CriptoTema7.pdf

onafolch

Criptografia i Seguretat

3º Grado en Ingeniería de Datos

Escuela de Ingeniería
Universidad Autónoma de Barcelona

CRIPTOGRAFIA

CRYPTOGRAPHIC PROTOCOLS

1. INTRODUCCIÓ

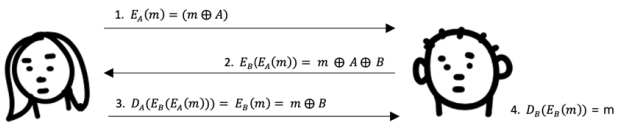
Ens podem trobar diferents situacions on calguin protocols que ens garanteixin un seguit de propietats de seguretat que els criptosistemes per si sols no poden proporcionar. És en aquest punt on intervenen els protocols criptogràfics, protocols entre dos o més usuaris que utilitzen mecanismes criptogràfics per protegir la informació.

2. SHAMIR'S THREE-STEP PROTOCOL

El protocol de tres passos de Shamir té com a objectiu permetre una comunicació secreta entre dues parts sense cap intercanvi previ de claus. El protocol suposa que el criptosistema utilitzat commuta, és a dir, serà el mateix xifrar un missatge m amb una clau k_1, i el resultat tornar-lo a xifrar amb una k_2, que xifrar el missatge m amb una k_2 i després xifrar-lo amb una k_1:

$$E_{k_1}(E_{k_2}(m)) = E_{k_2}(E_{k_1}(m))$$

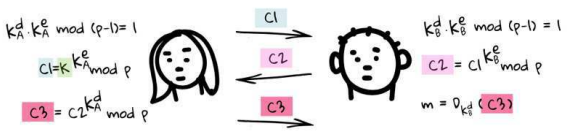
Alice vol enviar un missatge **m** a Bob. Alice crearà el missatge **m**, i també crearà la clau **A**. Tenint en compte que $E_A(x) = (x \oplus A)$, i que per tant $D_A(x) = (x \oplus A)$, Alice xifrarà el missatge m amb la seva clau A ($E_A(m) = (m \oplus A)$) i li enviarà el resultat a Bob. Bob crearà la seva clau **B** i encriptarà el missatge rebut amb la seva clau B ($E_B(E_A(m)) = m \oplus A \oplus B$) i li enviarà a Alice, que desencriptarà aquest missatge amb la seva clau A ($D_A(E_B(E_A(m))) = D_A(E_A(E_B(m))) = E_B(m) = m \oplus B$), i li envia al Bob. Quan el Bob ho rebi, ho desencriptarà amb la seva clau B, i obtindrà el missatge m ($D_B(E_B(m))$ = m).



Però aquest protocol no és totalment segur pel que fa el secret. Si hi ha una persona (Eve) està escoltant tots els missatges, només fent un xor de tots ells pot aconseguir el missatge m:

$$(m \oplus A) \oplus (m \oplus A \oplus B) \oplus (m \oplus B) = m$$

La solució a aquest problema és l'exponenciació, on l'encriptació es fa de la següent manera: $E_A(x) = x^A$. Alice tindrà una clau per xifrar (k_A^e) i una clau per desxifrar (k_A^d). Bob també tindrà una clau per xifrar (k_B^e) i una per desxifrar (k_B^d). Alice li envia el missatge encriptat amb la seva clau al Bob ($c1 = m^{k_A^e} \bmod p$), i quan el Bob ho rebi, encriptarà el que ha rebut amb la seva clau ($c2 = c1^{k_B^e} \bmod p$) i li enviarà a Alice. Alice desencriptarà aquest missatge amb la seva clau per desencriptar ($c3 = c2^{k_A^d} \bmod p$) i li enviarà el Bob, que quan ho desencripti amb la seva clau, obtindrà el missatge ($m = c3^{k_B^d} \bmod p$).



Exemple:

Pas	Alice		Bob
1.	$c_1 = 15^{21} \pmod{131} = 125$	$\xrightarrow{125}$	
2.		$\xleftarrow{27}$	$c_2 = (125)^{27} \bmod 131 = 27$
3.	$c_3 = (27)^{31} \bmod 131 = 129$	$\xrightarrow{129}$	
4.			$m = (129)^{53} \bmod 131 = 15$

$p = 131$
$k_A^e = 21\ i\ k_A^d = (k_A^e)^{-1}\ mod\ (p-1) = 31$
$k_B^e = 27\ i\ k_B^d = (k_B^e)^{-1}\ mod\ (p-1) = 53$
$m = 15$

Ona Folch

2. ZERO KNOWLEDGE PROOFS

És un protocol que inclou un provador i un verificador, que permet al provador demostrar una informació a un verificador sense revelar cap informació. Ha de complir les següents propietats:

1. **Correcció**: Si el provador coneix el valor secret, ha de poder convèncer al verificador que el coneix.
2. **Robustesa**: La probabilitat que el provador enganyi al verificador ha de ser molt petita.
3. **Coneixement nul**: el verificador no té cap informació sobre el valor secret que el provador coneix.

Discrete Logarithm

És un exemple d'una prova de coneixement nul aplicada al coneixement del logaritme discret d'un valor. Donats uns valors **y**, **g** i **p**, és difícil trobar per a quin valor **x** es compleix que $y = g^x \ mod \ p$. Permet al provador demostrar que coneix el valor **x** que compleix l'equació **y** sense necessitat de revelar aquest valor.

El protocol estableix dos paràmetres públics (p,g), i consisteix en repetir els següents 4 passos **n** vegades.

Pas	Provador (P)		Verificador (V)
1.	Tria $r \in_R \mathbb{Z}_p \setminus \{0,1\}$ Calcula $c = g^r \pmod p$	$\xrightarrow{c}$	
2.		$\xleftarrow{b}$	Tria un bit aleatori $b \in_R \{0,1\}$
3.	Calcula $h = r + b \cdot x \pmod{p-1}$	$\xrightarrow{h}$	
4.			Verifica que $c \cdot y^b = g^h \pmod p$

Si V escull b = 0, P podria verificar que sap **x** sense realment saber-ho, ja que hauria de calcular h = r mod(p-1) i no necessita el valor x. Però si V escull b = 1, no pot demostrar que sap x sense saber-ho, perquè necessita el valor per calcular x. La probabilitat de que P pugui enganyar V si es fa el procés n vegades és de 1/(2^n).

V no podria enviar sempre b = 1, ja que P en el primer pas, enlloc d'enviar **r** podria enviar $g^r/_y$. En el pas 3 aleshores enviaria **r** enlloc de **h** i podria enganyar a V.

Exemple:

Pas	Provador (P)		Verificador (V)
1.	Tria $r = 20 \in_R \mathbb{Z}_{89} \setminus \{0,1\}$ Calcula $c = 3^{20} = 73 \pmod{89}$	$\xrightarrow{c=73}$	
2.		$\xleftarrow{b=1}$	Tria un bit aleatori $b = 1$
3.	Calcula $h = 20 + 1 \cdot 9 = 29 \pmod{88}$	$\xrightarrow{h=29}$	
4.			Verifica que $c \cdot y^b = 73 \cdot 14^1 = 43 \pmod{89}$ $g^h \pmod p = 3^{29} = 43 \pmod{89}$

On p = 89 i g = 3. P coneix el logaritme discret de y = 14 (mod 89), que és x = 9. B tria b = 1 en el pas 2.

3. HOMOMORPHIC ENCRYPTION

La criptografia homomòrfica em permet realitzar càlculs amb text xifrat. Per exemple, si tenim $E(a)$ *i* $E(b)$, podem calcular $E(a+b)$, $E(a*b)$ o $E(a)*k$. Un exemple de criptosistemes homomòrfics és Pailler.

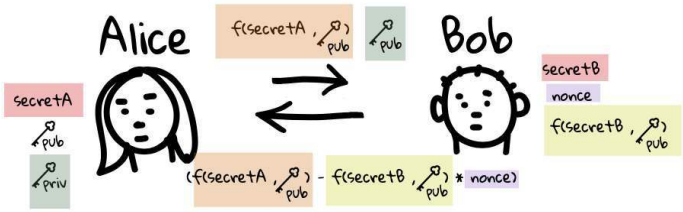
Paillier

És un algoritme homomòrfic asimètric per a la criptografia de clau pública, que ens permet la multiplicació homomòrfica de textos plans i els números xifrats es poden multiplicar per un escalar no xifrat, però no inclou la resta.

Ona Folch

1. Alice escull dos nombres aleatoris primers **p** i **q**, i calcula $\boldsymbol{n} = p * q$, $\boldsymbol{\lambda} = mcm(p-1, q-1)$, $\boldsymbol{L(x)} = (x-1)/n$
2. Alice escull un valor aleatori **g** el qual $mcd\left(\left(L\left(g^{\lambda} \bmod n^2\right)\right), n\right) = 1$
3. La clau pública d'Alice és Pk_A: (n, g)
4. Per encriptar un missatge **m**, Bob escull un valor aleatori **r** i calcula $\boldsymbol{c} = E(m) = g^m * r^n \bmod n^2$
5. $E(a+b) = E(a) * E(b) \bmod n^2 = g^{a+b} * (r_1 * r_2)^n \bmod n^2$
6. Per desencriptar un missatge c, Alice calcula $D(c) = L\left(c^{\lambda} \bmod n^2\right) = m$

Exemple: problema del milionari. Alice i Bob volen saber qui dels dos té més diners sense saber quants diners té l'altre. Primer de tot Alice xifrarà el seu patrimoni amb la seva clau pública, i li envia a Bob el patrimoni xifrat i la clau pública. Quan el Bob ho rep, xifra el seu patrimoni amb la clau pública de Alice, i resta els dos patrimonis xifrats i multiplica el valor resultant de la resta amb un *nonce*, de tal manera que no podran saber el resultat real de la resta. Aquest resultat li envia a Alice, la qual el desxifra amb la seva privada. Si el valor és positiu, significa que Alice té més diners, si és negatiu, significarà que Bob té més diners.



4. COMMITMENT PROTOCOLS

És una tècnica per la qual els usuaris es comprometen amb un valor escollit mentre el mantenen ocult per als altres, amb la capacitat de revelar aquest valor compromès més endavant.

Per exemple, volem que dues persones decideixin de manera justa qui comença alguna activitat en concret. Si el resultat final ((secretA + secretB)%2) és parell, comença la persona A, en cas contrari B.

Un no pot enviar el valor primer i que després l'envii l'altre, ja que el segon usuari podria escollir el valor segons el que rebi de A per tal d'ajustar el resultat segons el que li convingui.

Primer de tot s'envien el hash del compromís en l'ordre que sigui. Quan després s'enviin els números, el hash d'aquests números han de donar el mateix que el hash enviat prèviament.

Però, si aquests usuaris saben molt de criptografia, saben que poden trobar col·lisions. Si B vol fer trampes, pot tenir un nombre parell i un imparell que donin el mateix hash (h(parell) = h(imparell)), de tal manera que quan rebi el valor de A, enviarà el que a ell li interessi. Per solucionar aquest problema es posa un prefix per tal que no es pugui fer precoumputacions. Si A obliga a que el hash de l'altre persona sigui h(prefix + valor), B ja no tindrà tot el temps del món per fer càlculs, perquè no sap el prefix fins que el protocol comença.

5. BLIND SIGNATURES

S'utilitza per signar digitalment missatges de forma especial, on l'usuari A aconsegueix la signatura d'un missatge m per part de l'usuari B sense que B sàpiga quin missatge ha signat.

1. A tria un valor aleatori **r** tal que mcd(r,n) = 1, i el xifra amb la clau pública de B ($\boldsymbol{t} = r^e \bmod n$). A calcularà $\boldsymbol{m'}$ per tapar el missatge original m ($m' = m * t \bmod n$), i li envia a B.
2. B signarà m' amb la seva clau privada ($\boldsymbol{s'} = (m')^d \bmod n$) i li envia a A.
3. Quan A ho rebi, obtindrà la signatura de m calculant $\boldsymbol{s} = \frac{s'}{r}$.

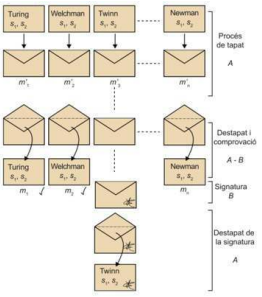
Ona Folch

Exemple:

Pas	Alice		Bob
1.	Tria $25 \in_R \mathbb{Z}_{551}$ t.q. $mcd(15,55)=1$ Calcula $t = 25^{19} = 310 \bmod 551$ Tapat : calcula $m' = 15 \cdot 310 = 242 \bmod 551$	$\xrightarrow{m'=242}$	
2.			Signa el valor $m' = 242$ calculant: $s' = 242^{451} = 14 \bmod 551$
		$\xleftarrow{s'=14}$	
3.	Obté la signatura de m calculant $s = \frac{14}{25} = 14 \cdot 529 = 243 \pmod{551}$		

- B signa m = 15 amb RSA
- (e,n) = (19,551)
- d = 451

Signar alguna cosa sense saber que és pot comportar alguns problemes de seguretat. Per assegurar que B no signa res fraudulent s'utilitza el procés de 'remenar i triar'.

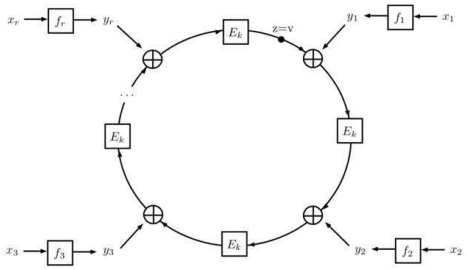


L'usuari A, en comptes d'enviar un únic valor tapat m' a B, calcula múltiples valors tapats $m'_1, m'_2, \cdots, m'_n$. És important que cada valor s'hagi tapat amb un element diferent, és a dir, per a cada m'_i tindrem un valor t_i diferent. Una vegada A ha enviat els n valors tapats a B, B demana a A que destapi n-1 valors, és a dir, A proporcionarà els corresponents t_i per a n-1 valors que B haurà triat aleatòriament. Una vegada destapats, B podrà veure els missatges, i si n-1 missatges són raonables, suposarà que el que ha de firmar també ho és. La probabilitat que A pugui enganyar a B és de 1/n.

6. RING SIGNATURES

En aquest protocol un usuari u_s que pertany a un grup d'usuaris usuari R = $\{u_1, u_2, \cdots, u_r\}$ $(amb\ S \in [1, r])$ signa un missatge m, de manera que un validador pot comprovar que la signatura ha estat realitzada per algun membre del grup R però, alhora, és computacionalment impossible saber quin usuari individual del grup ha realitzat la signatura.

Els usuaris del sistema disposaran d'un parell de claus pública-privada del criptosistema RSA. Per tal de realitzar una signatura d'anell, un usuari seleccionarà un conjunt de claus públiques (que formaran l'anell, el grup de possibles signants del missatge entre els quals hi serà el propi usuari) i generarà una signatura fent servir les claus públiques dels altres membres de l'anell i la seva clau privada. Aquesta signatura podrà ser després validada per un receptor, coneixent el missatge original i les claus públiques dels usuaris de l'anell.



El protocol fa servir una funció de combinació amb clau, que és la base del protocol de signatura d'anell. La funció de combinació rep una clau **k** d'l bits, un valor d'inicialització **v** de b bits, i un número arbitrari d'entrades $\boldsymbol{y_i}$ també de b bits, i retorna una cadena de b bits:

$$C_{k,v}(y_1, y_2, \cdots, y_r) = E_k(y_r \oplus E_k(y_{r-1} \oplus E_k(y_{r-2} \oplus E_k(\cdots \oplus E_k(y_1 \oplus v)\cdots)))) = z$$

Aquesta funció es basa en calcular reiteradament un xor entre dos valors, xifrant-ne després el resultat amb el criptosistema de clau simètrica. La clau **k** que es fa servir com a clau simètrica és el hash del missatge a signar (k = h(m)), i la seqüència d'entrada $(y_1, y_2, \cdots, y_r)$ és $y_i = f_i(x_i)$. Per últim, es força que la sortida **z** hagi de ser igual a **v**.

El procés de realització de signatura s'inicia quan l'usuari que vol realitzar la signatura del missatge m selecciona un conjunt d'usuaris, dels quals en coneix la clau pública, per formar part de l'anell de possibles signants R. És a dir, el signant obté r claus públiques $\{PK_i = (e_i, n_i), i \in \{1, \cdots, r\}\}$. Amb aquesta informació i els seu propi parell de claus (SK_s, PK_s) (fem servir l'índex s per referir-nos al signant) executa els següents passos:

1. El signant calcula els següents valors:
   - $k = h(m)$
   - b tal que $2^b > n_i$ per a $1 \leq i \leq r$
   - $v \in_R \{0,1\}^b$
   - $x_i \in_R \{0,1\}^b$ per a $1 \leq i \leq r$, per a $i \neq s$
   - $y_i = f_i(x_i)$ per a $1 \leq i \leq r$, per a $i \neq s$

Ona Folch

2. Troba el valor y_s que soluciona l'equació $C_{k,v}(y_1,y_2,\cdots,y_r) = v$
3. Calcula $x_s = f_s^{-1}(y_s)$
4. Calcula la signatura del missatge m, que és $\sigma = \{PK_1,\cdots,PK_r, v, x_1,\cdots,x_r\}$

Per validar la signatura, el verificador necessita σ i el missatge m sobre el que s'ha realitzat la signatura.

1. Calcula
   - $y_i = f_i(x_i)$ per a $1 \leq i \leq r$
   - $k = h(m)$
2. Verifica que es compleixi la igualtat $C_{k,v}(y_1,y_2,\cdots,y_r) = v$

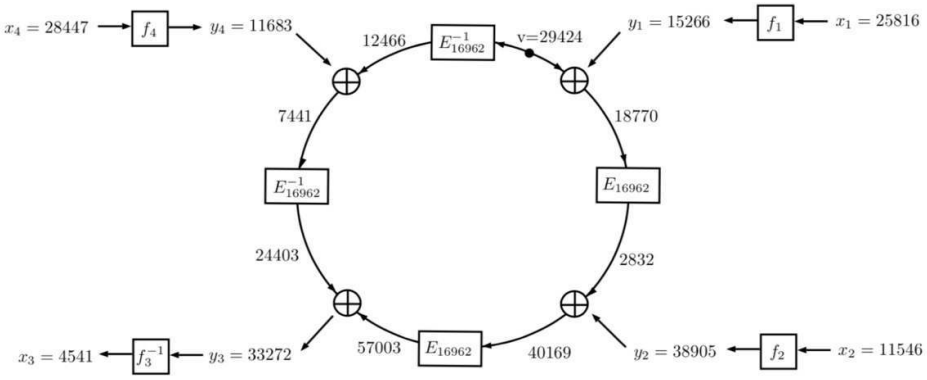
Exemple:

La funció hash serà la funció identitat (h(x) = x), i com a funció de xifrat simètric farem un xor ($E_k(x) = x \oplus k$). Claus de cada un dels usuaris:

$$PK_1 = (28907, 18541) \qquad PK_2 = (41917, 22491 \qquad PK_3 = (39407, 26077) \qquad PK_4 = (32743, 17539)$$

L'usuari que fa la signatura és el 3, i la seva clau privada és $SK_3 = (39407, 27013)$.

1. L'usuari 3 calcula:
   - $k = h(16962) = 16969$
   - $b = 16$, ja que $2^{16} = 65536 > n_i$ per a $1 \leq i \leq 4$
   - $v = 29424 \in_R \{0,1\}^{16}$
   - $x = \{25816, 11546, \emptyset, 28447\}$ amb $x_i \in_R \{0,1\}^{16}$
   - $y_1 = f_1(25816) = 25816^{18541} \ (mod\ 28907) = 15266$
   - $y_2 = f_2(11546) = 11546^{22491} \ (mod\ 41917) = 38905$
   - $y_4 = f_4(28447) = 28447^{17539} \ (mod\ 32743) = 11683$
2. Troba el valor y_s que soluciona l'equació $C_{k,v}(y_1,y_2,\cdots,y_r) = C_{16962,29424}(15266, 38905, y_3, 11683) = 29424$
   $y_3 = 33272$
3. Calcula $x_3 = f_3^{-1}(33272) = 33272^{27013} \ (mod\ 39407) = 4541$
4. Calcula la signatura del missatge m, que és $\sigma = \{PK_1,PK_2,PK_3,PK_4,v,x_1,x_2,x_3,x_4\} =$
   $= \{(28907, 18541), (41917, 22491), (39407, 26077), (32743, 17539), 29424, 25816, 11546, 4541, 28447\}$



El verificador, per verificar la signatura realitzarà els següents passos:

1. Calcula
   - $y_1 = f_1(25816) = 25816^{18541} \ (mod\ 28907) = 15266$
   - $y_2 = f_2(11546) = 11546^{22491} \ (mod\ 41917) = 38905$
   - $y_3 = f_2(4541) = 4541^{26077} \ (mod\ 39407) = 33272$
   - $y_4 = f_4(28447) = 28447^{17539} \ (mod\ 32743) = 11683$
   - $k = h(16962) = 19692$
2. Verifica que es compleixi la igualtat $C_{k,v}(y_1,y_2,\cdots,y_r) = C_{19692,29424}(15266, 38905, 33272, 11683) = v = 29424$

Ona Folch