capacitats més limitades com targetes intel·ligents. A l'apartat 6, es descriuen dues de les tècniques més utilitzades per a accelerar el procés de xifratge i desxifratge amb RSA.

Per tal d'aprofitar els avantatges de la criptografia de clau pública pel que fa a la gestió de claus i alhora la rapidesa de la criptografia simètrica per a xifrar, sovint es fa servir la tècnica coneguda com a sobre digital.

Sobre digital

La tècnica del **sobre digital** consisteix a xifrar un missatge amb una clau simètrica aleatòria k i xifrar la clau simètrica k amb una clau pública.

A l'hora de realitzar signatures digitals, si volguéssim signar directament els missatges amb un sistema de clau pública ens trobaríem també amb la limitació de la mida de la clau, que reduiria en gran mesura el conjunt de missatges que seríem capaços de signar. En aquest cas, el que es fa és combinar les signatures digitals amb les funcions hash que s'han vist en el mòdul anterior. És a dir, a l'hora de signar un missatge, l'usuari procedeix primer a calcular-ne un resum per mitjà d'una funció hash, signant després aquest resum (en comptes del missatge original). Com que les funcions hash tenen una sortida de mida fixada, això ens permet assegurar que podem signar qualsevol missatge (de qualsevol mida) amb una clau d'una mida concreta (fent ús d'una funció hash amb una sortida de mida inferior a la clau).

Signatura amb hash

El procediment que cal seguir per tal de signar un missatge m fent servir una funció hash H és:

- 1) Calcular el hash del missatge, $h = H(m)$.
- 2) Calcular la signatura del hash del missatge,

$$s = \text{signatura}(k_{\text{priv}}, h).$$

El receptor del missatge m podrà validar la signatura s , procedint de la manera següent:

- 1) Calcular el hash del missatge, $h = H(m)$.
- 2) Validar la signatura s , $v = \text{validació}(s, h, k_{\text{pub}})$.

Noteu que en aquest cas emissor i receptor s'han de posar d'acord no només amb l'algorisme de signatura que utilitzaran, sinó també amb la funció hash que faran servir.

6. Implementació dels algorismes de clau pública

Els algorismes de criptografia de clau pública descrits fins ara requereixen l'ús d'operacions computacionalment costoses per a funcionar. Per aquest motiu, a l'hora d'implementar aquests algorismes es fan servir optimitzacions que permeten millorar-ne l'eficiència. En aquest apartat veurem algunes de les optimitzacions més populars que s'utilitzen a l'hora d'implementar l'RSA i ElGamal.

6.1. Optimització del xifrat RSA

L'algorisme de generació de claus RSA que hem vist al subapartat 3.1. tria un exponent públic e de manera aleatòria, amb les condicions que e estigui en l'interval $(1, \phi(n))$ i que e sigui coprimer amb $\phi(n)$. A la pràctica, però, s'acostumen a triar valors d' e petits i amb pes de Hamming també petit, ja que això fa que el xifratge sigui més eficient.

En concret, per a xifrar (o per a validar una signatura) amb RSA necessitem calcular $x^e \pmod n$. Si fem servir l'algorisme d'exponenciació ràpida de multiplicar i elevar, caldran de l'ordre de $\log_2 e + pes(e)$ operacions per tal de realitzar el càlcul, amb $pes(e)$ representant el pes de Hamming de la representació binària d' e . Valors d' e petits minimitzen el terme $\log_2 e$, mentre que valors amb pes de Hamming petit minimitzen el terme $pes(e)$.

Així, alguns dels valors que es fan servir habitualment per a l'exponent públic e són 3 i 65537 ($2^{16} + 1$), que requereixen 3 i 17 operacions per a xifrar, respectivament. Aquest últim és el valor per ommissió que fa servir la llibreria OpenSSL.* El valor 65537 té alguns avantatges. En primer lloc, és un primer de Fermat, el que fa que calguin poques multiplicacions per a elevar a aquest valor i alhora simplifica la cerca dels primers p i q adequats. En segon lloc, i a diferència de 3, $2^{16} + 1$, és prou gran com per evitar certs atacs que es poden dur a terme amb valors d' e petits si no es fa servir *padding* de manera correcta. Al mòdul següent veurem alguns d'aquests atacs, així com les tècniques que es fan servir per a evitar-los.

La selecció de l'exponent públic fent servir aquestes consideracions fa que xifrar amb RSA sigui molt més ràpid que desxifrar, i també redueix el temps de comprovació de les signatures RSA. Tot i així, aquest no és l'únic factor que afecta la velocitat de les implementacions d'RSA. Tot i que l'algorisme d'exponenciació ràpida de multiplicar i elevar permet xifrar missatges amb RSA de manera molt ràpida, aquest no es fa servir per a desxifrar (ni per signar), ja que és vulnerable a atacs de canal lateral.

Pes de Hamming

El **pes de Hamming** d'una seqüència binària és el número d'uns que conté. El pes de Hamming és equivalent a la distància de Hamming entre una seqüència donada i la seqüència de zeros de la mateixa longitud.

Primers de Fermat

Els **primers de Fermat** són primers de la forma $F_n = 2^{(2^n)} + 1$. Els únics primers de Fermat el febrer de 2016 són $F_0 = 3$, $F_1 = 5$, $F_2 = 17$, $F_3 = 257$ i $F_4 = 65537$.

* <https://goo.gl/HhD8cz>

Atacs de canal lateral

Els atacs de **canal lateral** (en anglès, *side channel attacks*) són atacs que es basen en informació adquirida de la implementació física d'un criptosistema. Aquest tipus d'atacs poden fer servir, per exemple, informació sobre el temps d'execució, el consum energètic, els camps electro-magnètics, el so, etc.

Un atacant que analitzi el consum energètic d'un dispositiu que implementa l'algorisme de multiplicar i elevar pot obtenir directament la clau. D'una banda, per a cada bit de la clau, l'algorisme de multiplicar i elevar calcula una única potència si el bit és 0 o bé una potència i una multiplicació si el bit és 1. D'altra banda, el consum energètic d'un dispositiu implementant aquest algorisme és elevat mentre s'estan realitzant aquestes operacions (tant multiplicació com exponenciació), i baix quan no s'estan fent aquestes operacions. Així, veient la traça de consum energètic del dispositiu, es poden diferenciar clarament cadascuna de les iteracions de l'algorisme (moments de consum energètic elevat separats per instants de consum energètic baix). La durada dels moments de consum energètic elevat ens permet distingir quan s'estan processant bits de la clau a 1 o a 0. D'aquesta manera, només obtenint la traça de consum energètic d'una única operació, es pot obtenir la clau.

Existeixen altres algorismes d'exponenciació ràpida que ofereixen protecció contra atacs d'anàlisi del consum energètic, com ara l'algorisme d'exponenciació de Montgomery (en anglès, *Montgomery powering ladder*). Aquests algorismes fan que cada iteració, independentment de si aquesta tracta un bit de la clau fixat a 0 o a 1, tingui un còmput similar, de manera que el consum energètic de cada iteració és similar.

6.2. Optimització del desxifrat RSA

A més de triar valors d' e que permetin accelerar el xifratge i la validació de signatures digitals, algunes de les llibreries més populars emmagatzemen tres valors intermedis durant la generació de claus RSA, amb l'objectiu de reduir el temps de desxifratge i de signatura. Aquests valors són:

- exponent 1: $d_p = d \pmod{p-1}$
- exponent 2: $d_q = d \pmod{q-1}$
- coeficient: $q_{inv} = (1/q) \pmod{p}$

Aquests valors permeten optimitzar el càlcul de $c^d \bmod n$ per mitjà del teorema xinès del residu. Així, per a calcular $m = c^d \bmod n$ es procedeix a calcular:

$$\begin{aligned}m_1 &= c^{d_p} \bmod p \\m_2 &= c^{d_q} \bmod q \\h &= q_{inv}(m_1 - m_2) \bmod p \\m &= m_2 + hq\end{aligned}$$

Exemple de desxifrat RSA

Suposem que tenim la clau privada RSA de 64 bits formada pels valors següents:

$$\begin{aligned}d &= 7506774701030841737 \\n &= 10639337161500789943\end{aligned}$$

Suposem que en el moment de generar la clau hem desat també els valors següents:

$$\begin{aligned}p &= 696734729 \\q &= 15270283967 \\d_p &= d \bmod (p - 1) = 7506774701030841737 \bmod (696734729 - 1) = 259426577 \\d_q &= d \bmod (q - 1) = 7506774701030841737 \bmod (15270283967 - 1) = 8567289973 \\q_{inv} &= (1/q) \bmod p = 1/15270283967 \bmod 696734729 = 284277123\end{aligned}$$

Aleshores, per a desxifrar el missatge $c = 3510853621447083634$, procediríem de la manera següent:

$$\begin{aligned}m_1 &= c^{d_p} \bmod p = 3510853621447083634^{259426577} \bmod 696734729 = 627709010 \\m_2 &= c^{d_q} \bmod q = 3510853621447083634^{8567289973} \bmod 15270283967 = 11944757133 \\h &= q_{inv}(m_1 - m_2) \bmod p = 284277123(627709010 - 11944757133) \bmod 696734729 = 27 \\m &= m_2 + hq = 11944757133 + 27 \cdot 15270283967 = 4242424242\end{aligned}$$

Noteu que això és equivalent a fer directament:

$$\begin{aligned}m &= c^d \bmod n = 3510853621447083634^{7506774701030841737} \\&\bmod 10639337161500789943 = 4242424242\end{aligned}$$

En canvi, totes les operacions implicades tenen exponents i mòduls molt menors que els requerits per a aquest càlcul. D'aquesta manera, s'aconsegueix reduir el temps de càlcul.

Podem veure com s'emmagatzemen aquests valors en claus RSA de mida real fent servir l'eina OpenSSL:

```
openssl genrsa -out private.pem 1024
openssl rsa -noout -text -in private.pem
```

Hem vist com s'optimitza el xifratge i la validació de signatures a partir de la tria de l'exponent públic e i com s'optimitza el desxifratge i la realització de signatures emmagatzemant uns valors auxiliars relatius a la clau privada. Després de realitzar aquestes optimitzacions, ens podríem preguntar quin dels dos processos és, doncs, més ràpid. Per a comprovar-ho des d'un punt de vista pràctic, podem recórrer de nou a la llibreria Openssl. En concret, podem analitzar les diferències de temps necessàries per a signar i validar signatures digitals en RSA amb la sentència següent:

```
openssl speed rsa
```

Els resultats de l'execució de l'anterior sentència en un Intel Core i7-4770 CPU @ 3.40GH indiquen clarament que és molt més ràpid validar signatures, i, per tant xifrar, que no pas realitzar les signatures, o desxifrar. Vegeu la taula 3.

Taula 3. Temps per a signar i validar amb RSA

Mida de la clau	Temps per a signar	Temps per a validar	Signatures per segon	Validacions per segon
512 bits	0.000041s	0.000003s	24317.5	328106.1
1024 bits	0.000118s	0.000008s	8485.8	131766.6
2048 bits	0.000548s	0.000024s	1825.2	41486.7
4096 bits	0.005865s	0.000088s	170.5	11299.6

6.3. Optimització del xifrat ElGamal

El procés de xifrat amb ElGamal consta de tres operacions: dues exponenciacions, que permeten calcular $c_1 = \alpha^v \bmod p$ i un operand de c_2 , $c_2^{aux} = \beta^v$; i una multiplicació que realitza el càlcul final de c_2 , $c_2 = m \cdot c_2^{aux}$.

Com en el cas de l'RSA, les exponenciacions es poden calcular amb l'algorisme d'exponenciació ràpida de multiplicar i elevar, de manera que s'agilitza el càlcul. Ambdues exponenciacions fan servir com a exponent el paràmetre aleatori v , de manera que les exponenciacions poden optimitzar-se seleccionant valors v amb propietats especials, com ara valors amb pes de Hamming petit. Si es fa servir aquesta tècnica cal anar en compte, però, de tenir un número adequat d'exponents possibles.

Ara bé, en el cas del xifrat amb ElGamal, hi ha un altre detall important que cal tenir en compte: les dues exponenciacions que es calculen són completament independents del missatge que es xifra. Això fa que en algunes aplicacions aquests valors puguin precalcular-se amb anterioritat al procés de xifratge, en moments en què la càrrega del sistema és baixa. A l'hora de xifrar, caldrà, doncs, recuperar els valors precalculats i calcular únicament la multiplicació, operació que sí que depèn del missatge que es xifra.

6.4. Optimització del desxifrat ElGamal

El procés de desxifrar amb el criptosistema ElGamal consta també de tres operacions: una exponenciació, que permet calcular c_1^d ; una inversió, que calcula $(c_1^d)^{-1}$; i finalment una multiplicació, que recupera m calculant $c_2 \cdot (c_1^d)^{-1}$. El desxifratge es pot optimitzar unint les dues primeres operacions en una sola exponenciació, fent ús del petit teorema de Fermat, que és un cas concret del teorema d'Euler. El petit teorema de Fermat es basa en un p primer:

$$x^{p-1} = 1 \pmod p, \forall x \text{ tal que } \text{mcd}(x,p) = 1$$

Aprofitant aquesta igualtat, podem reduir el càlcul de $(c_1^d)^{-1}$ a una única exponenciació: c_1^{p-d-1} , és a dir, el valor c_1 elevat a l'exponent $p-d-1$. Comprovem que realment les dues alternatives són equivalents:

$$(c_1^d)^{-1} \pmod p = (c_1^d)^{-1} \cdot c_1^{p-1} \pmod p = c_1^{p-d-1} \pmod p$$

D'aquesta manera, desxifrar un missatge amb ElGamal se simplifica i passa a consistir en una exponenciació i una multiplicació.

7. Criptografia postquàntica

Com hem vist, els algorismes de criptografia de clau pública més populars avui en dia es basen en la dificultat de resoldre tres problemes matemàtics: la factorització d'enters, el logaritme discret i el logaritme discret sobre corbes el·líptiques. Aquests problemes, però, deixarien de ser difícils si disposéssim d'un ordinador quàntic de prou capacitat.

L'algorisme de Shor és un algorisme quàntic que permet resoldre aquests problemes en un temps polinomial respecte a la mida de l'entrada. L'algorisme de Shor va ser proposat per Peter Shor l'any 1994 a l'article "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer".

Hi ha tot un conjunt de famílies d'algorismes de criptografia de clau pública que es basen en altres problemes matemàtics, que es creuen difícils de resoldre tant en ordinadors clàssics com en ordinadors quàntics. A tots ells se'ls engloba sota el nom de criptografia postquàntica. Exemples d'aquestes famílies en són la criptografia basada en funcions hash, en codis, en reticles o en equacions quadràtiques multivariades. Així, per exemple, el criptosistema de McEliece, desenvolupat per Robert McEliece el 1978, està basat en la dificultat de descodificar un codi lineal i l'esquema de signatura de Merkle, ideat per Ralph Merkle el 1979, es basa en la generació d'un arbre de hashes.

Atès que els algorismes de criptografia postquàntica són segurs davant d'ordinadors quàntics, hom podria preguntar-se el motiu pel qual el seu ús no es troba molt més estès actualment que el dels algorismes que hem descrit en aquest mòdul, que no són segurs en aquesta situació. D'una banda, l'eficiència computacional dels algorismes de criptografia asimètrica postquàntics és baixa, molt pitjor que la que ens ofereixen actualment l'RSA o ElGamal. D'altra banda, alguns d'aquests esquemes són molt nous, i la comunitat criptogràfica encara no hi ha dipositat prou confiança. Per tal de generar aquesta confiança, és necessari que els criptoanalistes dediquin molt temps a analitzar-los. Finalment, els algorismes criptogràfics sovint necessiten un conjunt de protocols que estandarditzin el seu ús en diferents circumstàncies. Com veurem més endavant, aquest fet pot arribar a ser crític per a la seguretat dels esquemes. La criptografia postquàntica encara no ha arribat a la maduresa en aquest sentit.

Resum

En aquest mòdul hem après el concepte de criptografia de clau pública, concepte que engloba tot un conjunt d'algorismes criptogràfics que fan servir parells de claus: una clau pública coneguda per tothom i una clau privada només coneguda pel seu propietari. En els esquemes de xifrat amb clau pública, els usuaris poden xifrar missatges per a un destinatari utilitzant la clau pública d'aquest destinatari. Alhora, només el destinatari del missatge, que coneix la clau privada, serà capaç de desxifrar els missatges dirigits a ell. Oposadament, els esquemes de signatura digital permetran a un emissor d'un missatge signar-lo amb la seva clau privada (que només ell coneix), permetent que aquesta signatura sigui validada per a qualsevol que en conegui la clau pública.

També s'ha explicat l'intercanvi de claus de Diffie-Hellman, un protocol que permet a dues parts establir un secret comú per mitjà d'un canal insegur. Així mateix, ens hem centrat a descriure els dos criptosistemes de clau pública més utilitzats avui en dia, l'RSA i ElGamal.

La tècnica del sobre digital ens permet disposar dels avantatges de la criptografia de clau pública sense trobar-nos amb les limitacions de mida i de capacitat computacional d'aquesta, combinant criptografia de clau pública amb algorismes de criptografia simètrica.

Exercicis d'autoavaluació

1. Indiqueu quins dels següents parells de claus RSA, $k_{pub} = (n, e)$ i $k_{priv} = (d)$ són vàlids. Per als que no ho són, detalleu-ne el motiu.

- a) $k_{pub} = (3353361769, 1647529266), k_{priv} = (1853372443)$
- b) $k_{pub} = (2660610913, 700422517), k_{priv} = (339543773)$
- c) $k_{pub} = (111086984740301, 1890731431), k_{priv} = (66185553158551)$

2. L'Alice i en Bob són dos usuaris d'un sistema de clau pública RSA. Les seves respectives claus públiques i privades són:

$$k_{pubA} = (3714176377, 1471178161), k_{privA} = (696390481) \\ k_{pubB} = (3720779831, 2037827401), k_{privB} = (2233915321)$$

L'Alice vol enviar el missatge $m = 249$ xifrat a en Bob. Reproduïu el procés de xifratge realitzat per l'Alice i el procés de desxifratge que realitzarà en Bob en rebre el missatge, comprovant que efectivament el missatge rebut per en Bob coincideix amb el text en clar enviat per l'Alice.

3. L'Alice i en Bob s'intercanvien missatges xifrats amb RSA fent servir la convenció de xifrar cada caràcter del missatge per separat, fent servir la codificació ASCII i utilitzant únicament lletres majúscules. Un atacant intercepta aquest missatge de l'Alice dirigit a en Bob:

[2269693817, 1639486112, 1818812718, 2032163849, 3308958529, 251951562, 2224890518, 1639486112, 3489265165, 2032163849, 228316393, 1818812718]

L'atacant coneix la clau pública d'en Bob que l'Alice ha fet servir per a xifrar el missatge, així com les convencions que fan servir l'Alice i en Bob en els intercanvis de missatges:

$$k_{pubB} = (3720779831, 2037827401)$$

Quin és el missatge que l'Alice ha enviat a en Bob?

Nota: Trobeu el missatge sense calcular la clau privada d'en Bob, procés que no podríeu realitzar si les claus utilitzades en l'exercici fossin de mida real.

4. Indiqueu quins dels parells de claus ElGamal, $k_{pub} = (p, \alpha, \beta)$ i $k_{priv} = d$ són vàlids. Per als que no ho són, detalleu-ne el motiu.

- a) $k_{pub} = (1474315399, 79643891, 269853666), k_{priv} = 84990634$
- b) $k_{pub} = (3383730189, 2011758775, 2122190089), k_{priv} = 2878050547$
- c) $k_{pub} = (337681733, 14736556, 93610277), k_{priv} = 144823569$
- d) $k_{pub} = (98011540216022814571886828168594180107, \\ 73706495652936837455240336262679206568, \\ 30382876101164794220971335754154344479), \\ k_{priv} = 61488904351572748379732502783030097644$

5. Calculeu el logaritme discret de 12483 sobre la base de 36848 a \mathbb{Z}_{42841} , és a dir, trobeu el valor x tal que $36848^x = 12483 \pmod{42841}$. Podríeu fer aquest mateix càlcul per a valors de 1024 bits?

6. L'Alice i en Bob són dos usuaris d'un sistema de clau pública ElGamal. Les seves respectives claus públiques i privades són:

$$k_{pubA} = (3346900289, 2210916257, 849352893), k_{privA} = (713795492) \\ k_{pubB} = (3575204279, 1113291034, 792418784), k_{privB} = (2036555019)$$

L'Alice vol enviar el missatge $m = 424242$ xifrat a en Bob. Reproduïu el procés de xifratge realitzat per l'Alice i el procés de desxifratge que realitzarà en Bob en rebre el missatge, comprovant que efectivament el missatge rebut per en Bob coincideix amb el text en clar enviat per l'Alice.

7. L'Alice i en Bob són dos usuaris d'un sistema de clau pública RSA. Les seves respectives claus públiques i privades són:

$$k_{pubA} = (3714176377, 1471178161), k_{privA} = (696390481) \\ k_{pubB} = (3720779831, 2037827401), k_{privB} = (2233915321)$$

L'Alice vol enviar el missatge xifrat següent a en Bob:

$$m = 011235813213455891442333776109871597258441816765$$

Reproduïu el procés de xifratge realitzat per l'Alice.

8. Genereu dues signatures diferents per al missatge $m = 45678$ fent servir l'esquema de signatura ElGamal i valideu, després, les signatures generades. Feu servir el parell de claus:

$$k_{pub} = (p, \alpha, \beta) = (797445667, 386331185, 505206688)$$

$$k_{priv} = (d) = (373845532)$$

Solucionari

1. La clau vàlida és la b .

La clau a no és vàlida, ja que $e \cdot d \bmod \phi(n) \neq 1$:

$$1647529266 \cdot 1853372443 \bmod \phi(3353361769) = 1647529266 \cdot 1853372443 \bmod 3353239120 = 1499866678 \neq 1$$

La clau c no és vàlida, ja que el mòdul no és producte de dos primers:

$$n = 111086984740301 = 45613 \cdot 41141 \cdot 59197$$

2. Per tal de xifrar el missatge, l'Alice farà servir la clau pública d'en Bob i procedirà a calcular:

$$c = m^e \bmod n = 249^{2037827401} \bmod 3720779831 = 1920900242$$

Quan en Bob rebí el missatge xifrat c , procedirà a desxifrar-lo amb la seva clau privada, calculant: $m = c^d \bmod n = 1920900242^{2233915321} \bmod 3720779831 = 249$

3. L'atacant pot aprofitar el fet que coneix tots els possibles missatges que els usuaris s'intercanvien juntament amb el fet que l'RSA no és una xifra probabilística per a generar tots els possibles textos xifrats i comparar el resultat amb el missatge intercanviat. Així, l'atacant procediria a calcular el que es mostra a la taula 4.

Taula 4. Càlculs realitzats per l'atacant

Lletra	m	c
A	65	$65^{2037827401} \bmod 3720779831 = 3489265165$
B	66	$66^{2037827401} \bmod 3720779831 = 3192216487$
C	67	$67^{2037827401} \bmod 3720779831 = 2269693817$
D	68	$68^{2037827401} \bmod 3720779831 = 3031724104$
E	68	$69^{2037827401} \bmod 3720779831 = 1496836939$
...		
G	71	$71^{2037827401} \bmod 3720779831 = 2224890518$
H	72	$72^{2037827401} \bmod 3720779831 = 228316393$
O	79	$79^{2037827401} \bmod 3720779831 = 251951562$
P	80	$80^{2037827401} \bmod 3720779831 = 2032163849$
R	82	$82^{2037827401} \bmod 3720779831 = 1639486112$
T	84	$84^{2037827401} \bmod 3720779831 = 3308958529$
Y	89	$89^{2037827401} \bmod 3720779831 = 1818812718$

L'atacant descobrirà que el missatge xifrat correspon a la cadena:

$$[67, 82, 89, 80, 84, 79, 71, 82, 65, 80, 72, 89]$$

Aquesta cadena alhora codifica el missatge CRYPTOGRAPHY.

4. Les claus a i d són vàlides.

La clau b no és vàlida ja que $p = 3383730189$ no és primer.

La clau c no és vàlida ja que $\beta \neq \alpha^d \bmod p$: $14736556^{144823569} \bmod 337681733 = 93610276 \neq 93610277$.

5. Podem calcular el logaritme discret proposat per força bruta, és a dir, calculant $36848^x \bmod 42841$ per a tots els valors possibles de x (de 0 a 42840), fins a trobar el resultat que busquem, 12483. Així, trobarem que $36848^{4928} \bmod 42841 = 12483$. No podríem seguir aquest mateix enfocament per a valors de 1024 bits, ja que el número de possibilitats a provar és massa gran.

6. Per tal de xifrar el missatge, l'Alice farà servir la clau pública d'en Bob i procedirà a seleccionar un valor aleatori v , per exemple, $v = 8341864$ i calcular:

$$c_1 = \alpha^v \bmod p = 1113291034^{8341864} \bmod 3575204279 = 3323366879$$

$$c_2 = m \cdot \beta^v \bmod p = 424242 \cdot 792418784^{8341864} \bmod 3575204279 = 3166979642$$

El text xifrat és el parell $(c_1, c_2) = (3323366879, 3166979642)$.

Quan en Bob rebí el missatge xifrat (c_1, c_2) , procedirà a desxifrar-lo amb la seva clau privada, calculant:

$$m = \frac{c_2}{c_1} \mod p = \frac{3166979642}{3323366879203655019} \mod 3575204279 = 3166979642 \cdot 663237018 = 424242$$

Noteu que la solució de l'exercici no és única, ja que depèn de la selecció del paràmetre aleatori v .

7. El missatge m és més gran que el mòdul de la clau pública d'en Bob $n = 3720779831$. Per aquest motiu, el missatge no es pot xifrar directament amb RSA. Una alternativa es fer servir la tècnica del sobre digital: l'Alice genera una clau simètrica k aleatòria i xifra el missatge amb la clau k , $c = E_k(m)$, fent servir com a algorisme de xifrat E qualsevol esquema de clau simètrica. L'Alice xifra després la clau k amb la clau pública d'en Bob fent servir RSA, $c_k = E_{K_{pub}}(k)$. Finalment, l'Alice envia a en Bob els dos valors, c i c_k .

8. Seleccionem un valor aleatori h , per exemple, $h = 55$. Calculem:

$$\begin{aligned} r &= \alpha^h \mod p = 386331185^{55} \mod 797445667 = 673983968 \\ s &= (m - d \cdot r) \cdot h^{-1} \mod p - 1 = (45678 - 373845532 \cdot 673983968) \cdot 55^{-1} \\ &\quad \mod 797445666 = 1042804 \end{aligned}$$

La signatura correspondria al parell $(r, s) = (673983968, 1042804)$.

Per a realitzar una segona signatura, seleccionariem un segon valor aleatori h , per exemple, $h = 77$:

$$\begin{aligned} r &= \alpha^h \mod p = 386331185^{77} \mod 797445667 = 205790131 \\ s &= (m - d \cdot r) \cdot h^{-1} \mod p - 1 = (45678 - 373845532 \cdot 205790131) \cdot 77^{-1} \\ &\quad \mod 797445666 = 284046946 \\ (r, s) &= (205790131, 284046946) \end{aligned}$$

Per tal de validar les signatures, procedim a calcular:

$$t = \beta^{r^s} \mod p = 505206688^{673983968} \cdot 673983968^{1042804} \mod 797445667 = 137624270$$

comprovem que el valor sigui igual a $\alpha^m \mod p$:

$$\alpha^m \mod p = 386331185^{45678} \mod 797445667 = 137624270$$

Per a la segona signatura, realitzem el mateix procediment:

$$t = \beta^{r^s} \mod p = 505206688^{205790131} \cdot 205790131^{284046946} \mod 797445667 = 137624270$$

Comprovem que el valor sigui igual a $\alpha^m \mod p$:

$$\begin{aligned} \alpha^m \mod p &= 386331185^{45678} \\ &\mod 797445667 = 137624270 \end{aligned}$$

Noteu que la solució de l'exercici no és única, ja que depèn de les eleccions del paràmetre aleatori h .

Glossari

criptografia asimètrica *f* Conjunt d'algorismes criptogràfics que fan servir parells de claus: una clau pública coneguda i una clau privada només coneguda pel seu propietari.

criptografia de clau pública *f* Vegeu **criptografia asimètrica**.

DHKE *m* Vegeu **intercanvi de claus de Diffie-Hellman**.

Diffie-Hellman key exchange *m* Vegeu **intercanvi de claus de Diffie-Hellman**.

digital signature algorithm *m* Estàndard de signatura digital, implementa una variant de l'algorisme de signatura ElGamal.

sigla **DSA**

DSA *m* Vegeu **digital signature algorithm**.

ECDSA *m* Vegeu **elliptic curve digital signature algorithm**.

element primitiu a \mathbb{Z}_p *m* Element generador de \mathbb{Z}_p .

elliptic curve digital signature algorithm *m* Variant del DSA sobre corbes el·líptiques.

sigla **ECDSA**

ElGamal *m* Algorisme de criptografia de clau pública basat en el problema del logaritme discret sobre un cos finit.

falsificació existencial *f* Atac que permet generar almenys una signatura vàlida, sense que l'adversari tingui cap control sobre el missatge per al qual la signatura és vàlida.

falsificació selectiva *f* Atac que permet generar una signatura vàlida per a un missatge determinat, escollit pel mateix atacant.

falsificació universal *f* Atac que consisteix a generar una signatura vàlida per a qualsevol missatge donat.

intercanvi de claus de Diffie-Hellman *m* Protocol que permet a dues parts que no compartien prèviament cap clau obtenir un secret comú per mitjà d'un canal insegur.

sigla **DHKE**

en Diffie-Hellman key exchange

Institut Nacional d'Estàndards i Tecnologia *m* Agència dels Estats Units que promou la innovació per mitjà de l'estandardització i la metrologia.

no-repudi *m* Propietat de les accions en què el seu autor no en pot negar l'autoria.

RSA *m* Algorisme de criptografia de clau pública basat en el problema de la factorització d'enters.

signatura digital *f* Mecanisme criptogràfic que permet al receptor d'un missatge comprovar-ne la integritat i autenticitat.

sobre digital *m* Tècnica que combina criptografia simètrica i asimètrica per a xifrar contingut, amb la qual cosa permet obtenir els avantatges d'ambdues.

Bibliografia

- Bernstein, D. J.** (2009). *Introduction to post-quantum cryptography* (pàg. 1-14). Heidelberg: Springer.
- Diffie, W.; Hellman, M.** (1976). "New directions in cryptography". *IEEE transactions on Information Theory* (núm. 22.6, pàg. 644-654).
- ElGamal, T.** (1985). "A public key cryptosystem and a signature scheme based on discrete logarithms". *IEEE transactions on information theory* (núm. 31.4, pàg. 469-472).
- Menezes, A. J.; Oorschot, P. C. van; Vanstone, S. A.** (1996). *Handbook of applied cryptography*. Florida: CRC press.
- Institut Nacional d'Estàndards i Tecnologia** (2013). *FIPS PUB 186-4 Digital Signature Standard (DSS)*. Maryland: NIST.
<<http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>>
- Paar, C.; Pelzl, J.** (2009). *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media.
- Pornin, T.** (2013). *RFC6979: Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)*.
<<https://tools.ietf.org/html/rfc6979>>
- Rescorla, E.** (1999). *Diffie-Hellman Key Agreement Method*.
<<https://tools.ietf.org/html/rfc2631>>
- Rivest, R. L., Shamir, A.; Adleman, L.** (1978). "A method for obtaining digital signatures and public-key cryptosystems". *Communications of the ACM* (núm. 21.2, pàg. 120-126).