Elements de criptografia

Llorenç Huguet Rotger
Josep Rifà Coma
Juan Gabriel Tena Ayuso

PID_00185089

UOC
Universitat Oberta de Catalunya

Índex

Introducció

Tradicionalment, la criptografia té com a objectiu la transmissió o emmagatzemament de missatges indesxifrables per a tot receptor que no disposi de la clau de l'algorisme de desxifratge.

Lectura recomanada

Per a fer més comprensible aquest mòdul didàctic podeu resseguir el llibre de criptografia de J. Domingo, J. Herrera i H. Rifà-Pous dels estudis d'Informàtica i Multimèdia, de la UOC

Avui, la criptografia es presenta com la solució al problema de la vulnerabilitat dels sistemes de transmissió, o d'emmagatzematge, pel que fa al secret i a l'autenticitat de la informació transmesa, o emmagatzemada. L'objectiu pel que fa la privacitat i autenticitat associats a una xarxa de sistemes és evitar que un espia pugui violar o eliminar la protecció del sistema en referència a les línies de comunicació, a la connexió d'accés a la xarxa (paraules de pas) i a la utilització dels recursos d'un determinat sistema.

Fa temps, la criptografia era una activitat quasi exclusivament utilitzada en la diplomàcia i en la guerra, però, a partir de la Segona Guerra Mundial, l'aparició dels ordinadors ha fet que tots els sistemes criptogràfics utilitzats abans, excepte el mètode de Vernam (basat en claus d'un sol ús i del qual es pot demostrar matemàticament la inviolabilitat), formin part de la historia, ja que la velocitat en el tractament de la informació fa que sigui un joc de nens el problema de trobar les seves claus corresponents (criptoanàlisi).

D'aquesta simplicitat dels mètodes clàssics n'és exemple el sistema criptogràfic, anomenat de Juli Cèsar, per ser-ne ell el primer usuari, utilitzat encara durant la Segona Guerra Mundial, que consistia a numerar els caràcters alfabètics i xifrar el missatge m en el criptograma c, mitjançant una translació cíclica que avui enunciaríem com $c = (m + k) \pmod{25}$, en què m és el valor numèric assignat a cada lletra de l'alfabet $\{A = 0, B = 1, \ldots, Z = 24\}$, per exemple, i per a un cert valor de k prèviament triat (Cèsar agafava $k = 3$). El text AMOR quedaria amb valors numèrics 0,12,14,17 que es xifrarien en 3,15,17,20; és a dir, es transmetria el missatge xifrat (criptograma): DPRU.

De sempre, tota tècnica criptogràfica que opera sobre un missatge, sense tenir en compte la seva estructura lingüística, està basada en una operació executada per l'emissor, que transforma el missatge original en un missatge xifrat, mitjançant un algorisme que implementa aquesta operació, lligada a una clau k. Al mateix temps, aquesta operació posseeix l'operació inversa, executada pel receptor, que permet trobar el missatge original.

Tot sistema criptogràfic, també anomenat criptosistema, consta de cinc components: $\{M, C, K, E$ i $D\}$, en què M és el conjunt de tots els missatges per transmetre, C el de tots els missatges xifrats, K el de les claus per utilitzar, E el de

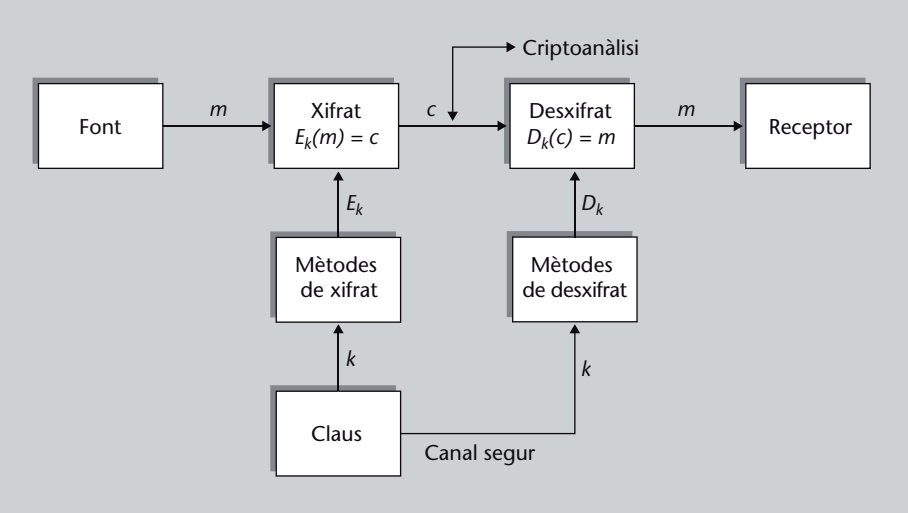
tots els mètodes de xifratge $E = \{E_k | E_k(m) \in C, \forall m \in M, \forall k \in K\}$ i D el de tots els mètodes de desxifratge $D = \{D_k | D_k(c) \in M, \forall c \in C, \forall k \in K\}$.

Cada mètode de xifratge d'E, i cada mètode de desxifratge de D, està definit mitjançant un algorisme, el qual és comú a tots els mètodes, i en què cada clau $k \in K$, distingirà l'instància corresponent a cada transformació E_k, i D_k, respectivament.

Per a tota clau $k \in K$, la transformació D_k és la inversa de E_k; és a dir:

$$D_k(E_k(m)) = m, \forall m \in M$$

Noteu, però, que això no vol dir que E_k sigui l'invers de D_k, en sentit matemàtic.



Tot criptosistema, tal com mostra la figura anterior, ha de complir almenys aquests tres requisits:

1) Tots els algorismes de xifratge i desxifratge E_k i D_k han de ser computacionalment eficients.

2) Els algorismes E_k i D_k han de ser fàcilment implementables.

3) La seguretat del sistema només ha de dependre del secret de les claus $k \in K$, i no dels algorismes corresponents de E i D.

Principis de Kerchoff

En criptografia, les propietats desitjables d'un criptosistema constitueixen els principis de Kerckhoff; els més importants són: si el criptosistema no és teòricament irrompible, almenys ho ha de ser en la pràctica. L'efectivitat del criptosistema no ha de dependre que el seu disseny romangui sota secret. El criptosistema ha de ser fàcil d'emprar. La clau ha de ser fàcilment memoritzable, per a evitar haver de recórrer a notes escrites. Els criptogrames hauran de ser alfanumèrics.

A més, sempre s'hauran de tenir en compte els objectius de privacitat i autenticitat, en què es considera:

- **Privacitat**: com la incapacitat, per a un criptoanalista, de determinar un missatge original a partir del criptograma que hagi pogut interceptar.

- **Autenticitat**: com la incapacitat, per a un criptoanalista, de substituir un criptograma fals c', en lloc del criptograma real c, sense que sigui detectat.

Consideracions als criptosistemes

Actualment, es consideren dos tipus de criptosistemes, segons la utilització i administració dels algorismes de xifratge i desxifratge. El criptosistema clàssic o convencional, en el qual la clau corresponent a tots dos algorismes és la mateixa o, en el seu defecte, una fàcilment deduïble de l'altra. En aquest cas, cada usuari disposa de la seva parella d'algorismes E_k i D_k particulars i cap altre usuari no en pot disposar sense conèixer la clau k. Són els anomenats **criptosistemes de clau privada o simètrics**.

L'any 1976 entra en joc un nou concepte de criptosistema, proposat per W. Diffie i M. Hellman, anomenats **criptosistemes de clau clau pública o asimètrics**, caracteritzats pel fet que conèixer l'algorisme E_k no revela cap informació sobre D_k o viceversa. És a dir, una de les claus pot ser revelada públicament sense perill que l'altra pugui ser deduïda.

Lectura recomanada

W. Diffie; M. Hellman. "New Directions in Criptography". *IEEE Transactions on Information Theory* (vol. IT-22).

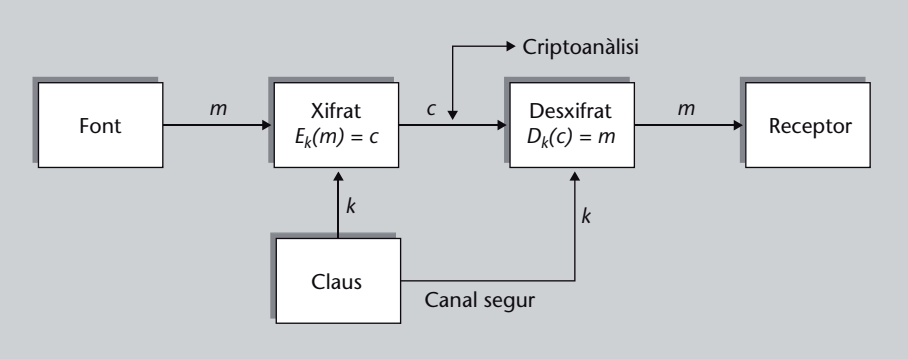
Objectius

En els materials didàctics d'aquest mòdul l'estudiant trobarà els continguts necessaris per a assolir els objectius següents:

1. Conèixer els sistemes criptogràfics de clau simètrica més comuns (DES, IDEA, AES).
2. Conèixer els sistemes criptogràfics de clau asimètrica més comuns (RSA, ElGamal).
3. Conèixer els principals algorismes usats en les funcions hash (MD5, SHA).
4. Conèixer els fonaments i algun exemple de sistema criptogràfic quàntic.
5. Conèixer els fonaments de la teoria de la codificació per a la correcció d'errors i els sistemes criptogràfics postquàntics basats en aquesta (McEliece, Niederreiter).

1. Criptosistemes simètrics o de clau privada

Ens referirem als criptosistemes simètrics o de clau privada quan l'emissor i el receptor comparteixen una única clau *k*. Per això, establim com a característica principal l'existència d'un canal segur per mitjà del qual l'emissor transmet al legítim receptor la seva clau privada *k* de manera que queda protegida davant d'un criptoanalista.



L'emissor vol enviar xifrat el missatge *m*, per la qual cosa, mitjançant l'algorisme de xifratge calcula el criptograma *c* a partir de *m* i de la clau *k*:

$$E_k(m) = c$$

El receptor ha de tenir la capacitat de desxifrar el criptograma *c*, a partir del coneixement de la clau *k*, és a dir, retrobar el missatge *m* mitjançant:

$$D_k(c) = m$$

1.1. Criptosistema DES

L'any 1977 l'NBS National Bureau of Standards, dels EUA, anuncià un algorisme de xifratge estàndard, el DES (*data encryption standard*, FIPS pub. 46, National Bureau of Standards (gener 1977)), perquè fos utilitzat per totes les agències federals, amb el propòsit que fossin compatibles tots els sistemes de protecció d'informació utilitzats en els diferents estats, sota un sistema criptogràfic comú admès com a estàndard. La nova agència que va substituir la NBS, el NIST (National Institute of Standards and Technology) va certificar el

DES l'any 1987 i després el 1993, fins que el 1997 ja no el va certificar. Durant aquests anys fou considerat estàndard a escala mundial i avui encara és utilitzat en l'intercanvi d'informació entre els caixers automàtics i els bancs respectius. Ni que sigui per motius històrics ens sembla interessant fer-ne referència.

El DES consisteix en un algorisme de xifratge-desxifratge de blocs de 64 bits, mitjançant una clau k, també de 64 bits (dels quals només 56 bits són efectius).

Els 64 bits d'entrada (missatge original) es transformen per mitjà d'una permutació inicial PI, la sortida de la qual es divideix en dos subblocs L_0 i R_0 de 32 bits cada un, els quals estan subjectes a un conjunt de 16 transformacions, d'acord amb una certa funció f i 16 subclaus k_i ($(i = 1, \ldots, 16)$). Després de fer les 16 transformacions s'ajunten els subblocs R_{16} i L_{16} i se li aplica la inversa de la permutació inicial: PI^{-1}.

Enllaç d'interès

Es pot trobar un programa de simulació del DES, d'ús lliure, a l'adreça: www.criptored.upm.es

Si T_i és el resultat de la i-èsima iteració, aleshores T_i està formada per dues parts, la L_i, que denota els 32 bits de més a l'esquerra de T_i, i R_i, que denota els 32 bits de més a la dreta de T_i; això és, T_i és la concatenació de L_i amb R_i. El càlcul es fa d'aquesta manera:

$$L_i = R_{i-1},$$

$$R_i = L_{-1} \oplus f(R_{i-1}, k_i)$$

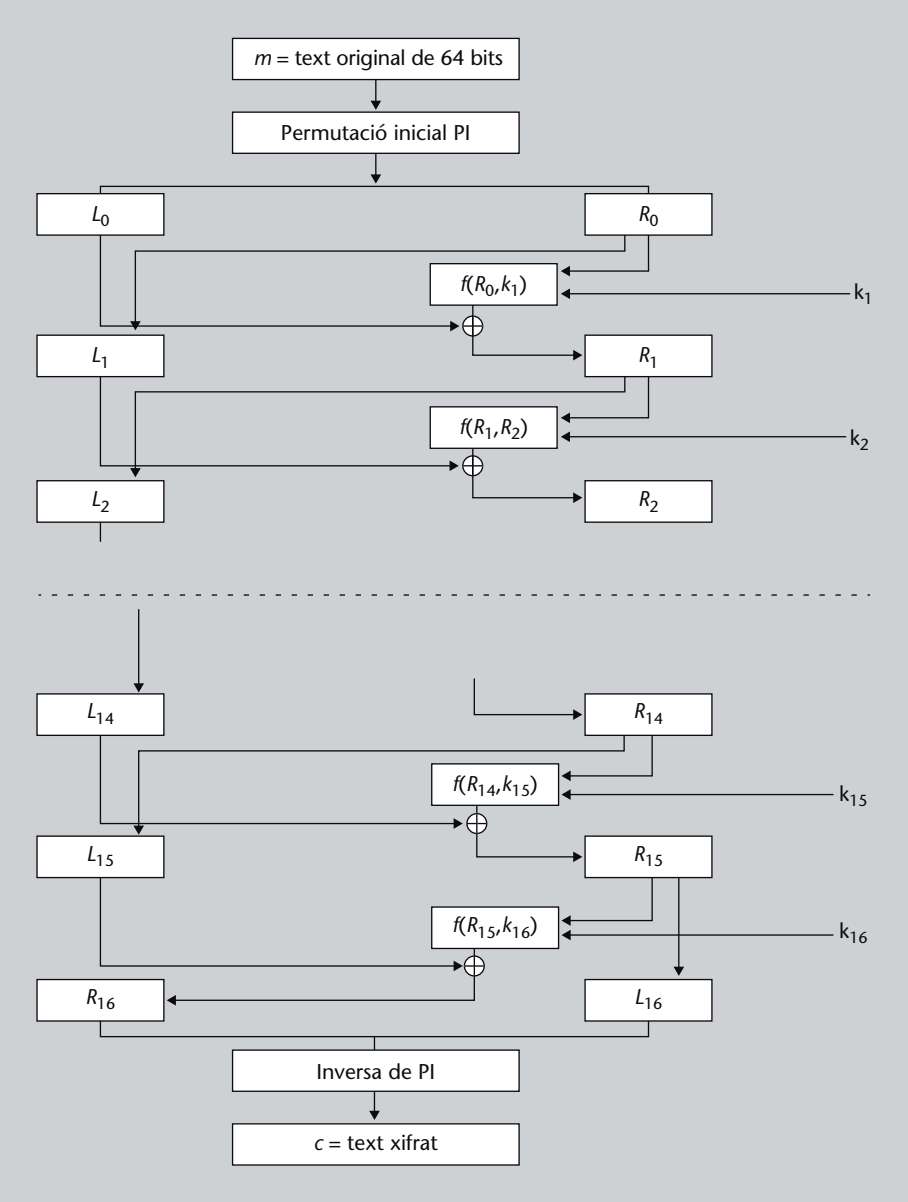
en què $\oplus$ és l'operació *or* exclusiva i k_i és una subclau de 48 bits obtinguda a partir de la clau original k.

La funció f transforma els 32 bits del bloc R_{i-1}, mitjançant la subclau k_i en els 32 bits del bloc R_i. Per fer això, primer s'expandeixen els 32 bits de R_{i-1} en un bloc de 48 bits, utilitzant una taula d'expansió E per tal de calcular la *or* exclusiva de $E(R_{i-1})$ i k_i, el resultat de la qual es divideix en vuit blocs B_i de 6 bits d'entrada i 4 de sortida. Aquests bits de sortida són concatenats per a donar un nou bloc de 32 bits. Finalment, la sortida de la funció f és el resultat d'aplicar una certa permutació P al bloc de 32 bits anterior. És a dir:

$$f(R_{i-1}, k_i) = P(S_1(B_1), S_2(B_2) \ldots S_8(B_8))$$

Cada una de les 16 iteracions de l'algorisme DES utilitza una clau diferent de 48 bits, k_i, calculada a partir de la clau k de 64 bits, la qual posseeix 8 bits de control en les posicions 8,16,24,32,40,48,56,63, mitjançant una permutació P_1 de 56 bits. El resultat $P_1(k)$ es divideix en dues parts de 28 bits cada un, les quals sofreixen un cert desplaçament a l'esquerra diferent per a cada subclau k_i.

La figura següent detalla els passos de l'algorisme de xifratge DES.



Nota

El fet que no siguin intercanviats R_{16} i L_{16} abans d'aplicar-los la permutació PI^{-1} obeeix al fet que l'algorisme que descrivim també s'utilitzarà per al desxifratge.

Per a augmentar la fortalesa del DES davant de possibles criptoanàlisis es va proposar la reiteració dels processos de xifratge/desxifratge operant, successivament, sobre el mateix bloc amb diverses claus independents. Aquest és el cas del doble o triple xifratge.

1) Doble xifratge. Donades les claus independents k_1 i k_2, els algorismes de xifratge i desxifratge estan determinats per:

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

$$m = D_{k_1}(D_{k_2}(c))$$

2) Triple xifratge. Donades les claus independents k_1, k_2 i k_3, els algorismes de xifratge i desxifratge estan determinats per:

$$c = E_{k_3}(D_{k_2}(E_{k_1}(m)))$$
$$m = D_{k_1}(E_{k_2}(D_{k_3}(c))$$

El mètode de triple xifratge pot ser utilitzat per a evitar l'atac criptoanalític del *meet in the middle*.

1.2. Criptosistema IDEA

L'*international data encryption algorithm* és un criptosistema de clau simètrica, el qual fou dissenyat per enfortir les debilitats detectades en el DES. Avui dia és utilitzat en correu electrònic segur (PGP)

L'algorisme IDEA opera sobre blocs de 64 bits com a text original, i dóna un text xifrat de 64 bits, mitjançant una clau de 128 bits. És a dir, opera sobre la mateixa longitud de blocs de bits que el DES, però amb una longitud de clau doble, la qual cosa afegeix complexitat a la criptoanàlisi.

Les operacions es fan sobre subblocs de 16 bits, cosa que en permet l'adaptació a arquitectures de 16 bits. En l'algorisme hi ha tres tipus d'operacions: l'*or* exclusiva $\oplus$, la suma mòdul 2^{16} i el producte mòdul $2^{16}+1$ (aquest valor $2^{16}+1$ és un nombre primer, per la qual cosa podem considerar inversos dins el cos finit associat).

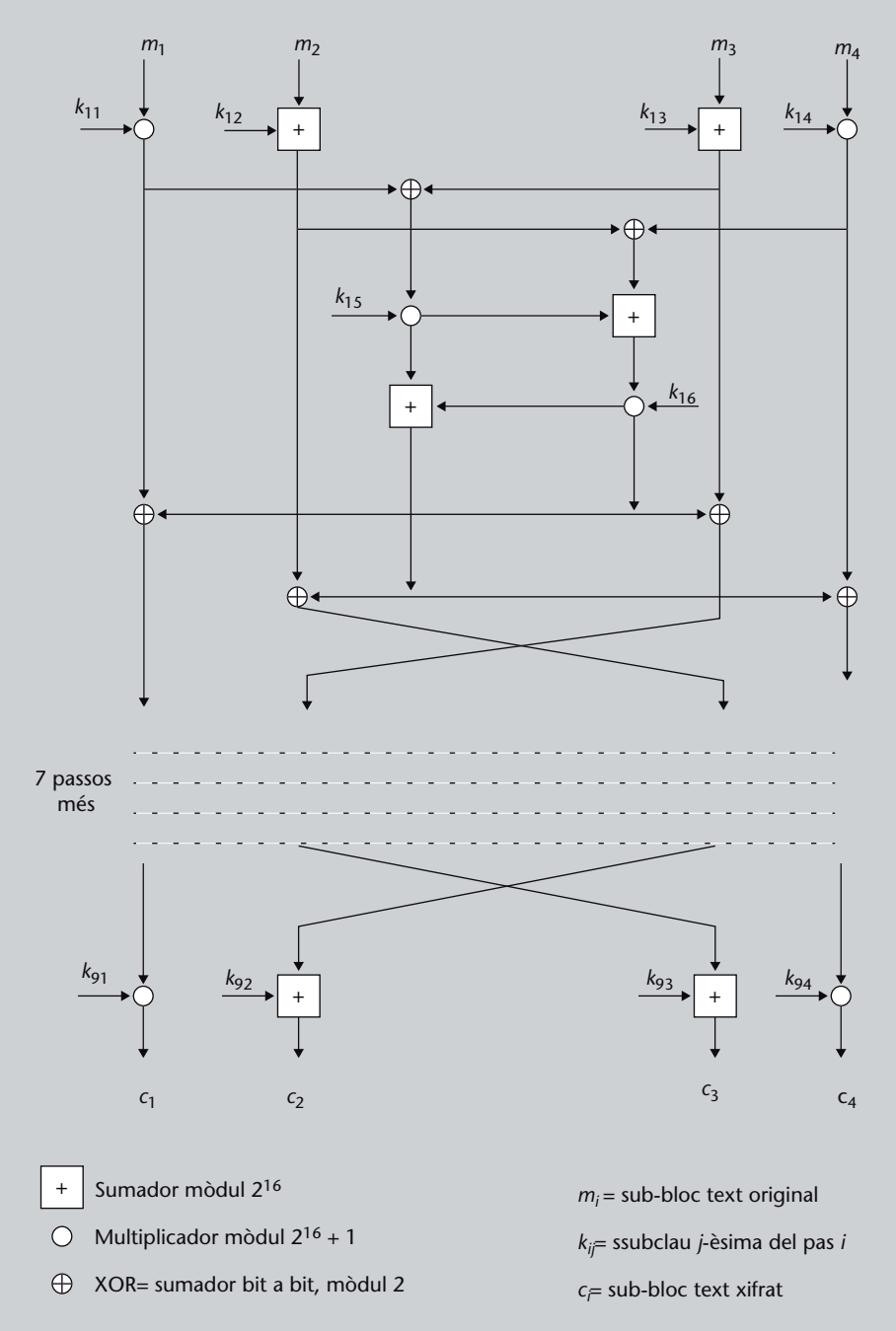
Enllaç d'interès

Es pot trobar un programa de simulació del IDEA, d'ús lliure, a l'adreça: www.criptred.upm.es

Sigui m un bloc de 64 bits per xifrar; aquest es dividirà en 4 blocs de 16 bits, m_1, m_2, m_3, m_4, que seran l'entrada a l'algorisme de xifratge. L'algorisme consta de vuit passos amb les mateixes característiques. A cada pas, intervenen els quatre subblocs del text i sis subclaus, també de 16 bits, de manera que entre pas i pas els blocs segon i tercer s'intercanvien. Per acabar, hi ha un novè pas, en el qual intervenen els quatre subblocs de text i només quatre subclaus.

La clau de 128 bits se subdivideix en vuit subblocs de 16 bits cadascun, els quals constitueixen les primeres vuit subclaus utilitzades per l'algorisme: sis d'aquestes en el primer pas, i les dues restants són les dues primeres del segon pas. Per poder continuar, a la clau inicial se li aplica una rotació de 25 bits a l'esquerra i la nova clau resultant se subdivideix en vuit subblocs de 16 bits. Ara l'algorisme utilitzarà les dues subclaus anteriors seguides de les quatre primeres subclaus provinents de la subdivisió actual. Les quatre restants s'utilitzaran en el tercer pas, i així successivament

El diagrama de blocs del criptosistema IDEA es mostra a la figura següent.



1.3. Criptosistema AES

Durant el període que el DES va estar en vigor (1977-2001) es van proposar desenes de xifratges alternatius, molts de domini públic. L'NSA (National Security Agency) també va donar a conèixer el 1994 un nou xifratge per a ser usat en telefonia i comerç electrònic: Skipjack.

Aquest algorisme de xifratge, en blocs de 64 bits, va ser declarat secret, i els xips que s'han implementat impedeixen accedir al codi font. Potser fa així l'agència perquè no vol que un dels seus criptosistemes circuli obertament per tot el món, o per a poder escoltar converses telefòniques xifrades, o totes dues coses.

Enllaç d'interès

Es pot trobar un programa de simulació de l'AES, d'ús lliure, anomenat *AES Inspector* a l'adreça: www.formaestudio.com/rijndaelinspector

Al gener de 1997, el NIST (National Institute of Standards and Technology), veient que la seguretat del DES estava ja compromesa (per motius estrictament computacionals, perquè el DES mai no ha estat trencat), va convocar a concurs públic l'adjudicació del nou estàndard de xifratge. Es denominaria AES, acrònim d'*advanced encryption standard*. En la convocatòria s'especificava una sèrie de requisits mínims: un xifratge en bloc de 128 bits; amb claus de 128, 192 i 256 bits; la possibilitat de ser implementat tant en maquinari com en programari i estar disponible gratuïtament. Però el més destacat de la convocatòria era que el concurs estava obert a tothom i que el procés de selecció seria totalment transparent.

Amb aquestes idees es va iniciar una etapa llarga de selecció que va culminar, l'any 2000, en l'elecció del criptosistema Rijndael dels investigadors belgues Vincent Rijmen i Joan Daemen.

Des d'aquest moment, el DES ja té substitut: AES. Naturalment, la decisió del NIST d'adoptar el nou criptosistema només obliga a l'administració federal nor-americana, i pel que fa a la informació no classificada; però amb tota seguretat, l'AES serà el xifratge més usat en els pròxims anys. La recomanació del xifratge per part del NIST és tot un certificat de garantia per a empreses i organitzacions. Prova d'això és que l'NSA mateixa ha aprovat l'ús d'AES per a xifrar informació classificada: la **secreta** amb claus de 128 bits i la **d'alt secret** amb claus de 192 i 256 bits.

El procés de xifratge de cada bloc de 128 bits de text original conté tres transformacions, o capes diferents en què es tracten els bits, que consten del següent:

- **Capa de mescla lineal**: difusió dels bits: *ShiftRow* i *MixColumns*.
- **Capa no lineal**: *ByteSub* (similar a les *S-boxes* del DES).
- **Capa d'addició de clau**: operacions amb la funció *or* exclusiva entre l'estat intermedi i la subclau de cada ronda.

Les operacions implicades en AES s'expressen en termes algebraics emprant certa aritmètica de bytes. En aquesta aritmètica, la suma i el producte de bytes són justament la suma i el producte en el cos finit $\mathbb{F}_{2^8}$, construït a partir del polinomi primitiu: $p(X) = X^8 + X^4 + X^3 + X + 1$.

Criptoanàlisi

Disposar d'un estàndard de xifratge d'ús generalitzat té un gran inconvenient: que tothom el vol trencar. En exigir claus amb una longitud mínima de 128 bits, el NIST dotava al seu xifratge d'una seguretat més que suficient contra un atac per força bruta; l'únic mètode que va vèncer l'anterior estàndard, el DES. Descobrir una clau de 128 bits comprovant una per una totes les possibles és una tasca pràcticament eterna per a qualsevol ordinador d'avui dia. Fins i tot per a tots junts. I 256 bits són garantia de sobres contra tots els ordinadors electrònics que es construeixin en les pròximes dècades. Serà necessari quelcom més que força bruta per a derrotar el AES. Durant el concurs, el vencedor Rijndael (igual que la resta dels finalistes) va provar ser immune a tots els mètodes de criptoanàlisi coneguts fins aquell moment.

1.4. Modes d'operació dels criptosistemes de clau privada

Per a augmentar la seguretat dels algorismes de xifratge i desxifratge s'utilitzen de diversos modes:

- **Xifratge en bloc**: el text original es processa en blocs disjunts de 64 bits, els blocs de sortida dels quals, també de 64 bits, es concatenen per a formar el text xifrat. Aquest mode se sol denotar **ECB** (*electronic code-book*). Aquesta manera de xifrar/desxifrar impedeix, d'una banda, la supressió o inserció de blocs de text xifrat, perquè en qualsevol cas el receptor seria incapaç de desxifrar el criptograma rebut i, per tant, quedaria alertat de les possibles intrusions. D'altra banda, els atacs estadístics també s'han complicat, per la interdependència del text xifrat al llarg de tot el procés. Una alternativa és l'anomenat *xifratge en blocs encadenats*, consistent a dividir el text per xifrar en blocs i fer dependre el bloc *n*-èsim de text xifrat/desxifrat del bloc $(n-1)$-èsim. És a dir:

$$c_n = E_k(m_n \oplus c_{n-1})$$

$$m_n = D_k(c_n) \oplus c_{n-1}$$

  El primer bloc d'entrada al procés de xifratge està format per l'*or* exclusiva entre el primer bloc del missatge i els 64 bits del vector inicial $c_0 = VI$, el qual és compartit pels algorismes de xifratge i de desxifratge. Aquest mode se sol denotar **CBC**(*cipher block chaining*).

- **Xifratge en flux**: operant sobre un o més bits, desplaçant prèviament un conjunt de bits de l'operació anterior en un nombre suficient per a desar els nous (*stream cipher*).

  Dins d'aquest mode de xifratge, es considera el mode **CFB** *cipher feedback*, que consisteix en una *or* exclusiva entre els *n* bits més a l'esquerra de la informació de sortida del procés de xifratge i els *n* bits de la informació d'entrada produint *n* bits de text de xifratge (*n* és el nombre de bits per xifrar). El criptograma s'obté a partir d'un valor inicial *VI* i del criptograma anterior. Per a carregar aquests *n* bits en el dispositiu de xifratge, desplaçarem el contingut del valor inicial *VI n* bits a l'esquerra.

  Aquesta modalitat permet el tractament de blocs de menys de 64 bits. Normalment s'utilitza per a la seguretat de missatges molt repetitius i per a xifrar/desxifrar fitxers en què no convé emmagatzemar informació inútil.

  Hi ha altres mètodes, implementats amb registres de desplaçament; LFSR (*linear feedback shift register*), en els quals la clau *k* s'usa per a controlar un generador de claus variables (*running key generator*), que produeix una seqüència binària $k_1,k_2...k_n$ (*n* ha de ser molt més gran que la longitud de la clau). Així, els dígits del text xifrat es formen a partir del text original, en binari:

$$c_i = m_i \oplus k_i$$

  Evidentment, el desxifratge es farà de manera simètrica: $m_i = c_i \oplus k_i$

Secret perfecte de Shannon

El fet que els registres de desplaçament que generen la seqüència binària $k_1,k_2...k_n$ tinguin un període finit està en contraposició dels requisits del secret perfecte de Shannon, perquè, si el text per xifrar és molt llarg, es repetirà la clau de manera determinista.

2. Criptosistemes de clau pública

L'any 1976 entra en joc el nou concepte de criptosistema, proposat per Diffie i Hellman, amb dues claus, una de les quals pot ser revelada públicament sense perill que se'n pugui deduir l'altra.

En un criptosistema amb claus públiques cada usuari té un algorisme de xifratge E_k, registrat en un directori públic, i un algorisme de desxifratge D_k que només coneix l'usuari. Mentre que D_k es defineix en funció de la clau privada, E_k es defineix mitjançant un algorisme o funció que no permeti, des del punt de vista computacional, la revelació de D_k. Aquests tipus d'algorismes o funcions reben el nom d'*unidireccionals*.

Separabilitat escriptura/lectura

Aquest tipus de criptosistemes són molt indicats per a la protecció de fitxers públics, ja que el fet de poder escriure informació sobre el fitxer no implica poder-la llegir i viceversa, perquè les claus d'escriptura i lectura són independents, malgrat que estan relacionades.

En aquest cas, dos algorismes diferents proporcionen el secret i l'autenticitat. Per exemple, si l'usuari A vol transmetre el missatge m a l'usuari B, tots dos connectats al mateix directori públic, només ha de cercar l'algorisme de xifratge E_{k_B} de B en el directori públic i transmetre el missatge xifrat $c = E_{k_B}(m)$. Quan l'usuari B rep c ha d'aplicar D_{k_B}, que només ell coneix, i troba el missatge original $D_{k_B}(c) = m$ (ja que $D_{k_B}(c) = D_{k_B}(E_{k_B}(m)) = m$).

En aquest cas, queda assegurat el secret, però no en queda protegida l'autenticitat, ja que tots els usuaris poden conèixer E_{k_B}.

Per a poder aconseguir l'autenticitat haurem d'exigir a les transformacions E_k i D_k, de cada usuari, que verifiquin que E_k és la transformació inversa de D_k; és a dir, per a tot missatge m i per a tot usuari amb clau k, aleshores: $E_k(D_k(m)) = m$.

Signatura digital

En general, s'envia la signatura per una banda i el missatge, xifrat o no, per una altra banda; segons la necessitat de privacitat del missatge. Així, si A vol enviar un missatge signat a B, enviarà m, o $E_{k_B}(m)$ i la signatura corresponent: $s = D_{k_A}(m)$. Aleshores, B recupera m amb la seva clau privada, que compara amb el resultat d'aplicar el xifratge amb la clau pública de A a s. Si tots dos resultats coincideixen s'accepta l'autenticació i en cas contrari es rebutja. Fins i tot, tal com veurem més endavant, el que s'enviarà és el missatge m, xifrat o no, i la signatura d'un resum del missatge: $h(m)$ (funció resum). En tal cas la signatura serà $s = D_{k_A}(h(m))$ (vegeu l'algorisme de signatura DSA).

En aquest cas l'usuari A podrà signar els seus missatges mitjançant la seva transformació secreta D_{k_A}. En efecte, si A vol enviar el missatge m a B, autenticant la seva procedència, podrà signar digitalment aquest missatge fent $s = D_{k_A}(m)$, en què s serà la signatura i D_{k_A} la transformació de desxifratge de l'usuari A, que només ell coneix; per tant, només ell podrà fer aquesta operació.

Ara, un cop signat el missatge, A enviarà el missatge xifrat $c = E_{k_B}(s)$. Quan B rep c pot trobar s, ja que $s = D_{k_B}(c)$, que serà inintel·ligible per a ell. Com que ha estat previngut que A li enviava un missatge, per a desxifrar el contingut de s només ha de cercar en el directori públic l'algorisme E_{k_A} d'A. I en efecte, $E_{k_A}(s) = m$ era el missatge que li pretenia transmetre A.

2.1. Funcions unidireccionals

La idea fonamental de Diffie i Hellman per a la definició de criptosistema amb clau pública és l'existència de les funcions unidireccionals (*one way functions*).

Definició 2.1 (Funció unidireccional).

Una funció f sobre un domini U es diu *unidireccional* si $\forall x \in U$, $f(x)$ és fàcilment calculable, mentre que per a quasi tots els $y \in f(U)$, no és computacionalment factible trobar $x \in U$, tal que $f(x) = y$.

Noteu que la definició no és precisa: els termes *fàcilment calculable*, *quasi per a tot* i *computacionalment factible* són molt imprecisos, tot i que es poden definir matemàticament de manera que tinguin un sentit clar i perfectament precís.

Definició 2.2 (Funció unidireccional amb trapa).

Una família de funcions invertibles f_k amb domini U_k, amb índex k, es diu funció *unidireccional amb trapa* si, donat k, és poden trobar algorismes E_k i D_k que calculin fàcilment $f_k(x)$ i $f_k^{-1}(y)$ $\forall x \in U_k$ i $\forall y \in f(U_k)$; i, no obstant això, quasi per a tot k i $\forall y \in f(U_k)$, no és computacionalment eficient trobar $f_k^{-1}(y)$, només coneixent E_k.

Una de les primeres candidates a funció unidireccional fou la del logaritme discret, proposada pels mateixos Diffie i Hellman. En efecte, donats un nombre primer gran, p, i α un element primitiu del cos $\mathbb{F}_p$, la funció exponencial discreta: $f(x) = \alpha^x \pmod p$, en què $1 < x < p$ és computacionalment eficient calcular-la.

En canvi, la inversa de l'exponencial discreta; el logaritme discret $x = log_\alpha(y)$ no és computacionalment eficient calcular-lo si p–1 té un factor primer gran.

Algorisme de multiplicar i elevar

Per a valors de x grans, podem usar el mètode binari d'exponenciació (**D. E. Knuth.** (1981). *The Art of Computer Programming. Vol. 2: Semi-Numerical Algorithms*. Addison Wesley).
Per exemple (vegeu l'algorisme 3.2 del mòdul "Cossos finits", calcular α^{25} es pot fer d'aquesta manera: $\alpha^{25} = \alpha^{16+8+1} = (((\alpha^2)^2)^2)^2 \cdot ((\alpha^2)^2)^2 \cdot \alpha$

2.2. Criptosistema RSA

A partir del concepte de funció unidireccional, Diffie i Hellman defineixen l'estructura d'un criptosistema de clau pública. No obstant això, no van proporcionar cap implementació concreta del tal estructura, tret de cas del protocol de distribució de claus privades que veurem més endavant.

R. L. Rivest, A. Shamir i L. Adleman, del MIT, en el seu article "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Comm. of ACM (núm. 21, vol. 2, pàg. 120-126, febrer de 1978), presentaren un criptosistema

de clau pública que complia totes les condicions enumerades anteriorment (el criptosistema RSA), basat en el teorema d'Euler i en la dificultat de factoritzar un valor $n = p \cdot q$, en què p i q són primers.

La funció unidireccional de l'RSA és l'exponencial discreta: $f_k(x) = x^e \pmod n$; en què $0 < x < n = p \cdot q$ i en què $k = (e,n)$; p i q són dos primers molt grans i e compleix $0 < e < \varphi(n)$ i $mcd(e,\varphi(n)) = 1$. L'algorisme E_k per a calcular $f_k(x)$ és relativament fàcil; és l'exponenciació pel mètode de multiplicar i elevar esmentat. Fer públic aquest algorisme requereix divulgar n i e.

Càlcul de $\varphi(n)$

Per a valors grans de p i q, no és computacionalment eficient el càlcul de $\varphi(n)$, per a qui no coneix els valors de p i q.

La funció inversa és:

$$f_k^{-1}(y) = y^d \pmod n$$

Teorema d'Euler

El teorema d'Euler assegura que la funció f_k^{-1} és la inversa de f_k, és a dir: $f_k^{-1}(f_k(x)) = x \pmod n$, si x és relativament primer amb n.

en què d és l'únic $0 < d < n$ tal que $e \cdot d = 1 \pmod{\varphi(n)}$. L'algorisme f_k^{-1} és fàcil de calcular per a qui coneix la clau (d,n). Però només coneixerà d qui conegui $\varphi(n)$, difícilment calculable per a qui no coneix la factorització de n en p i q (aquesta és la trapa de la funció unidireccional utilitzada).

Trapa

Donat $\varphi(n)$ és fàcil generar el parell de nombres e i d que satisfan la condició d'inversos $(\mathrm{mod}\ \varphi(n))$, quan e o d són relativament primers amb $\varphi(n)$. És a dir, donat e, és fàcil calcular d (o viceversa) si coneixem $\varphi(n)$. No obstant això, si e i n són coneguts, sense revelar $\varphi(n)$, no és computacionalment eficient calcular d.

2.2.1. Descripció del criptosistema RSA

El criptosistema RSA consisteix a associar a cada caràcter de l'alfabet, en què estan escrits els missatges originals, un valor numèric i aleshores xifrar el missatge per blocs de la mateixa longitud i amb un valor numèric comprès en un cert rang.

Suposem $m \in [2,n-1]$ corresponent a un cert bloc per xifrar. L'**algorisme de xifratge** es redueix al càlcul d'una exponencial en què la clau és el parell de nombres (e,n):

$$c = E_{(e,n)}(m) = m^e \pmod n$$

L'**algorisme de desxifratge**, per a poder obtenir m a partir de c, consisteix també en una exponenciació, en què la clau és ara un altre parell de nombres (d,n):

$$m = D_{(d,n)}(c) = c^d \pmod n$$

La manera d'obtenir un bon esquema de xifratge i desxifratge està en l'habilitat d'obtenir $\varphi(n)$. Rivest, Shamir i Adleman suggereixen aquest tractament:

1) Trobar el valor $n = p \cdot q$, en què p i q són dos nombres primers grans (als inicis ja se suggerien d'un centenar de dígits cadascun).

2) Coneixent p i q, calcular: $\varphi(n) = (p-1)\cdot(q-1)$.

3) Agafar e relativament primer amb $\varphi(n)$.

4) Calcular $d = e^{-1} \pmod{\varphi(n)}$.

Vegem-ne un exemple, encara que els valors emprats no són els que es podrien fer servir en la realitat.

Exemple 2.1.

Suposem $p = 13$ i $q = 17$. Aleshores $n = 13 \cdot 17 = 221$ i $\varphi(n) = 12 \cdot 16 = 192$. Escollint $e = 11$ $((e,\varphi(n))=(11, 192)=1)$, calculem el valor de d, tal que $d \cdot e = 1 \pmod{\varphi(n)}$ i trobem $d = 35$.* La clau pública serà (11,221) i la clau privada serà (35,221).

*** Per exemple, usant l'algorisme d'Euclides estès, segons l'algorisme 1.4 del mòdul "Cossos finits".**

Aleshores el xifratge serà: $c = E_{(11,221)}(m) = m^{11} \pmod{221}$ i el desxifratge: $m = D_{(35,221)}(c) = c^{35} \pmod{221}$.

Si suposem que el conjunt de missatges originals és $M = \{A,B,...,Y,Z\}$ i la corresponent assignació numèrica és $M_n = \{2,3,...,26,27\}$, i volem xifrar el missatge $m = EDI$ o, numèricament, 060510, farem successivament:

- $6^{11} \pmod{221} = 141$
- $5^{11} \pmod{221} = 164$
- $10^{11} \pmod{221} = 173$

i ens donarà el criptograma $c = 141164173$.

Per al desxifratge anirem agafant successivament blocs de tres dígits i farem:

- $141^{35} \pmod{221} = 6$
- $164^{35} \pmod{221} = 5$
- $173^{35} \pmod{221} = 10$

que ens donarà el missatge original $m = 060510$ o, en caràcters: $m = EDI$.

2.2.2. Signatura digital, basada en l'RSA

Els algorismes de xifratge i desxifratge de l'algoritme RSA són commutatius; és a dir, $D_k(E_k(m) = E_k(D_k(m))$, i per tant, l'esquema RSA pot ser utilitzat per a tots dos objectius de privacitat i autenticitat. Segons aquesta commutativitat es pot utilitzar l'RSA per a construir signatures digitals.

En aquest cas, si suposem dos usuaris A i B, amb claus públiques (e_A,n_A) i (e_B,n_B) i claus privades (d_A,n_A) i (d_B,n_B), respectivament, podrem arbitrar un sistema de signatura digital com s'ha esmentat anteriorment.

Signatura digital, basada en l'RSA

En general, ja que els valors de m seran molt grans, se signarà un resum de m i la signatura serà $s = D_{(d_A,n_A)}(h(m))$, en què $h(m)$ és la funció resum (*hash*) de m.
En aquest cas, en lloc de procedir com indica el segon pas del xifratge, el missatge m s'enviarà a part, xifrat o no. La funció resum haurà de ser coneguda, també per l'usuari B, per a poder procedir a fer la verificació del segon pas del desxifratge.

Si l'usuari A vol enviar el missatge m, signat digitalment, a l'usuari B, es procedirà:

Per part de l'usuari A:

1) Signar m amb la seva clau privada: $s = D_{(d_A,n_A)}(m)$

2) Xifrar la signatura amb la clau pública de B: $c = E_{(e_B,n_B)}(c)$

Per part de B, un cop rebut el criptograma c:

1) Desxifrar el criptograma c amb la seva clau privada: $D_{(d_B,n_B)}(c) = s$

2) Verificar, a partir de la signatura s, si el criptograma c ha estat enviat realment per A: $E_{(e_A,n_A)}(s) = m$

Exemple 2.2.

Usuari A:

Siguin $p = 29$ i $q = 7$ els valors escollits per A. Aleshores $n_A = 29 \cdot 7 = 203$ i $\varphi(n_A) = 28 \cdot 6 = 168$.

Suposem que A tria: ($e_A = 19$,$n_A = 203$) com a clau pública; aleshores ($d_A = 115$,$n_A = 203$) serà la seva clau privada.

Usuari B:

Siguin $p = 13$ i $q = 17$ els valors escollit per B. Aleshores $n_B = 13 \cdot 17 = 221$ i $\varphi(n_B) = 12 \cdot 16 = 192$.

Suposem que B tria: ($e_B = 11$,$n_B = 221$) com a clau pública; aleshores ($d_B = 35$,$n_B = 221$) serà la seva clau privada.

Si suposem que el conjunt de missatges originals és $M = \{A,B,...,Y,Z\}$ i la corresponent assignació numèrica és $M_n = \{2,3,...,26,27\}$, i volem signar digitalment el text original EDI o, numèricament, 060510, farem successivament:

- $6^{115} \pmod{203} = 13$
- $5^{115} \pmod{203} = 93$
- $10^{115} \pmod{203} = 101$

i ens donarà la signatura $s = 013096101$.

Per al xifratge anirem agafant successivament blocs de tres dígits de la signatura s i farem:

- $013^{11} \pmod{221} = 208$
- $096^{11} \pmod{221} = 216$
- $101^{11} \pmod{221} = 186$

que ens donarà el criptograma $c = 208216186$.

Quan l'usuari B ha rebut c, el divideix en blocs de tres xifres i el desxifra amb la seva clau privada:

- $208^{35} \pmod{221} = 013$
- $216^{35} \pmod{221} = 096$
- $186^{35} \pmod{221} = 101$

el missatge s que recupera és inintel·ligible (13 equival al caràcter L, 96 i 101 no tenen equivalència dins del conjunt de missatges originals). Ara, l'usuari B ha de comprovar que 013096101 és la signatura digital de l'usuari A, i per això ha de xifrar el resultat obtingut de l'operació anterior amb la clau pública d'A.

- $013^{19} \pmod{203} = 6$
- $096^{19} \pmod{203} = 5$
- $101^{19} \pmod{203} = 10$

que ens donarà el text original 060510 o, en caràcters, EDI.

Nota

L'usuari B accepta EDI com el missatge que li ha enviat A, senzillament perquè per a ell és intel·ligible; però això en molts casos no seria suficient. Tal com veurem, en altres casos, l'estratègia de signatura serà diferent.

2.3. Criptosistema ElGamal

El criptosistema de clau pública ElGamal es basa en la funció unidireccional exponencial discreta. Aquest criptosistema ha servit de base per a la definició d'un algorisme de signatura alternatiu a l'RSA, el DSA.

Vegeu també

L'algorisme DSA s'estudia al subapartat 2.4. d'aquest mòdul.

2.3.1. Descripció del criptosistema ElGamal

Si fixem un cos finit $\mathbb{F}_p$ i un element primitiu α, suposarem que el missatge que volem xifrar es correspon a un element $m \in \mathbb{F}_p$. A partir d'aquí, cada usuari U tria a l'atzar un enter $r_U \in [2,p-1]$, que serà la seva clau privada. La seva clau pública serà $y_U = \alpha^{r_U} \pmod p \in \mathbb{F}_p$.

Llavors, si un usuari A vol transmetre a l'usuari B el criptograma corresponent al missatge original $m \in \mathbb{F}_p$, haurà de fer les operacions següents dins de $\mathbb{F}_p$.

Llavors, l'usuari B podrà recuperar m a partir del parell de nombres rebuts, fent les operacions següents dins de $\mathbb{F}_p$:

Fortalesa

Qualsevol enemic que vulgui calcular K^{r_B}, a partir del coneixement de K i y_B, però sense conèixer r_B, haurà de calcular prèviament el logaritme discret $log_\alpha(y_B)$. En la dificultat d'aquest càlcul es basa la fortalesa del criptosistema ElGamal.

1) Calcular $\beta = K^{r_B} \pmod p$

2) Calcular $c/\beta \pmod p = m$

Nota

El resultat d'aquesta operació ha de ser, efectivament, igual a m, ja que

$$K^{r_B} \pmod p = (\alpha^{r_B})^k \pmod p = (y_B)^k \pmod p.$$

Tot seguit veurem un exemple d'utilització de l'algorisme, per a poder-lo entendre millor, encara que els valors de l'exemple són petits comparats amb els que hauríem de fer servir en la realitat.

Exemple 2.3.

Suposem el cos finit $\mathbb{F}_{23}$ i sigui $\alpha = 5$ l'element primitiu triat*. Si $r_A = 13$, la clau pública de l'usuari A és $y_A = 5^{13} \pmod{23} = 21$. Si $r_B = 17$, la clau pública de l'usuari B és $y_B = 5^{17} \pmod{23} = 15$.

Si l'usuari A vol transmetre a l'usuari B el missatge $m = 18$ efectuarà els càlculs següents:

1) Agafarà a l'atzar un enter, per exemple $k = 7$, i calcularà $K = 5^7 \pmod{23} = 17$

2) Xifrarà $m = 18$, sabent que $y_B = 15$, com: $c = E_{15}(18) = 18 \cdot 15^7 \pmod{23} = 14$.

3) Transmetrà el parell de nombres (17,14).

Llavors l'usuari B podrà recuperar m a partir del parell de nombres rebuts, (K,c), fent les operacions següents dins de $\mathbb{F}_{23}$:

1) Calcularà $\beta = 17^{17} \pmod{23} = 11$

2) Calcularà $c/\beta = 14/11 \pmod{23} = 18$; $(11^{-1} \pmod{23} = 21)$.

Aquest resultat coincideix amb el valor del missatge original, $m = 18$.

*** Tal com s'ha vist en el mòdul "Cossos finits".**

2.3.2. Signatura digital, basada en el ElGamal

El criptosistema ElGamal no compleix la condició de commutativitat que hem vist en el subapartat de signatura digital basada en l'RSA; en aquest cas no es compleix $E_k(D_k(m)) = m$. No obstant això, es pot adaptar el sistema per a l'autenticació mitjançant un altre tipus de signatura digital.

Suposem que el missatge que signa un usuari A, per a ser tramsmés a un usuari B, és $m \in \mathbb{F}_p$. Per a l'execució de la signatura, l'usuari A ha de fer les operacions següents dins de $\mathbb{F}_p$:

1) Triar un enter k tal que $mcd(k,p-1) = 1$ i calcular $K = \alpha^k \pmod p$.

2) Trobar un enter s tal que $m = r_A \cdot K + k \cdot s \pmod{p-1}$, en què r_A i k són valors que només coneix l'usuari A; per tant, només ell serà capaç de calcular la signatura s.

3) La signatura digital és el parell de nombres (K,s). Aleshores transmetrà (K,s,m), encara que, opcionalment, pot voler també xifrar el missatge m, $c = E_{y_B}(m)$, i transmetre (K,s,c).

Signatura ElGamal

Es pot calcular, de manera directa, fent $s = (m - r_A \cdot K) \cdot k^{-1} \pmod{p-1}$.

Per a la validació de la signatura del missatge m, l'usuari B ha de comprovar que es compleix la igualtat següent:

$$\alpha^m = (y_A)^K \cdot K^s \pmod p$$

en què y_A és la clau pública de l'usuari A i, per tant, disponible per a l'usuari B.

En cas que la igualtat es compleixi, l'usuari B donarà com a autèntic el missatge m que li ha enviat l'usuari A. En cas contrari, el missatge m no serà validat.

Verificació

Efectivament: $(y_A)^K = (\alpha^K)^{r_A}$
$K^s = (\alpha^k)^s = \alpha^{(m-r_A \cdot K)} =$
$\alpha^m \cdot (\alpha^K)^{-r_A}$
El producte de totes dues igualtats resulta ser α^m.

Exemple 2.4.

Suposem que continuem amb les mateixes hipòtesis de l'exemple anterior: $\mathbb{F}_{23}$, $\alpha = 5$, $r_A = 13$ i $r_B = 17$

La clau pública de l'usuari A és $y_A = 21$ i la clau pública de l'usuari B és $y_B = 15$.

Si A vol transmetre el missatge $m = 18$ a l'usuari B de manera secreta i autenticada, farà els càlculs següents:

1) Escollirà a l'atzar un enter, per exemple $k = 7$ ($mcd(7,22) = 1$), i calcularà $K = 5^7 \pmod{23} = 17$

2) Calcularà s tal que $m = r_A \cdot K + k \cdot s \pmod{22}$. És a dir:

$s = (m - r_A \cdot K) \cdot k^{-1} \pmod{22} = (18 - 13 \cdot 17) \cdot 7^{-1} \pmod{22} = 15$; $(7^{-1} \pmod{22} = 19)$

3) Per a transmetre el missatge xifrat (en l'exemple anterior $c = 14$) i signat, l'usuari A envia: $(K,s,c) = (17,15,14)$.

Llavors, l'usuari B pot validar la transmissió a partir de m (en l'exemple anterior, a partir de c, havia retrobat $m = 18$) i el parell de nombres (K,s).

Efectuant el càlcul $(y_A)^K \cdot K^s \pmod p = 21^{17} \cdot 17^{15} \pmod{23} = 6$, que coincideix amb $\alpha^m \pmod p = 5^{18} \pmod{23} = 6$.

Per tant, l'usuari *B* donaria per vàlida la signatura *s* del missatge *m*.

2.4. Algorisme DSA com a alternativa a la signatura digital RSA

El 1991, el NIST (National Institute of Standards and Technology) va fer la proposta de l'algorisme DSA (*digital signature algorithm*) com a estàndard de signatura digital DSS (*digital signature standard*).

Aquest algorisme DSA fou desenvolupat per l'NSA (National Security Agency), a partir de la signatura digital d'ElGamal, però amb el propòsit de reduir-ne la longitud. Aquest algorisme conté els paràmetres següents:

- p, un nombre primer de $2 \cdot L$ bits, en què L és un múltiple de 64 i està comprés entre 512 i 1.024 bits.
- q, un factor primer de $p-1$ d'uns 160 bits. Sigui $n = (p-1)/q$
- α, tal que $\alpha = g^n \pmod p$, en què g és un nombre menor que $p-1$ i de manera que $\alpha \pmod p > 1$.
- x, un nombre qualsevol menor que q.
- y, tal que $y = \alpha^x \pmod p$.
- $h(.)$, una funció unidireccional resum.

Funció unidireccional resum

Una funció resum, h, és un seguit d'operacions que transformen un missatge m, de longitud variable, en una seqüència de bits, $h(m)$, de longitud fixa, de pocs bits, tal com veurem en el proper subapartat.

Els nombres p, q i α són públics per a tots els usuaris de la xarxa, mentre que x és la clau privada i y és la clau pública.

Sigui m el missatge que A vol transmetre a B, el qual vol signar perquè B en pugui fer l'autenticació.

L'usuari A, del qual se suposen coneguts els paràmetres anteriors, excepte x, haurà de fer les operacions següents:

1) Triar un nombre aleatori k, més petit que q.

2) Generar dos valors r i s, tals que:
$r = (\alpha^k \pmod p) \pmod q$
$s = ((h(m) + x \cdot r) \cdot k^{-1}) \pmod q$

3) Enviar el missatge m i la seva signatura digital (r,s)

L'usuari B, en rebre el missatge m, i la seva signatura digital corresponent (r,s), podrà fer el procés d'autenticació següent:

1) Seleccionar del directori públic els paràmetres de A: p,q,α i $h(.)$

2) Calcular: $w = s^{-1} \pmod q$

$u_1 = (h(m) \cdot w) \pmod q$
$u_2 = (r \cdot w) \pmod q$

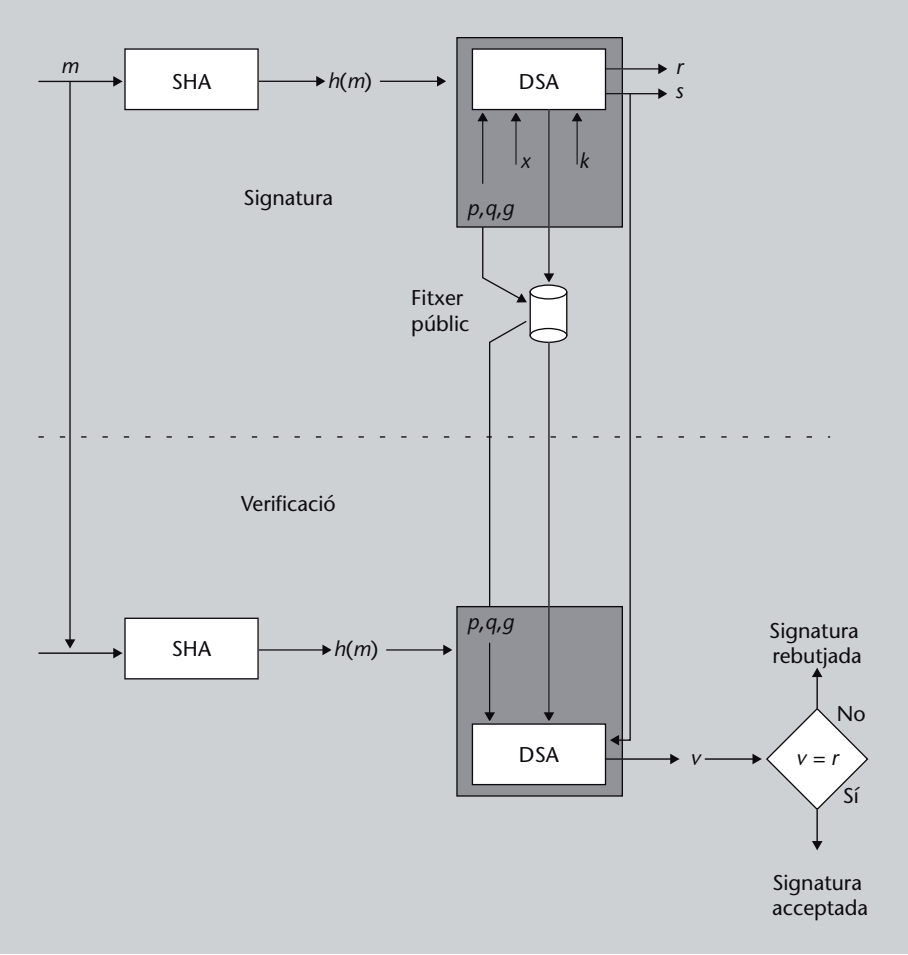
3) Calcular: $v = ((\alpha^{u_1} \cdot y^{u_2}) \pmod p) \pmod q$

4) Autenticar: Si $v = r$ aleshores la signatura digital de A, (r,s), és acceptada per B.

Verificació de signatura

Per simplicitat, i sense perdre rigor en la prova, obviarem els càlculs modulars.
$v = \alpha^{u_1} \cdot y^{u_2} = (\alpha^{h(m)\cdot s^{-1}}) \cdot (\alpha^x)^{r\cdot s^{-1}} = \alpha^{(h(m)+x\cdot r)\cdot s^{-1}}$,
i segons la definició de s, aquest resultat és, efectivament, $\alpha^k = r$.

El diagrama següent representa el procés d'aquesta signatura digital.



Exemple 2.5.

Suposem que el conjunt de missatges originals és $M = \{A,B,...,Y,Z\}$ i la corresponent assignació numèrica $M_n = \{2,3,...,26,27\}$. Si l'usuari A vol signar digitalment el text original EDI o, numèricament, 060510, tenint en compte els paràmetres del criptosistema, farà les operacions següents:

Paràmetres públics:
$p = 29, q = 7$ $(n = (p-1)/q = 28/7 = 4)$
$\alpha = 5^4 \pmod{29} = 16$; $(g = 5)$
Clau privada: $x = 3$
Clau pública $y = 16^3 \pmod{29} = 7$
Paràmetre aleatori secret: $k = 6$

Llavors l'usuari A calcula (suposant la funció fictícia resum anterior):

$h(m) = h(06,05,10) = 60 \pmod{29} = 2$
$r = (16^6 \pmod{29}) \pmod 7 = 20 \pmod 7 = 6$
$s = (2 + 3 \cdot 6) \cdot 1/6 \pmod 7 = 6 \cdot 1/6 \pmod 7 = 1$

Funció resum fictícia

Per simplicitat de l'exemple, hem agafat com a funció resum, fictícia, el producte dels valors numèrics dels caràcters imparells (mod 29). Aquesta proposta no té res a veure amb una funció resum real, com les que detallarem més endavant.

Per tant, $DSA(EDI) = DSA(060510) = (6,1)$ i transmetrà cap a l'usuari B: $(EDI,6,1)$.

L'usuari B, a la recepció d'aquest triple, fa la verificació, després de seleccionar els paràmetres p,q,g i $h(.)$:

Calcula:

$h(EDI) = 2$, ja que $(h(06,05,10) = 6 \cdot 10 \pmod{29})$
$w = 1^{-1} \pmod 7 = 1$
$u_1 = 2 \cdot 1 \pmod 7 = 2$
$u_2 = 6 \cdot 1 \pmod 7 = 6$

Verifica que $v = (16^2 \cdot 7^6 \pmod{29}) \pmod 7 = 20 \pmod 7 = 6 (= r)$ i per tant, valida la signatura.

Criticitat de la funció resum

Si suposem que enviem un text diferent amb la mateixa signatura (EDJ, 6, 1), l'usuari B faria els càlculs següents:
$h(EDJ) = 8$;
$(h(06,05,11) = 6 \cdot 11 \pmod{29} = 8)$
$w = 1^{-1} \pmod 7 = 1$
$u_1 = 8 \cdot 1 \pmod 7 = 1$
$u_2 = 6 \cdot 1 \pmod 7 = 6$
$v = (16^1 \cdot 7^6 \pmod{29}) \pmod 7 = 23 \pmod 7 = 2 (\neq r = 6)$, amb la qual cosa no s'autenticaria la firma.
No obstant això, hem d'observar que si el missatge signat hagués estat BUS, com que $h(BUS) = h(03,22,20) = 3 \cdot 20 \pmod{29} = 2$, derivaria la mateixa signatura. Amb la qual cosa es pot veure que l'elecció de la funció resum és crítica respecte a aquest algorisme.

2.5. Funcions resum: MD5 i SHA-1

El propòsit de les funcions resum és el de proporcionar una empremta, de pocs bits, d'un fitxer, missatge o qualsevol altre bloc de dades, per gran que sigui, el qual es vol autenticar.

Una funció resum, h, ha de tenir les propietats següents:

1) S'ha de poder aplicar la funció h a un bloc de dades de qualsevol llargària.

2) El resultat de la funció h ha de ser de longitud fixa, de pocs bits.

3) Ha de ser computacionalment eficient calcular $h(m)$ per a qualsevol m, tant en implementacions de maquinari com de programari.

4) Per a un bloc determinat x, no ha de ser computacionalment eficient trobar un missatge m tal que $h(m) = x$.

5) Per a un missatge determinat m, no ha de ser computacionalment eficient trobar un altre missatge $m' \neq m$ tal que $h(m) = h(m')$.

6) No ha de ser computacionalment eficient trobar un parell de missatges (m,m') tal que $h(m) = h(m')$.

Enllaç d'interès

Es pot trobar un programa de simulació de les funcions resum MD5 i SHA-1, d'ús lliure, a l'adreça: www.criptored.upm.es

Utilitat de les propietats

Les primeres tres propietats són tres requisits per a poder dur a la pràctica les funcions resum i l'autenticació de missatges.

La quarta propietat serveix per a assegurar que no ha de ser computacionalment eficient trobar la inversa de les funcions resum (funcions unidireccionals, *one-way functions*).

La cinquena i sisena propietats garanteixen que no ha de ser possible trobar un missatge alternatiu amb la mateixa seqüència resum que l'original.

Seguidament passarem a examinar els dos algorismes que implementen les funcions resum més utilitzades actualment: l'algorisme *message-digest*, MD5, i l'algorisme *secure hash*, SHA.

2.5.1. L'algorisme MD5

El *message-digest* MD5 el va desenvolupar Ron Rivest i consisteix en un algorisme que té com a entrada un missatge de llargada arbitrària i produeix una sortida de 128 bits (el *message digest*).

Eficiència del MD5

Rivest fa avinent que, amb l'MD5, la dificultat de trobar dos missatges amb la mateixa seqüència resum és de l'ordre de 2^{64} operacions i, per altra banda, que la dificultat de trobar un missatge amb una seqüència resum determinada és de l'ordre de 2^{128} operacions.

El missatge d'entrada es processa en blocs de 512 bits, el procés del qual consta dels passos següents:

- **Pas 1: afegir bits de farciment**. S'insereixen bits en el missatge perquè la seva longitud final sigui congruent amb 448 (mod 512). Aquest enfilall de bits de *padding* consisteix en un 1 seguit de la quantitat necessària de zeros.

Efecte del farciment

Els bits de farciment s'afegeixen sempre, fins i tot si el missatge ja té la longitud que volem. Per exemple, si el missatge té 448 bits de longitud, llavors s'afegeixen 512 bits de farciment per a arribar a una longitud de 960 bits. Això vol dir que afegim una quantitat de bits de farciment que oscil·lla entre 1 i 512.

- **Pas 2: afegir la longitud**. Al resultat del pas anterior li afegim la representació en 64 bits de la longitud del missatge abans d'afegir-hi els bits de farciment. Per tant, aquest camp conté la longitud del missatge original (mod 2^{64}). Després dels dos primers passos tenim un missatge amb una longitud en bits igual a un múltiple de 512. El missatge es divideix en blocs de 512 bits, anomenats $Y_0, Y_1 \cdots Y_{L-1}$; per tant, podem expressar la longitud del missatge com $L \cdot 512$ bits, o també veure el missatge com un múltiple de 16 paraules de 32 bits.

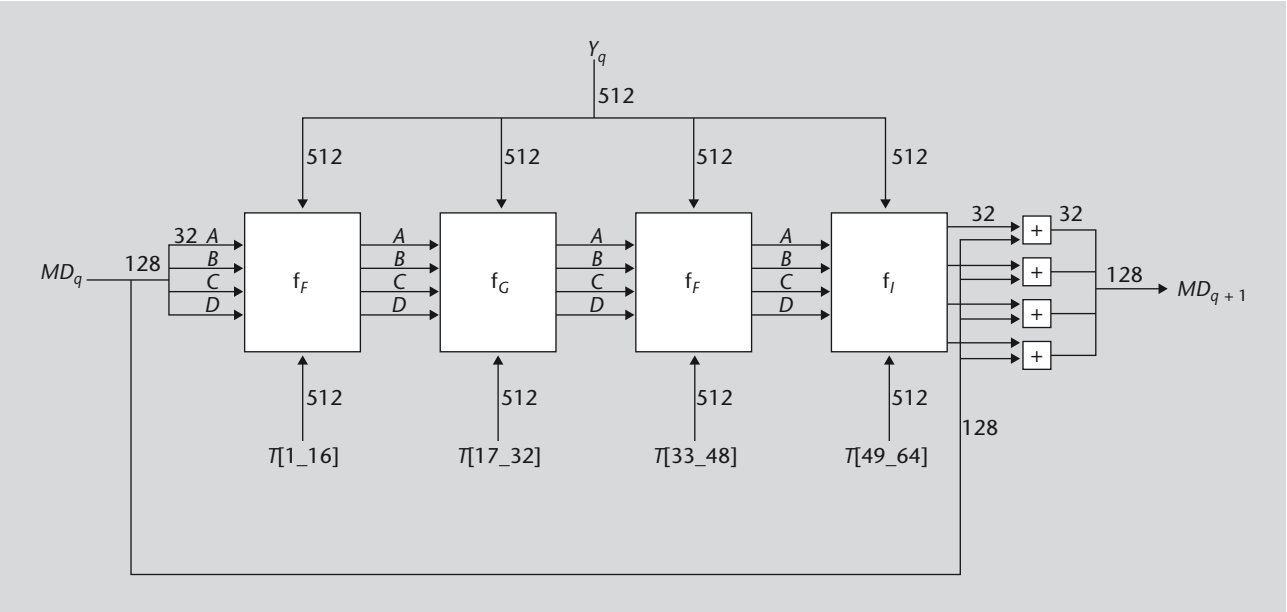
- **Pas 3: iniciar la memòria intermèdia MD**. Per a desar tant els resultats intermedis com el resultat final, s'utilitza una memòria intermèdia de 128 bits, la qual es representa per quatre paraules de 32 bits, A,B,C,D, que s'inicien amb els valors hexadecimals següents:

$$A = 01234567$$
$$B = 89ABCDEF$$
$$C = FEDCBA98$$
$$D = 76543210$$

- **Pas 4: Processament del missatge en blocs de 512 bits (16 paraules de 32 bits)**. La part central de l'algorisme és un mòdul que té quatre etapes de funcionament.

  Les quatre etapes tenen una estructura similar, però utilitzen funcions lògiques primitives diferents.

  A la figura següent hem etiquetat cada etapa amb f_F, f_G, f_H, f_I per a indicar que totes tenen la mateixa estructura, f, però amb una funció primitiva diferent: F,G,H,I.



Cada etapa té com a entrada el bloc respectiu de 512 bits (Y_q) i el valor de la memòria intermèdia A,B,C,D de 128 bits i actualitza el contingut de la memòria intermèdia. En cada etapa també s'utilitza una quarta part dels 64 elements d'una taula $T[1...64]$ que proporciona un conjunt pseudoaleatori de seqüències de 32 bits, que serveixen per a eliminar qualsevol regularitat de les dades d'entrada. Per tant, el procés del bloc Y_q consisteix a agafar com a entrada el mateix Y_q i el resultat (*message digest*) intermedi corresponent MD_q per a produir el MD_{q+1}. Les sumes que es fan al final de les quatre etapes són sumes (mod 2^{32}).

- **Pas 5: sortida**. Després de processar els L blocs de 512 bits, la sortida MD_{L-1} del bloc de procés L-èsim és la seqüència resum de 128 bits.

2.5.2. L'algorisme SHA-1

L'algorisme SHA-1 fou desenvolupat pel NIST i publicat com a estàndard de processament d'informació federal FIPS (*federal information process standard*, PUB 180) l'any 1993.

L'algorisme té com a entrada un missatge de longitud més petita que 2^{64} bits i produeix una sortida de 160 bits (el *message digest*). El missatge d'entrada es processa en blocs de 512 bits, el procés del qual consta dels passos següents:

- **Pas 1: afegir bits de farciment. Pas 2: afegir la longitud**. Igual que en l'algorisme MD5.

- **Pas 3: iniciar la memòria intermèdia SHA**. Per a desar tant els resultats intermedis com el resultat final, s'utilitza una memòria intermèdia de 160 bits. Podem representar la memòria intermèdia com a cinc paraules de 32 bits, A,B,C,D,E, que s'inicien amb els valors hexadecimals següents:

Comparació entre SHA i MD5

Els algorismes MD5 i SHA són bastant similars perquè tots dos deriven del mateix algorisme MD3.
La diferència més òbvia, i la més important, és que la seqüència resultant d'aplicar l'SHA és 32 bits més llarga que la de l'MD5. Llavors, l'SHA és un algorisme més fort, des del punt de vista de la criptoanàlisi, que el MD5. En canvi, l'SHA consta de 80 passos i l'MD5 només de 64, cosa que fa que l'SHA s'executi vora un 25% més lent. Hem de notar que els dos algorismes tenen una gran quantitat de sumes mòdul 2^{32}, i per tant, tots dos funcionen molt bé en arquitectures de 32 bits.

$$A = 67452301$$
$$B = EFCDAB89$$
$$C = 98BADCFE$$
$$D = 10325476$$
$$E = C3D2E1F0$$

- **Pas 4: processament del missatge en blocs de 512 bits (16 paraules de 32 bits)**. La part central de l'algorisme és un mòdul que té 80 etapes de processament. La lògica d'aquest mòdul és la següent:

  Cada etapa té com a entrada l'actual bloc de 512 bits, Y_q, i els 160 bits de la memòria intermèdia *ABCDE* i, com a resultat, actualitza el contingut de la memòria intermèdia.

  A cada etapa es fa ús de la suma per una constant K_t. De fet, només s'utilitzen quatre constant diferents. Els valors hexadecimals són els següents:

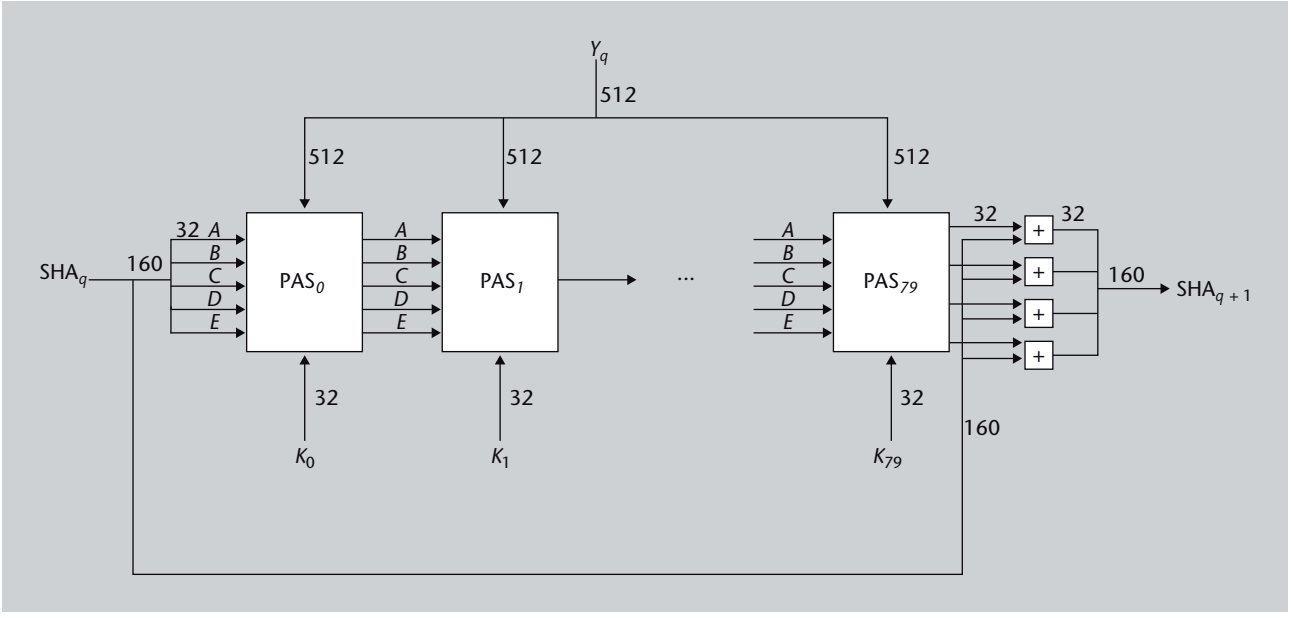
  $0 \leq t \leq 19$, aleshores $K_t = 5A827999$
  $20 \leq t \leq 39$, aleshores $K_t = 6ED9EBA1$
  $40 \leq t \leq 59$, aleshores $K_t = 8F1BBCDC$
  $60 \leq t \leq 79$, aleshores $K_t = CA62C1D6$

  És a dir, l'algorisme SHA, per a processar un bloc Y_q de 512 bits, agafa com a entrada el bloc Y_q i el valor intermedi en aquell moment de la seqüència resum SHA_q que s'agafa de la memòria intermèdia *ABCDE*. Després, el resultat de les 80 etapes se suma a SHA_q, i el resultat és SHA_{q+1}, que es colloca a la memòria intermèdia *ABCDE*. Aquesta darrera suma es fa amb dos sumands de 160 bits cada un, i l'operació es fa de manera independent per a cada una de les cinc paraules de 32 bits que té cada sumand (suma (mod 2^{32})).

SHA-2 i SHA-3

Actualment, s'utilitza la variant SHA-2, desenvolupada el 2005 a partir de l'SHA-1, en què la sortida pot ser de 224, 256, 384 o 512 bits, per tal d'augmentar la dificultat que pugui ser trencat. A més a més, està en marxa un concurs públic per a dissenyar el nou estàndard SHA-3, que segurament es farà públic el 2012.

- **Pas 5: sortida**. Després de processar els *L* blocs de 512 bits, la sortida SHA_{L-1} del bloc de procés *L*-èsim és la seqüència resum de 160 bits.



2.6. Infraestructura de clau pública: PKI

En l'àmbit de les comunicacions obertes és imprescindible garantir la identitat dels usuaris, a més dels serveis de seguretat: confidencialitat, integritat, autenticació i no-repudi.

Perquè aquestes operacions es puguin fer de manera fiable s'han de complir dues condicions:

- Que la clau privada es custodiï de manera segura i no es desvetlli a ningú. Per a aconseguir això, la clau privada s'emmagatzema en un suport físic impossible de duplicar, com una targeta intel·ligent. A més per a accedir al contingut de la targeta es necessita un número personal, que solament el seu propietari legítim coneix.

- Que es pugui determinar a quina persona pertany una clau pública. D'aquesta manera es pot saber, per exemple, qui ha portat a terme la signatura electrònica d'un document.

Lectura recomanada

Per a fer més comprensible aquest subapartat és recomanable el mòdul 7 del llibre de criptografia de J. Domingo, J. Herrera i H. Rifà-Pous dels estudis d'Informàtica i Multimèdia de la UOC.

Per a donar compliment a aquestes dues condicions, s'utilitza el *certificat electrònic*, emès per una autoritat de certificació (CA). El suport tecnològic del certificat electrònic és la criptografia de clau pública. Així es pot veure un certificat electrònic com un document electrònic que associa una clau pública amb el seu propietari.

Per això el certificat digital contindrà la clau pública juntament amb dades de caràcter personal del posseïdor de la clau (nom, DNI...). Normalment conté més informació (dates de validesa i altres), i també es refereix a l'àmbit d'utilització del certificat, el que es coneix com a política de certificació. Per exemple, si és un certificat d'ús personal o ens acredita per a actuar en nom d'una empresa.

En fer una signatura electrònica se sol adjuntar el certificat electrònic del signant, de manera que se'n pot extreure la clau pública per a verificar la signatura i alhora comprovar la identitat del signant.

Una infraestructura de clau pública (PKI, *public key infrastructure*) és una estructura de sistemes informàtics, procediments d'operació, protocols, polítiques de certificació, repositoris d'informació, estàndards, declaracions de pràctiques i recursos humans, la finalitat dels quals és oferir als usuaris una plataforma per a la gestió de la identitat digital.

Una PKI disposa dels elements i de l'arquitectura necessaris per a integrar tots els procediments de sol·licitud de certificats, verificació d'identitats, generació de claus, emmagatzematge i publicació de certificats electrònics, renovació, revocació, etc.

Les infraestructures de clau pública es fonamenten en la interacció de diferents subsistemes, dels quals destaquen els següents:

- **Autoritat de certificació, CA**. Una autoritat de certificació (**CA,** *certificate authority*), és una entitat de confiança, la finalitat de la qual és emetre, renovar i revocar certificats electrònics. Les autoritats de certificació constitueixen el nucli de les infraestructures de clau pública, que permeten utilitzar els **certificats electrònics** amb total seguretat.

  Exemples de CA

  Actualment, un usuari pot escollir entre múltiples CA per a aconseguir un certificat electrònic, però les més utilitzades són internacionalment, Verisign, a Espanya l'FNMT i en l'àmbit català el CATCert.

  Verisign* és una de les empreses de més reputació i prestigi internacional en el món de la certificació digital i la seguretat de la informació. Encara que el seu ventall de serveis és molt ampli (solucions comercials per a comerç electrònic, servidors segurs, targetes intel·ligents, servidors de noms de domini, consultoria...), el més conegut és el de CA per a l'expedició de certificats electrònics, àmpliament utilitzats a Internet.

  * http://www.verisign.com

  Fàbrica Nacional de Moneda i Timbre (FNMT)**, és un organisme públic nacional espanyol que depèn del Ministeri d'Economia que té establerta una arquitectura de certificació, CERES, per a autenticar i garantir la confidencialitat de les comunicacions entre ciutadans, empreses o altres institucions i administracions públiques per mitjà de xarxes obertes de comunicació.

  ** http://www.fnmt.es

  CATCert*** és l'Agència Catalana de Certificació, que emet i gestiona l'idCAT, que és un certificat electrònic que garanteix la identitat de les persones a Internet i permet operar amb diferents administracions.

  *** http://www.catcert.cat

- **Autoritats de registre, RA**. Una autoritat de registre (**RA,** *registration authority*) és una entitat encarregada de portar a terme els processos de verificació d'identitat, sol·licitud i distribució de certificats electrònics. Normalment, en una PKI, els usuaris finals no interactuen directament sobre la CA, sinó que canalitzen les seves operacions per mitjà d'una o diverses RA.

  No obstant això, aquests subsistemes no poden expedir certificats electrònics per si mateixos.

- **Certificats electrònics**. Un certificat electrònic és un arxiu o document electrònic expedit i signat per una CA en el qual es vincula una identitat a una clau pública, lligada, al seu torn, a la clau privada corresponent.

  Per a obtenir un certificat electrònic, l'usuari es dirigeix a una RA (autoritat de registre); aquesta verifica la identitat de l'usuari i demana a la CA que expedeixi el certificat.

- **Directori** *lightweight directory access protocol*, **LDAP**. La seva finalitat, dintre d'una PKI, és la de mantenir un registre d'usuaris i actuar com a magatzem per als certificats electrònics i la llista de certificats revocats (CRL), que veurem més endavant. El protocol LDAP és una versió simplificada del protocol X.500 que especifica tant el model d'informació com els mecanismes d'accés a aquesta.

2.6.1. Sistemes gestors de certificats electrònics: la recomanació X.509

L'auge de la certificació electrònica neix de la consolidació internacional del protocol estàndard X.509. A partir d'aquest moment, no solament apareixen en el mercat multitud d'aplicacions que aprofiten els serveis de la identitat digital, sinó també un gran nombre de paquets de programari que implementen les funcions bàsiques d'una PKI. Aquests paquets reben el nom de *sistemes gestors de certificats electrònics*.

Les solucions d'implementació per als sistemes gestors de certificats electrònics es poden classificar en tres categories: les integrades en el sistema operatiu, les lliures (de codi obert) i les comercials.

La recomanació X.509 de l'ITU-T forma part de la sèrie de recomanacions X.500, la finalitat de les quals és definir un servei de directori. Per directori s'entén un servidor o conjunt distribuït de servidors que gestionen una base de dades d'informació sobre usuaris. Actualment, parlar d'X.509 i certificats electrònics és parlar del mateix concepte. Actualment aquest estàndard de certificació electrònica s'utilitza en protocols d'Internet d'ús tan estès com SSL, SMIME (*secure multipurpose Internet mail extensions*) i IPSec (*IP security*).

2.6.2. Llistes de certificats revocats: CRL

Una llista de certificats revocats (CRL, *certificate revocation list*) és un document electrònic expedit i signat per una CA en el qual s'inclouen els números de sèrie de tots aquells certificats que, sense haver expirat, han estat revocats per algun motiu.

En rebre un certificat electrònic, l'usuari ha de consultar la CRL de la CA que signa el certificat per a verificar-ne la validesa. El protocol OCSP (*online certificate status protocol*) permet fer consultes en temps real sobre la base de dades de certificats revocats d'una CA. Alguns navegadors web ja inclouen suport per a OCSP.

La validació de certificats en temps real és imprescindible per al desenvolupament del comerç electrònic.

La recomanació X.509 defineix un format estàndard per a les llistes de certificats revocats, de manera anàloga a l'estructura suggerida per als certificats electrònics.

3. Criptografia quàntica i postquàntica

De manera molt resumida podríem dir que, avui dia, la criptografia de clau pública s'utilitza per a la distribució de claus privades, per a la signatura digital i altres protocols d'autenticació i, en canvi, la criptografia de clau privada s'utilitza per a aconseguir la privacitat de les dades. Els algorismes de clau pública més utilitzats són RSA, DSA i ECDSA (l'algorisme DSA utilitza corbes el·líptiques). Què passaria si en pocs anys algú anunciés la construcció d'un gran ordinador quàntic?

Atès que la criptografia de clau pública es basa en la factorització o en el problema del logaritme discret o del logaritme el·líptic i que no existeix l'*algorisme de Shor* per els ordinadors clàssics, sembla que amb l'adveniment de la computació quàntica la seguretat basada en RSA, DSA i ECDSA serà compromesa. Tot i així, no podem dir que la criptografia està sota sospita de desaparició amb l'adveniment de la computació quàntica. Hi ha tot un camp criptogràfic postquàntic que podrà resistir els grans ordinadors clàssics i els ordinadors quàntics.

Algorisme de Shor

El 1994, Shor va descobrir un algorisme capaç de factoritzar un nombre producte de dos primers amb una complexitat polinòmica sobre un hipotètic ordinador quàntic. La base de l'algorisme és una transformada discreta de Fourier, que en un ordinador quàntic, i de manera probabilística, dóna el resultat. S'han fet proves reals d'aquest algorisme amb prototipus d'ordinador quàntic que han funcionat. El 2001 en l'IBM Almaden Research Center, fent servir un prototipus d'ordinador quàntic que funcionava amb registres de 7 qubits basats en NMR (*nuclear magnetic resonance*) han factoritzat el nombre $15 = 3 \cdot 5$

Alguns d'aquests sistemes de xifratge que es creu que són resistents a la computació clàssica i quàntica són:

- **Criptografia basada en funcions resum**
- **Criptografia basada en la teoria de codis**
- **Criptografia basada en la combinatoria**
- **Criptografia de clau privada com l'AES**. Hi ha un algorisme que pot simplificar els càlculs per a trencar l'AES en un ordinador quàntic. És l'algorisme de Grove, però no és tan espectacular com l'algorisme de Shor. En el cas de Grove es passa d'una complexitat $O(n)$ en el cas clàssic a una complexitat $O(\sqrt{n})$ en el cas quàntic.

Tant el sistema RSA com el de McEliece (basat en la teoria de codis, que veurem més endavant) han estat proposats el mateix any 1978. Tots dos han aconseguit resistir més de 30 anys l'esforç del criptoanàlisi per trencar-los.

L'RSA es basa en la dificultat de la factorització. En l'any 1978 el millor algorisme conegut per a factoritzar tenia una complexitat exponencial de l'ordre

$$O(exp(\log(n)^{1/2} \log\log(n)^{1/2})).$$

Amb el temps, aquesta complexitat ha estat millorada, bàsicament utilitzant nous conceptes matemàtics i, actualment, sembla que ja no es pot aconseguir cap més millora, i es manté una complexitat de l'ordre

$$O(exp(\log(n)^{1/3}\log\log(n)^{1/3})).$$

La complexitat del sistema de McEliece era de l'ordre $O(exp(n/(2\cdot\log(n))))$, el 1978. Hi ha hagut millores i sembla que la millor complexitat possible continuarà essent més o menys la mateixa, potser que en lloc del 2 hi haurà una constant una mica menor.

Complexitat

La complexitat de l'RSA es diu *subexponecial* i la del sistema McEliece es diu *exponencial*.

La pregunta, ara, és òbvia. Per què, actualment, no fem servir el sistema de McEliece en lloc d'RSA? La resposta ràpida és que la mida de la clau fa que optem per RSA en lloc de McEliece. Per al mateix nivell de seguretat, RSA utilitza claus d'alguns milers de bits, mentre que McEliece n'hauria d'utilitzar de vora un milió de bits.

Encara falta temps perquè la computació quàntica sigui una realitat. En el camí podem pensar en ordinadors clàssics cada vegada més potents (no cal que siguin potents treballant aïllats, però podem pensar en grans sistemes de computació distribuïda).

Alguns sistemes de xifratge, com l'RSA, amb quatre mil bits de clau, es creu que seran resistents als atacs amb ordinadors clàssics grans, però no ho seran als grans ordinadors quàntics. Algunes alternatives, com ara McEliece, amb una clau de quatre milions de bits, es creu que serà capaç de resistir els atacs dels grans ordinadors clàssics i quàntics.

Ens hem centrat en la criptografia de clau pública, ja que els ordinadors quàntics sembla que tenen molt poc efecte sobre la criptografia de clau privada i la criptografia basada en funcions resum.

Encara que els ordinadors quàntics no són encara una realitat, sí que hi ha certs fenòmens físics basats en la mecànica quàntica que es poden fer servir en criptografia. Un exemple clar n'és el sistema de distribució de claus basat en l'algorisme de Bennet i Brassard. A banda d'aquest algorisme criptogràfic basat en la mecànica quàntica, si ara ens situem en l'època en què ja funcionin els ordinadors quàntics tenim alguns plantejaments que hem de començar a estudiar:

Vegeu també

L'algorisme de Bennet i Brassard l'estudiarem en el subapartat 3.1.

- **Eficiència**: el programari de criptografia postquàntica és més lent que el programari criptogràfic d'avui.

- **Confiança**: avui dia estem utilitzant sistemes criptogràfics clàssics que han sobreviscut després de molts anys d'esforços criptoanalítics per trencar-los.

En considerar nous sistemes quàntics, que són recents i no del tot provats, és necessari que també els criptoanalistes tinguin temps per buscar els atacs a aquests sistemes i depurar-los.

- **Usabilitat**: és important desenvolupar programari i maquinari adaptant les implementacions als nous conceptes amb molta cura per a evitar pèrdues de temps o efectes laterals. Implementacions de conceptes com els d'aleatorietat o farciment (*padding*) s'han de millorar en les noves tecnologies i segurament hi haurà sistemes híbrids, que comparteixin les noves tecnologies amb les actuals, que necessitaran estandarditzacions.

3.1. Criptografia quàntica

A diferència dels sistemes criptogràfics convencionals, ja siguin de clau pública o privada (que basen la seva seguretat en el fet de mantenir una clau de manera privada), els sistemes criptogràfics quàntics basen la seva fortalesa en un fenomen físic. El 1984 Bennet i Brassard varen dissenyar (a escala teòrica) un protocol criptogràfic basat en un fet de la física quàntica: la impossibilitat de mesurar simultàniament un parell d'observables (principi d'incertesa de Heisenberg, 1927).

La seguretat tradicional d'un mètode de distribució de claus s'ha basat en problemes intractables a causa de la seva complexitat computacional. El mètode de Benet i Brassart es basa en un fet físic inviolable.

Podem imaginar un núvol de fotons que vibren en totes les direccions perpendiculars a la seva línia de propagació. Si els fem passar per un filtre polaritzat en vertical, els fotons que vibren verticalment passaran pel filtre i, els altres, ho faran amb una probabilitat $\cos^2(\varphi)$, en què φ és l'angle que separa la seva direcció de vibració de la vertical. Només els fotons que vibren horitzontalment ($\cos(\varphi) = 0$) no passaran pel filtre.

El protocol que descrivim a sota permet a dos usuaris A i B compartir una clau secreta sobre un canal no segur. Al final del protocol, A i B tindran la seguretat que cap enemic criptoanalista no ha interceptat la seva comunicació i podran usar la clau compartida per a xifrar els missatges següents (normalment amb una sistema com l'AES). Vegem, doncs, aquest protocol:

Protocol d'Ekert

El 1991 Artur Ekert va presentar un altre protocol diferent del de Bennet i Brasard. Aquest es basa en el fet que A i B reben els fotons d'una parella "entrellaçada". En aquest cas, la seguretat es basa en l'efecte Einstein-Podolsky-Rosen.

- A envia a B una seqüència de polsos de fotons. Cada pols està polaritzat aleatòriament en una de les quatre direccions: vertical ($\uparrow$), horitzontal ($\leftarrow$), segons la diagonal primària ($\nwarrow$) i segons la diagonal secundària ($\nearrow$).

- B fa servir, aleatòriament, un detector polaritzat per a detectar polaritzacions vertical-horitzontal o polaritzacions en diagonal, però no totes dues al mateix temps.

Per exemple:

A envia a *B*:	↖	↖	↑	↑	←	↗	←	↗	↑	←
B fa servir:	⊗	⊗	⊕	⊗	⊕	⊕	⊕	⊗	⊕	⊗
B obté:	↖	↖	↑	↗	←	↑	←	↗	↑	↖

Els polsos 4, 6 i 10 també podien haver estat ↖, ← i ↗, respectivament.

- *A* i *B* es comuniquen per mitjà d'un canal no segur per veure quins detectors ha fet servir *B* que no hagi fet servir *A*. Ells dos, *A* i *B* es guarden només els bits que corresponen als detectors correctes.

  En el nostre cas, els dos usuaris estan d'acord que el bit 1 es representarà per ← i ↖, mentre que el bit 0 serà ↑ i ↗. O sigui, que *A* i *B* hauran generat entre ells dos la seqüència:

  1101100

  Aquesta seqüència no és coneguda per l'espia que intenta interferir en la comunicació entre *A* i *B*, ja que la conversa (sobre un canal insegur) entre *A* i *B* només deia quins detectors s'havien usat correctament. I cada detector pot donar, indistintament, tots dos resultats 1's i 0.

Qualsevol espia que intercepti els fotons que envia *A* els haurà de reenviar a *B* i, bàsicament, té dos grans problemes:

- Quan escolta l'enviament entre *A* i *B*, l'espia només pot deduir amb seguretat els bits que corresponen a detectors que ell mateix està usant i que, a la vegada, coincideixin amb els detectors que estan utilitzant *A* i *B*.

- Una de cada quatre vegades, el detector que està usant l'espia no coincideix ni amb el d'*A* ni amb el de *B*, que a la vegada coincideix amb el d'*A*. En aquests casos, *A* i *B* es posen d'acord en el detector que usen, però el bit obtingut per ells dos serà diferent. Si *A* i *B* descobreixen (per un canal que no cal que sigui secret) alguns dels bits obtinguts en el protocol podran deduir la presència de l'espia i avortar el procés.

Hi ha alguns prototipus d'aquest protocol que estan funcionant sobre fibra òptica entre unes distàncies de vora 200 km (Toshiba Research-2003).

Tot i que la física quàntica ens assegura la validesa del protocol anterior, des dels seus primers intents de construcció de prototipus hi ha hagut diversos problemes en la implementació que n'han ajornat la comercialització. No és tan senzilla la construcció "segura" de tots els dispositius implicats en el protocol. Segurament els països asiàtics estan al capdavant de la recerca en aquesta

àrea. El Japó ha anunciat plans per a tenir satèl·lits amb comunicacions de tipus quàntic el 2013 i Xina el 2016.

3.2. Els codis correctors d'errors en la criptografia postquàntica

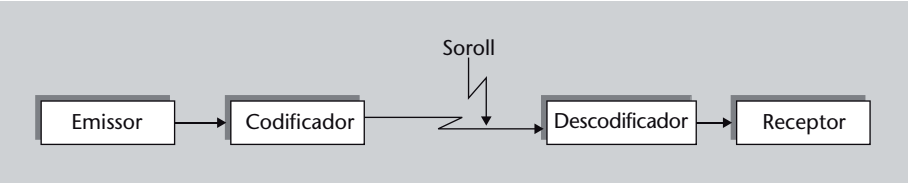
En aquest subapartat presentarem dos sistemes criptogràfics, basats en la teoria de codis correctors d'errors. En particular ens centrarem en els criptosistemes de McEliece i de Niederreiter, definits a partir de codis dels quals es coneix un sistema de descodificació computacionalment eficient. En tots dos casos, la funció unidireccional utilitzada pel xifratge es basa en una transformació de la matriu generadora del codi, en el cas de McEliece, o de la matriu de control, en el cas de Niederreiter, per a treballar amb un codi equivalent però que fa que l'algorisme de descodificació sigui ara computacionalment ineficient.

Per a comprendre millor aquests dos criptosistemes, vegem unes nocions bàsiques de codis correctors d'errors.

3.2.1. Nocions bàsiques de codis correctors d'errors

Els codis detectors i correctors d'errors tenen com a objectiu principal reduir la probabilitat mitjana d'error en la descodificació, mitjançant la incorporació de redundàncies en la transmissió.

Considerem ara el sistema de transmissió d'informació, representat pel diagrama següent.



El principi general de la detecció d'errors és el següent: el codificador afegeix, segons una regla C, una certa quantitat d'informació suplementària a la informació útil, i el descodificador, en primer lloc, verifica si aquesta llei és respectada. Si no ho és, estem segurs que, almenys, s'ha produït un error. Aquesta llei C, que caracteritza el codi, consisteix en una correspondència biunívoca entre la informació per enviar i el missatge enviat.

En el que segueix, considerem el cos finit $\mathbb{F} = \mathbb{F}_q$, en què $q = p^m$ i p un nombre primer.

Definició 3.1 (Codi bloc).

Donada una font d'informació $S = \{A_1, A_2, \cdots, A_k\}$ i un alfabet $\mathbb{F}$, es considera el producte cartesià $\mathbb{F}^n$.

Anomenarem *codi bloc*, de longitud n, qualsevol subconjunt $C \subset \mathbb{F}^n$ de manera que a cada element $A_i \in S$, li fem correspondre, de manera única, un vector $v_i \in \mathbb{F}^n$, que anomenarem **paraula codi**.

Si el cardinal de S és M ens referirem al codi C com:

$$C(M,n) = \{v_i = (v_{i_1}, v_{i_2}, \cdots, v_{i_n}), \text{ tal que } v_{i_j} \in \mathbb{F}\}$$

Podem considerar un codi bloc com un conjunt de seqüències d'elements d'un cos finit $\mathbb{F}$ (paraules codi) de la mateixa llargària. Un codi bloc és binari quan $q = 2$.

Els paràmetres per considerar d'un codi bloc $C(M,n)$, a més de la seva longitud, n, i el seu nombre de paraules codi M, són: la taxa de transmissió: $(1/n)log_q(M)$ i la seva distància mínima entre totes les paraules codi, d.

Exemple 3.1.

Imaginem el cas en què hem d'emetre dos possibles missatges, $S = \{A_1, A_2\}$ en què A_1 = "Fa sol", A_2 = "Plou". El canal pel qual ha de circular la transmissió és binari, és a dir, l'alfabet serà $\mathbb{F}_2 = \{0,1\}$.

Un codi bloc per a S pot ser $C(2,3) = \{A_1 \to (0,0,0);\ A_2 \to (1,1,1)\}$.

Definició 3.2 (Distància de Hamming).

Donats dos elements $x, y \in \mathbb{F}^n$, definim la seva distància de Hamming com:

$$d_H(x,y) = \#\{i : 1 \leq i \leq n, x_i \neq y_i\},$$

és a dir, la distància entre dos vectors x i y és la quantitat de components diferents que tenen entre ells.

Distància i mètrica

La distància de Hamming satisfà les propietats de la definició matemàtica de distància i defineix una mètrica.

$\forall x,y,z \in \mathbb{F}^n$:
1) $d_H(x,x) = 0$
2) $d_H(x,y) = d_H(y,x)$
3) $d_H(x,y) + d_H(y,z) \leq d_H(x,z)$

Definició 3.3 (Distància mínima).

Donat un codi $C(M,n)$, definirem la distància mínima d, del codi, com:

$$d = min\{d_H(x,y) : x \neq y, x,y \in C\}.$$

Definició 3.4 (Regla de descodificació a distància mínima).

Donat un codi $C(M,n)$, definirem la regla de descodificació a distància mínima com la que descodifica un vector rebut $u \in \mathbb{F}^n$ per la paraula codi que està a mínima distància d'aquest. És a dir, estableix una aplicació $\mathbb{F}^n \longrightarrow C$, tal que si $u \longrightarrow v$, resulta que v és la paraula codi per a la qual $d_H(v,u) = min\{d_H(v',u), \forall v' \in C\}$.

Definició 3.5 (Capacitat correctora).

Podem considerar en $\mathbb{F}^n$ les boles centrades en les paraules codi de radi el màxim valor possible de manera que les boles siguin disjuntes. El radi d'aquestes boles es pot calcular com $c = \lfloor \frac{d-1}{2} \rfloor$ i aquest valor s'anomena *capacitat correctora del codi*.

Direm que el codi **és *c*-corrector**.

3.2.2. Codis lineals

Els codis lineals permeten una bona solució al problema de la codificació/descodificació, considerant la regla de descodificació a distància mínima.

Considerem l'estructura d'espai vectorial de $\mathbb{F}^n = \{u = (u_1, \cdots, u_n) : u_i \in \mathbb{F}\}$, de dimensió n, sobre $\mathbb{F}$, amb les operacions suma i producte per escalars habituals en un espai vectorial:

$$u + v = (u_1 + v_1, \cdots, u_n + v_n),\ u,\ v \in \mathbb{F}^n$$
$$k \cdot (u_1, \cdots, u_n) = (k \cdot u_1, ..., k \cdot u_n);$$

Definició 3.6 (Codi lineal).

Un codi bloc s'anomenarà *lineal* si és subespai vectorial de $\mathbb{F}^n$. Si la dimensió d'aquest subespai és k, el codi contindrà $M = q^k$ paraules codi, de longitud n, i serà denotat $C(n,k)$.

Definició 3.7 (Pes d'un vector).

El pes $w_t(v)$ d'un vector $v \in \mathbb{F}^n$ és el nombre de coordenades no nul·les d'aquest vector. És a dir:

$$w_t(v) = \#\{v_i \neq 0 : v_i \in \mathbb{F}\}.$$

Definició 3.8 (Error de transmissió).

Un error de transmissió es correspon amb un canvi de coordenada entre la paraula codi d'entrada i el vector de sortida.

Aquestes característiques ens donen la capacitat detectora i correctora d'errors.

Lema 3.9.

Un codi lineal, amb distància mínima $d \geq 2c + 1$, pot detectar fins a $d - 1$ errors i corregir-ne fins a c si s'utilitza l'esquema de descodificació a distància mínima.

Definició 3.10 (Matriu generadora).

De la definició de codi lineal resulta que tot conjunt de k vectors de $\mathbb{F}^n$, linealment independents, constitueix una base d'un codi lineal $C(n,k)$. Així, tot codi lineal pot ser definit per una matriu $k \times n$, en què les k files són els k vectors independents, d'una certa base de C.

Una matriu tal, denotada $G_{k\times n}$, és anomenada *matriu generadora del codi*, ja que tota paraula codi v és una combinació lineal de les k files d'aquesta matriu.

Podem escriure matricialment:

$$v = a \cdot G$$

i donant a $a = (a_1, a_2, \cdots, a_k)$ tots els valors possibles (q^k en total), obtenim totes les paraules codi de C.

Coincidència dels valors de pes i de distància

Un codi lineal té la propietat que la suma de dues paraules codi és una paraula codi; per tant:
$d_H(u,v) =$
$\#\{i : u_i \neq v_i\} =$
$\#\{i : u_i - v_i \neq 0\} =$
$w_t(u - v)$
Així, en un codi lineal, la distància entre dues paraules codi és igual al pes d'una altra paraula codi i, en conseqüència, la distància mínima, no nul·la, coincideix amb el pes mínim del conjunt de paraules codi no nul·les.

Definició 3.11 (Matriu de control).

Un codi lineal $C(n,k)$ pot ser descrit, a més, per una altra matriu. En efecte, el subespai ortogonal al codi, el qual és de dimensió $n-k$, pot ser descrit per una matriu, H, en què les $n-k$ files són els $n-k$ vectors linealment independents d'aquest subespai, ortogonals a tots els vectors de C. O sigui: $H^T_{(n-k)\times n} \cdot G_{k\times n} = 0_{(n-k)\times k}$.

Aleshores tot vector v del codi té la propietat de ser ortogonal a aquesta matriu, és a dir, verifica:

$$H \cdot v^T = 0 \Longleftrightarrow v \in C(n,k)$$

Aquesta matriu H permet controlar si un vector determinat pertany o no al codi, i per això s'anomena *matriu de control*.

Ortogonalitat

Ortogonal o perpendicular voldrà dir que el producte escalar és zero.

Exemple 3.2.

Per a construir un codi lineal binari $C(6,3)$ podem agafar la matriu generadora següent:

$$G_{3\times 6} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

La qual ens permet fer la codificació següent:

a	$v = a \cdot G$	$w_t(v)$
(0,0,0)	(0,0,0,0,0,0)	0
(1,0,0)	(1,1,1,0,0,0)	3
(0,1,0)	(0,1,0,0,1,1)	4
(0,1,1)	(0,1,0,0,1,1)	3
(0,0,1)	(0,0,1,1,1,0)	3
(1,1,0)	(1,0,0,1,0,1)	3
(1,0,1)	(1,1,0,1,1,0)	4
(1,1,1)	(1,0,1,0,1,1)	4

El pes mínim del codi, que coincideix amb la distància mínima, és 3 i, per tant, podrà detectar fins a 2 errors de transmissió però només en podrà corregir 1.

Per altra banda, es pot verificar que la matriu $H_{3\times 6}$ és una matriu de control per al codi $C(6,3)$ anterior, ja que les tres files de H són una base del subespai ortogonal a C.

$$H_{3\times 6} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Donat un codi lineal $C(n,k)$ i un vector $e \in \mathbb{F}^n - C$ construïm el conjunt $C + e = \{v + e : v \in C\}$, anomenat **coset** (o traslladat) de C, de líder (o representant) e, que conté q^k vectors distints de $\mathbb{F}^n$.

Teorema 3.12.

Un codi lineal $C(n,k)$ admet q^{n-k} cosets diferents, que constitueixen una partició de tot l'espai $\mathbb{F}^n$.

Demostració: Considerem la relació d'equivalència sobre $\mathbb{F}^n$, induïda pel codi C, definida com:

$$e,u \in \mathbb{F}^n;\ u \backsim e \Longleftrightarrow u - e \in C$$

que ens indueix la partició de $\mathbb{F}^n$ en classes d'equivalència, en què dos vectors de $\mathbb{F}^n$ són equivalents si i només si pertanyen a la mateixa classe. De fet, tenim:

$$u - e \in C \Longleftrightarrow u \in C + e$$

d'on deduïm que les classes d'equivalència coincideixen amb els cosets. ∎

Òbviament, cada coset conté q^k elements (tants com paraules codi).

Definició 3.13 (L'aplicació síndrome).

Donat un codi lineal $C(n,k)$, considerem la seva matriu de control H. Aquesta matriu de mida $(n-k) \times n$ ens permet definir una aplicació lineal, de l'espai $\mathbb{F}^n$ al subespai de les $(n-k)$-es de $\mathbb{F}$, o sigui $\mathbb{F}^{n-k}$:

$$S : \mathbb{F}^n \longrightarrow \mathbb{F}^{n-k},$$

tal que cada vector $u \in \mathbb{F}^n$ es transforma en $S(u) = H \cdot u^T$. Aquest valor $S(u)$ rep el nom de **síndrome del vector u**.

Lema 3.14.

Hi ha una correspondència biunívoca entre els q^{n-k} cosets possibles i les q^{n-k} síndromes possibles.

A causa d'aquesta correspondència biunívoca entre cosets i síndromes, i havent fet l'elecció dels líders dels cosets com un vector de pes mínim dins de

cada coset, la síndrome d'un vector qualsevol caracteritza el líder del coset de manera única, i per tant, podem considerar la redefinició de la regla de descodificació a distància mínima:

Definició 3.15 (Regla de descodificació via la síndrome).

Si $u \in \mathbb{F}^n$ és el vector rebut:

1) Calcular $S(u) = H \cdot u^T$

2) Determinar el líder e tal que $S(e) = H \cdot e^T = S(u)$

3) Descodificar u per la paraula codi $v^* = u - e$.
(Efectivament, $v^* \in C$, ja que $H \cdot (v^*)^T = H \cdot (u-e)^T = H \cdot u^T - H \cdot e^T = S(u) - S(e) = 0$).

Exemple 3.3.

Descodificar el vector $u = (1,1,0,1,0,1)$, usant la descodificació via síndrome, per al codi $C(6,3)$ de l'exercici anterior (aquest codi és 1-corrector).

$S(u) = H \cdot u^T = H \cdot (1,1,0,1,0,1)^T = (1,0,0)$

El vector $e = (0,1,0,0,0,0) \in \mathbb{F}^n - C$, compleix que $S(e) = S(u)$:

$$S(e) = H \cdot e^T = H \cdot (0,1,0,0,0,0)^T = (1,0,0)$$

Així, l'estimació que fem de la paraula codi enviada és:

$$v^* = u - e = (1,1,0,1,0,1) - (,0,1,0,0,0,0) = (1,0,0,1,0,1) \in C$$

Correcció de l'error:

Si només s'ha produït un error, aleshores estem segurs que $v^* = v$ és realment la paraula codi que s'havia enviat. En cas contrari, si s'ha produït més d'un error, aleshores la descodificació, tot i que $v^* \in C$, seria incorrecta.

3.2.3. Els codis lineals cíclics: *BCH* i *RS*

Per a fer l'estudi d'aquestes famílies de codis associarem als vectors de l'espai $\mathbb{F}^n$ un polinomi, de grau inferior o igual a $n-1$, tal que els seus coeficients coincideixin amb les coordenades del vector. Aleshores un codi lineal cíclic podrà ser considerat com un ideal de l'anell de polinomis de grau inferior o igual a $n-1$ i coeficients en $\mathbb{F}$ (denotarem aquest anell per $\mathbb{F}^n[X] \approx \mathbb{F}[X]/X^n - 1$).

Definició 3.16 (Codis cíclics).

Un codi C de longitud n és anomenat *cíclic* si tota permutació cíclica d'una paraula codi és també una paraula codi. És a dir:

$$\forall v = (v_0, v_1, \cdots, v_{n-1}) \in C \implies v^{\pi} = (v_{n-1}, v_0, \cdots, v_{n-2}) \in C$$

Per tal de poder estudiar les propietats algebraiques d'aquests codis, és còmode utilitzar una descripció polinòmica. A tota paraula codi li associem un polinomi de la manera següent:

$$v = (v_0, v_1 \ldots, v_{n-1}) \in C \implies v(X) = v_0 + v_1 X + \ldots + v_{n-1} X^{n-1}$$

Si notem per $C(X)$ el conjunt de polinomis associats a les paraules codi de C, observeu que: $v^{\pi}(X) = X \cdot v(X) - v_{n-1} X^n$; és a dir, que: $v^{\pi}(X) = X \cdot v(X) \pmod{X^n - 1}$.

Lema 3.17.

$\mathbb{F}^n[X]$, amb la suma habitual de polinomis i el producte $a(X)^* b(X) = a(X) \cdot b(X) \pmod{(X^n - 1)}$ té estructura d'anell commutatiu.

Lema 3.18.

Un codi lineal $C(n,k)$ és cíclic $\iff$ $C(X)$ és un ideal de $\mathbb{F}^n[X]/(X^n - 1)$.

Aquests dos lemes anteriors ens permeten escriure el resultat següent, que és la base de la construcció dels codis lineals i cíclics.

Teorema 3.19.

Sigui C un codi lineal i cíclic de longitud n (ideal de $\mathbb{F}^n[X]/(X^n - 1)$). Sigui $g(X)$ un polinomi mònic (el coeficient del terme de grau més gran és 1) de grau més petit dins $C(X)$. Sigui r el grau de $g(X)$. Aleshores s'acompleix:

1) $g(X)$ és l'únic polinomi mònic de grau r dins $C(X)$.

2) $g(X)$ genera $C(X)$ com a ideal principal en $\mathbb{F}^n[X]/(X^n - 1)$ (és a dir, $\forall v(X) \in C(X)$ existeix $h(X)$ tal que $v(X) = g(X) * h(X)$).

3) $g(X)$ divideix $X^n - 1$ (amb n la longitud del codi).

4) $\{X^i \cdot g(X), 0 \leq i \leq (n - r - 1)\}$, genera $C(X)$ com a subespai vectorial. és a dir, $\forall v(X) \in C(X)$ existeixen certs coeficients a_i per als quals $v(X) = \sum_{i=0}^{n-r-1} a_i X^i \cdot g(X) \pmod{X^n - 1}$

Aquest teorema ens permet assegurar que tot polinomi $g(X) \in \mathbb{F}^n[X]$, de grau r, divisor de $X^n - 1$ genera un codi lineal i cíclic $C(n,k)$ que té per **matriu generadora**:

$$G_{k\times n} = \begin{pmatrix} g_0 & g_1 & \cdots & g_{n-1} & \cdots & 0 & 0 & 0 \\ 0 & g_0 & g_1 & \cdots & g_{n-1} & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \cdots & g_0 & g_1 & \cdots & g_{n-1} \end{pmatrix}.$$

és a dir, les files de $G_{k\times n}$, en què $k = n - r$, són els coeficients de $X^i \cdot g(X)$ (mod $X^n - 1$).

Observeu que multiplicar $g(X)$ per X^i (mod X^n–1) equival a desplaçar a la dreta (amb retroalimentació) i posicions els coeficients de $g(X)$, $i = 1, \cdots, n - r - 1$, entès com un vector de $\mathbb{F}^n$.

3.2.4. Els codis cíclics BCH

Els codis BCH van ser introduïts per Hocquenghem (1959) i Bose i Chaudhuri (1960), en què l'estructura algebraica es basa en els cossos finits, $\mathbb{F}_{p^m}$. En el nostre cas agafarem $p = 2$.

Aquest tipus de codis estan definits per dos paràmetres m i c i verifiquen:

Teorema 3.20.

Per a tot enter n de la forma $n = 2^m - 1$, $m \geq 3$, i tot enter c tal que $n - c \cdot m > 0$, existeix un codi cíclic c-corrector, de longitud n, dimensió $k \geq n - c \cdot m$ i distància mínima $d \geq 2 \cdot c + 1$, que té per polinomi generador:

$$g(X) = mcm(m_1(X), m_3(X), \ldots, m_{2\cdot c-1}(X))$$

amb $m_i(X)$ com a polinomi mínim de α^i i α un element primitiu de $\mathbb{F}_{2^m}$,

Polinomi mínim de α^i

Sigui α un element primitiu de $\mathbb{F}_{2^m}$, Sigui t tal que $(\alpha^i)^t = 1$ i sigui s el més petit enter tal que t divideix $2^s - 1$. Aleshores, el polinomi mínim de α^i es pot calcular com:

$$m_i(X) = \Pi_{j=0}^{s-1}(X - (\alpha^i)^{2^j})$$

Definició 3.21 (Els codis cíclics BCH).

Un codi amb les característiques del teorema anterior s'anomena **codi BCH primitiu**.

La matriu de control dels codis BCH és:

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^3 & \alpha^6 & \cdots & \alpha^{3(n-1)} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & \alpha^{2c-1} & \alpha^{(2c-1)\cdot 2} & \cdots & \alpha^{(2c-1)\cdot(n-1)} \end{pmatrix},$$

ja que $v(X) = v_0 + v_1X + ... + v_{n-1}X^{n-1}$, $v_i \in \mathbb{F}$, serà el polinomi associat a una paraula codi v, si i només si, $v(\alpha^i) = 0, \forall i = 1,3,\cdots,2c-1$. És a dir, les paraules codi són múltiples del polinomi $g(X)$ que té, per construcció, com a zeros els elements $\alpha,\alpha^3,\cdots,\alpha^{2c-1}$ (i, també, $\alpha^2,\alpha^6,\cdots,\alpha^{2c}$).

En la taula següent podem veure el paràmetres d'alguns codis BCH.

n	t	k	$g(X)$
7	1	4	$X^3 + X + 1$
–	2	1	$X^6 + X^5 + X^4 + X^3 + X^2 + X + 1$
15	1	11	$X^4 + X + 1$
–	2	7	$X^8 + X^7 + X^6 + X^4 + 1$
–	3	5	$X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1$
31	1	26	$X^5 + X^4 + X^2 + 1$
–	2	21	$X^{10} + X^9 + X^8 + X^6 + X^5 + X^3 + 1$
–	3	16	$X^{15} + X^{11} + X^{10} + X^9 + X^8 + X^7 + X^5 + X^3 + X^2 + X + 1$
–	5	11	$X^{20} + X^{18} + X^{17} + X^{13} + X^{10} + X^9 + X^7 + X^6 + X^4 + X^2 + 1$
–	7	6	$X^{25} + X^{24} + X^{21} + X^{19} + X^{18} + X^{16} + X^{15} + X^{14} + X^{13} + X^{11} + X^9 + X^5 + X^2 + X + 1$

3.2.5. El codis cíclics RS

Reed i Solomon van introduir el 1960 una classe particular de codis BCH que millorava, encara més, les prestacions de correcció i la facilitat de descodificació. En particular, els codis d'aquesta família són els que tenen la millor capacitat detectora i correctora, donada la parella de paràmetres n i k.

Definició 3.22 (Els codis cíclics RS).

Un codi de Reed-Solomon (RS) primitiu és un codi cíclic sobre $\mathbb{F}_{2^m}$, de longitud $n = 2^m - 1$ i dimensió k, amb distància mínima $d = n - k + 1$, i que té per polinomi generador:

$$g(X) = (X - \alpha) \cdot (X - \alpha^2) \cdots (X - \alpha^{d-1}),$$

en què α és un element primitiu de $\mathbb{F}_{2^m}$.

Correcció de paquets d'errors

Els símbols de les paraules codi són elements $\alpha^i \in \mathbb{F}_{2^m}$, la qual cosa vol dir que cada símbol que es transmet pel canal és un element de m coordenades binàries. Per tant, un codi de Reed-Solomon (n,k), en realitat transmet $m \cdot k$ bits d'informació mitjançant una paraula codi de $n \cdot m$ bits. En conseqüència, una consideració important és que els errors ara no necessàriament han de ser bits aleatoris, sinó que podem considerar que, cada error és un paquet de m bits. Aquesta característica millora la capacitat correctora global, ja que en realitat pot corregir fins a c paquets de m errors (cada error és un canvi d'un α^i per un α^j).

Nota

Un codi lineal sempre satisfà que $d \leq n-k+1$ (fita de Singleton). Un codi que satisfà la igualtat s'anomena *MDS* **(màxima distància separable)**. Evidentment, els codis RS són de màxima distància separable.

La matriu de control dels codis RS és:

$$H_{(n-k)\times n} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \ldots & \alpha^{2(n-1)} \\ 1 & \alpha^3 & \alpha^6 & \ldots & \alpha^{3(n-1)} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & \alpha^{d-1} & \alpha^{2(d-1)} & \ldots & \alpha^{(d-1)\cdot(n-1)} \end{pmatrix},$$

ja que $v(X) = v_0 + v_1 X + \ldots + v_{n-1} X^{n-1}$, $v_i \in \mathbb{F}^m$. En aquest cas, $v(X)$ serà el polinomi associat a la paraula codi v, si i només si $v(\alpha^i) = 0, \forall i = 1,2,\cdots,d-1$. És a dir, les paraules codi són múltiples del polinomi $g(X)$ que, per construcció, té per zeros els elements $\alpha, \alpha^2, \cdots, \alpha^{d-1}$.

El 1969, Berlekamp i Massey varen donar un algorisme molt eficient de descodificació, basat en el teorema de Dirichlet. Aquests codis han estat àmpliament utilitzats en sistemes d'emmagatzematge de dades (CD, DVD...), en mòdems d'alta velocitat (ADSL, DSL...) televisió digital (TDT, MPEG2, MPEG4...) i també proposats per a ser usats en criptografia.

3.3. Els criptosistemes de McEliece i de Niederreiter

En aquest subapartat veurem els dos criptosistemes de McEliece i de Niederreiter, en què la funció unidireccional es basa en codis correctors d'errors.

3.3.1. Criptosistema de McEliece

Robert McEliece va proposar el 1978, un criptosistema de clau pública fonamentat en el fet que l'algorisme de descodificació d'un codi lineal, en general, no és computacionalment eficient (R. J. McEliece (1978). "A public-key cryptosystem based on algebraic coding theory". *DSN Progress Report* (pàg. 42-44)).

En la seva proposta va utilitzar els codis de Goppa (de la mateixa família de codis que els BCH i RS, els anomenats **codis alternants**), dels quals es coneix un algorisme de descodificació computacionalment eficient. En particular, l'algorisme de descodificació de Patterson, que té un funcionament molt semblant al de Berlekamp-Massey.

La idea principal del criptosistema és transformar la matriu generadora, $G_{k\times n}$, del codi alternant escollit, $C(n,k)$, c-corrector, i transformar-la en una matriu generadora d'un nou codi lineal $C'(n,k)$, que notarem per $G'_{k\times n}$ (que també serà c-corrector).

Per a dur a terme aquesta transformació, s'utilitzen dues matrius; una matriu binària, no singular, $S_{k\times k}$ i, per tant, invertible, i una matriu de permutació $P_{n\times n}$.

Així, la **clau privada** serà $G_{k\times n}$ i les matrius $S_{k\times k}$ i $P_{n\times n}$, mentre que la **clau pública** serà: $G'_{k\times n} = S_{k\times k} \cdot G_{k\times n} \cdot P_{n\times n}$ i el paràmetre c (la capacitat correctora del codi $C(n,k)$).

Suposem un usuari B que vol enviar un missatge xifratge a l'usuari A, que té a G', tal com hem descrit abans, com a clau pública:

- **Algorisme de xifratge**. Si m és el missatge que ha de xifrar B, el dividirà en blocs de k símbols. Sigui m_i un d'aquests blocs; aleshores el criptograma corresponent serà:

$$c_i = E_{G'}(m_i) = m_i \cdot G' + e,$$

  i $e \in \mathbb{F}^n$ és un vector d'error arbitrari, escollit per B, tal que $w_t(e) \leq c$.

- **Algoritme de desxifratge**. Aquest algorisme fa ús del coneixement, per part de l'usuari A, de S^{-1} i P^{-1}. Així, rebut c_i, l'usuari A farà:

  1) Calcular $c'_i = c_i \cdot P^{-1} = (m_i \cdot G' + e) \cdot P^{-1} = m_i \cdot S \cdot G + e \cdot P^{-1}$, en què $e \cdot P^{-1}$ és un error que pot corregir G, ja que evidentment $w_t(e \cdot P^{-1}) = w_t(e)$, ja que la permutació no canviarà el nombre de coordenades no nul·les.

  2) Aplicar a c'_i l'algorisme de descodificació del codi $C(n,k)$. Aquest corregirà l'error $e \cdot P^{-1}$ i ens retornarà el vector $m_i \cdot S \in \mathbb{F}^k$.

  3) Multiplicant, ara, el vector rebut per S^{-1} retrobarem m_i. Efectivament, $m_i = m_i \cdot S \cdot S^{-1}$.

Malgrat que es tracta d'un criptosistema en el qual els processos de xifratge i desxifratge són relativament ràpids, actualment gairebé no s'utilitza. Això és degut principalment a les seves mides de clau (2^{19} bits per a la clau pública) i al factor d'expansió del missatge, que fa que el criptograma tingui una mida un 60% més gran que el missatge original. No obstant això, atès que l'algorisme de Shor no afecta aquest criptosistema, sembla que ofereix resistència a la criptoanàlisi basada en computació quàntica.

3.3.2. Criptosistema de Niederreiter

H. Niederreiter el 1986 va proposar un criptosistema, dual al de McEliece, basat en l'ús de codis GRS (*generalized Reed-Solomon*, estretament lligats als codis de Goppa, i definits per la seva matriu de control. La proposta d'utilització d'aquests codis es basa en la possibilitat de reduir la mida dels paràmetres respecte de la dels de Goppa (H. Niederreiter (1986). *Knapsack-type cryptosystem and algebraic coding theory. Problems of Control and Information Theory*).

En la seva proposta Niederreiter va utilitzar els codis GRS, dels quals es coneix un algorisme de descodificació computacionalment eficient. La idea principal és transformar la matriu de control, $H_{(n-k)\times n}$, del codi escollit, sobre $\mathbb{F}_{2^m}$, c-corrector ($c = \lfloor (d-1)/2 \rfloor$), i transformar-la en una matriu de control d'un nou codi codi lineal, que notarem per $H'_{(n-k)\times n}$ (que també serà c-corrector).

Per a dur a terme aquesta transformació s'utilitzen dues matrius: una matriu binària, no singular, $S_{(n-k)\times(n-k)}$ i, per tant, invertible, i una matriu de permutació $P_{n\times n}$.

Així, la **clau privada** serà $H_{(n-k)\times n}$ i les matrius $S_{(n-k)\times(n-k)}$ i $P_{n\times n}$, mentre la **clau pública** serà $H'_{k\times n} = S_{(n-k)\times(n-k)} \cdot H_{(n-k)\times n} \cdot P_{n\times n}$ i el paràmetre c (la capacitat correctora del codi $C(n,k)$).

Suposem un usuari B que vol enviar un missatge xifratge a un altre usuari A, que té H', tal com hem descrit abans, com a clau pública:

- **Algorisme de xifratge**. Si m és el missatge que ha de xifrar B, el dividirà en blocs m_i de n símbols, tals que $w_t(m_i) \leq c$. Aleshores el criptograma corresponent serà:

$$c_i = (H') \cdot (m_i)^T \in \mathbb{F}_{2^m},$$

  que dóna la síndrome de m_i.

- **Algorisme de desxifratge**. Aquest algorisme fa ús del coneixement, per part de l'usuari A, de S^{-1} i P^{-1}. Així, rebut c_i, l'usuari A farà:

  1) Calcular $c'_i = S^{-1} \cdot c_i = S^{-1} \cdot H' \cdot (m_i)^T = H \cdot P \cdot (m_i)^T = H \cdot (m'_i)^T$, en què c'_i és la síndrome de $m'_i = m_i \cdot P^T$, calculada a partir de H.

  2) Aplicar l'algorisme de descodificació a c'_i, per a trobar $m_i \cdot P^T$.

  3) $m_i = m_i \cdot P^T \cdot (P^{-1})^T$

La taula següent mostra alguns resultats interessants sobre l'eficiència dels criptosistemes de McEliece i Niederreiter.

Equivalència dels dos criptosistemes

Yuan Xing Li i altres van demostrar que els criptosistemes de McEliece i Niederreiter són equivalents, en termes de seguretat (Y. Xing Le; R. H. Deng; X. Mei Wang (1994). "On the equivalence of McEliece's and Niederreiter public-key cryptosystems". *IEEE Transaction on Information Theory*).

	(n,c)	(2048,32)	(2048,40)	(4096,22)	(4096,45)
McEliece	text original	1928	1888	4024	3904
	criptograma	2048	2048	4096	4096
	mida clau pública	73 *kb*	86 *kb*	123 *kb*	234 *kb*
Niederreiter	text original	232	280	192	352
	criptograma	352	4408	264	540
	mida clau pública	73 *kb*	86 *kb*	123 *kb*	234 *kb*

Lectura recomanada

R. Overbeack; N. Sendrier. (2009). "Code-based cryptography". A: D. Bernstein; J. Buchmann; J. Ding (eds.). *Post-Quantum Cryptography* (pàg. 95-145). Springer.

Exercicis d'autoavaluació

1. Un usuari d'una xarxa ha rebut el criptograma 1611,3556,4744,3504, resultat de xifrar caràcters, individualment, de $M = \{A,B,\cdots,Y,Z\} \approx \{A = 02,B = 03,\cdots,Y = 26,Z = 27\}$; emprant el criptosistema RSA amb la clau pública $n = 7597$ i $e = 4947$. Trobeu el missatge original.

2. Utilitzeu un criptosistema RSA per a dos usuaris A i B, amb el mateix valor de

$$n = 151953280470109$$

i claus públiques, respectivament, $e_A = 17$ i $e_B = 19$. Suposeu que l'usuari A vol xifrar, per enviar a B, el text:

m = els ordinadors quantics poden deixar obsolets els metodes actuals de xifratge

a) Quin serà el resultat del xifratge?
b) Quina seria la signatura *RSA* del missatge?
c) Simuleu la verificació de la signatura per part de B.

3. A partir d'un cos finit $\mathbb{F}_{71}$ de 71 elements i un element primitiu $\alpha = 7$, considerem un criptosistema ElGamal.
a) Agafant $k = 2$, si el criptograma corresponent a $m = 30$ és $c = (57,49)$, trobeu el valor de la clau pública utilitzada.
b) Si s'agafa un altre valor de k, i resulta que per al mateix valor de m obtenim $c = (b,59)$, quin és el valor de b?

4. Construïu un criptosistema ElGamal per a dos usuaris A i B, en un cos $\mathbb{F}_p$, amb $p = 1231451311$ i $\alpha = 21$. Suposeu que l'usuari A té per clau privada $x_A = 113$ i que l'usuari B té per clau privada $x_B = 97$. Suposeu el text: $m = HOLA$, emprant els símbols de $M = \{A,B,\cdots,Y,Z\} \approx \{A = 02,B = 03,\cdots,Y = 26,Z = 27\}$
a) Quin serà el resultat del xifratge de m, agafant el valor de $k = 247$, que A enviarà a B? Feu el desxifratge corresponent per a retrobar m.
b) Quina seria la signatura del missatge, per part d'A?
c) Simuleu la verificació de la signatura per part de B.

5. Sobre la signatura DSA:
a) Suposem $p = 124540019$, $q = 17389$,$g = 10083255$, $x = 12496$ i $k = 9557$. Agafant la funció resum fictícia, $h(m) = m$, comproveu que la signatura de $m = 5246$ és $(r,s) = (13752,9137)$.
b) Simuleu-ne la verificació.

6. L'Anna i en Bernat decideixen utilitzar el protocol quàntic de Bennett-Brassard per a intercanviar-se una clau de sessió. Al començament decideixen que les polaritzacions \ i | indicaran un **1** i les polaritzacions / i — indicaran un **0**.

L'Anna envia 20 fotons a en Bernat. L'Ernest, que és un espia que està al cas de la comunicació entre A i B, els intercepta tots i els reenvia a B amb la mateixa polarització en què els ha detectat.

Posteriorment A i B es comuniquen públicament entre ells i decideixen que els bits 1, 2, 3, 4, 5, 7, 8, 9, 10, 20 seran desestimats ja que, en aquests, no han fet servir la mateixa polarització. La polarització dels bits restants, tal i com l'ha deduïda B, és: / / / / / | | | | |.

Sabent que els bits 13, 14, 15, 16, que inicialment havia enviat A són, respectivament, 1, 0, 0, 1, sabríeu dir amb quina polaritat A ha enviat els fotons que representen aquests bits? I quin polaritzador ha utilitzat l'Ernest en aquests bits? Per què?

Solucionari

1. Per a resoldre el problema ens ajudarem del programari SAGE. En primer lloc cal factoritzar n:

```
sage: n = 7597
sage: factor(n)
71 * 107
```

A continuació trobarem l'invers $e = 4947$ mòdul $\varphi(n) = 70 * 106 = 7420$:

```
sage: e = 4947
sage: phi = 7420
sage: d = inverse_mod(e,phi)
sage: print d
3
```

Trobarem el missatge original desxifrant el missatge rebut amb la funció *power_mod*(C,d,n):

```
sage: power_mod(1611,d,n)
2
sage: power_mod(3556,d,n)
3
sage: power_mod(4744,d,n)
4
sage: power_mod(3504,d,n)
5
```

Finalment, $m = ABCD$.

2. Utilitzarem el programari SAGE per simplificar els càlculs. Primer de tot necessitem passar del text alfabètic a missatges numèrics que ens permetin efectuar les operacions RSA. Per a fer-ho necessitem definir algunes operacions prèvies, les de codificar/descodificar lletres, codificar/descodificar cadenes, xifrar/desxifrar nombres, xifrar/desxifrar missatges. Donem per fet que la programació elemental utilitzada és coneguda per l'estudiant:

```
alphabet = ' abcdefghijklmnopqrstuvwxyz'
L = len(alphabet)
def codifica_char(lletra):
    return alphabet.index(lletra)

def descodifica_char(n):
    return alphabet[n]

def codifica_text(missatge):
    return [codifica_char(c) for c in text]

def descodifica_chain(llista):
    return ''.join([descodifica_char(v) for v in llista])

def chiper_RSA(llista,n,e):
    return [power_mod(valor,e,n) for valor in llista]

def unchiper_RSA(llista,n,d):
    return [power_mod(valor,d,n) for valor in llista]
```

a) Amb aquestes definicions prèvies podem xifrar el missatge que ens donen:

```
sage: m = 'els ordinadors quantics poden deixar obsolets els metodes actuals de xifratge'
sage: n = 151953280470109
sage: e = 17
sage: chiper_RSA(codifica_text(m),n,e)

[762939453125, 93211876845592, 32590367823381, 0, 74138716094102, 103704942061406, 17179869184,
114274128424688, 97563638183746, 1, 17179869184, 74138716094102, 103704942061406, 32590367823381,
0, 69016003153490, 33625452816007, 1, 97563638183746, 105460815098081, 114274128424688, 129140163,
32590367823381, 0, 83278712378725, 74138716094102, 17179869184, 762939453125, 97563638183746, 0,
```

```
17179869184, 762939453125, 114274128424688, 119465547730806, 1, 103704942061406, 0, 74138716094102,
131072, 32590367823381, 74138716094102, 93211876845592, 762939453125, 105460815098081, 32590367823381,
0, 762939453125, 93211876845592, 32590367823381, 0, 19568778972781, 762939453125, 105460815098081,
74138716094102, 17179869184, 762939453125, 32590367823381, 0, 1, 129140163, 105460815098081,
33625452816007, 1, 93211876845592, 32590367823381, 0, 17179869184, 762939453125, 0, 119465547730806,
114274128424688, 16926659444736, 103704942061406, 1, 105460815098081]
```

b) Per a signar el missatge hem de tenir calculat el valor d que forma part de la clau privada i que l'enunciat del problema no ens dóna. Com que el nombre n no és gaire llarg, el podem factoritzar:

```
sage: factor(151953280470109)
1738934123 * 87383
```

Ara podem calcular $\varphi(n) = 1738934122 * 87382$ i l'invers de e mòdul $phi(n)$.

```
sage: e = 17
sage: phi = 1738934122 * 87382
sage: d = inverse_mod(e,phi)
print d
8938325967565
```

Llavors la signatura la podem trobar amb la mateixa funció que la xifra, però substituint el paràmetre e per d:

```
sage: missatge_signat = chiper_RSA(codifica_text(m),n,d)

[42782583250323, 30308012709953, 132858054847474, 0, 94150478556282,
30541430930646, 108907931565622, 76520780070238, 18215677713314, 1,
108907931565622, 94150478556282, 30541430930646, 132858054847474, 0,
44035504519670, 121144087766341, 1, 18215677713314, 16213728552100,
76520780070238, 84208423477578, 132858054847474, 0, 26232893409272,
94150478556282, 108907931565622, 42782583250323, 18215677713314, 0,
108907931565622, 42782583250323, 76520780070238, 54058519904057, 1,
30541430930646, 0, 94150478556282, 77779196084924, 132858054847474,
94150478556282, 30308012709953, 42782583250323, 16213728552100,
132858054847474, 0, 42782583250323, 30308012709953, 132858054847474, 0,
21980253538582, 42782583250323, 16213728552100, 94150478556282,
108907931565622, 42782583250323, 132858054847474, 0, 1, 84208423477578,
16213728552100, 121144087766341, 1, 30308012709953, 132858054847474, 0,
108907931565622, 42782583250323, 0, 54058519904057, 76520780070238,
96067072003657, 30541430930646, 1, 16213728552100]
```

c) L'usuari B pot comprovar la signatura del missatge que li ha enviat l'usuari A utilitzant la clau pública d'aquell i observant que el resultat dóna el missatge inicial:

```
sage: descodifica_chain(unchiper_RSA(missatge_signat,n,e))

'els ordinadors quantics poden deixar obsolets els metodes actuals de xifratge'
```

3. a) Diguem que la clau pública és $a = \alpha^U$, llavors el missatge xifratge és $c = m \cdot (a)^k \pmod{p}$, o sigui, $57 = 30 \cdot (a)^2 \pmod{71}$ i, per tant $a = \sqrt{57 \cdot 30^{-1}} \pmod{71} = \sqrt{9} \pmod{71}$.

```
sage: mod((57 * 30^(-1))^(1/2), 71)
9
```

O sigui, la clau pública pot ser $\alpha^U = \pm 3 \pmod{71}$.

b) Si agafem un altre valor de k tenim que $\alpha^k = b$ i $30 \cdot a^k = 59$, en què a és la clau pública. Llavors $b = 30 \cdot (59)^{-1} \pmod{71} = 33$.

4. Utilitzarem el programari SAGE. Entrem les dades de l'exercici:

```
sage: p=1231451311
sage: alfa=21
sage: xA=113
sage: xB=97
sage: k=247
sage: m=9161302
sage: R=IntegerModRing(p)
```

Amb la darrera instrucció indiquem que treballarem en l'anell $\mathbb{Z}_p$. Ara ja podem calcular:

a)
```
sage: YB=R(alfa^xB)
print YB
4198807
sage: YA=R(alfa^xA)
print YA
120638760
sage: K= R(alfa^k)
print K
840316018
```
El criptograma c, corresponent a m, es troba fent:
```
sage:c=R(m*YB^k)
print c
399347538
```
Aleshores, A envia c = 399347538 a B. Ara, per a retrobar el missatge enviat, B farà:
```
sage: beta= R(K^xB)
sage: mr=R(c/beta)
print mr
9161302
```
Efectivament el missatge rebut mr coincideix amb l'enviat m.

b) Per a poder fer la signatura A, ha de calcular a l'anell $\mathbb{Z}_{p-1}$, i per això retornem el valor enter a K:
```
sage: R1=IntegerModRing(p-1)
sage: kinv=R1(1/k)
sage: K1=Integer(K)
sage:  s=R1((m-xA*K1)*kinv)
print s
14451644
```
Per tant, el valor de la signatura de m és s = 144516444

c) Per a poder fer la verificació de la signatura anterior, fem els càlculs següent
```
sage: M=YA^k*K^s
print M
542102987
sage:N=R(alfa^m)
print N
542102987
```
Donat que els valors de M i N coincideixen, la signatura es donaria per vàlida.

5. Utilitzarem el programari SAGE. Entrem les dades de l'exercici:

```
sage: p=124540019
sage: q=17389
sage: q=10083255
sage: x=12496
sage: k=9557
sage: m=5246
```

Ara ja podem calcular:

a)
```
sage: n=(p-1)/q
print n
7162
sage: alfa=mod(g^n,p)
print alfa
57574454
sage: y= alfa^x
print y
33942608
```
Per a la signatura de m calculem:
```
sage: r=mod(alfa^k,q)
print r
13752
sage: s=mod((m+x*r)/k,q)
print s
9137
```

Per tant, la signatura (r,s) = 13752,9137) coincideix amb la donada.

b) Per a simular la verificació de la signatura procedim:

```
sage: w=mod(1/s,q)
sage: u1=mod(m*w,q)
sage: u2=mod(r*w,q)
sage: v=mod(alfa^u1*y^u2,q)
print v
13752
```

Atès que el valor de v = 13752 coincideix amb el valor de r, la signatura es donaria per vàlida.

6. La polarització en què A ha enviat els bits esmentats ha de ser coherent amb la que ha rebut B, ja que aquests bits no han estat desestimats. Per tant, A ha enviat \ / — |.

La polarització que ha utilitzat l'Ernest en el bit 13 és $\oplus$ i en el bit 15 $\otimes$. En els bits 14 i 16 pot haver utilitzat $\oplus$ o $\otimes$, indistintament.

Bibliografia

Domingo, J.; Herrera, J. ; Rifà-Pous, H. (2006). *Criptografia*. Barcelona: Ed. UOC.

Rifà, J.; Huguet, L. (1991). *Comunicación digital: Teoría matemática de la información. Codificación algebraica. Criptología*. Barcelona: Ed. Masson.

Simmons, G. J. (1992). *Contemporary cryptology: the science of information integrity*. Nova York: IEEE Press.