5a Edició Premi Poincaré 2008

Tercer premi

Criptografia. El criptosistema RSA

Autora: Mireia Mor Conejo
Centre: Col·legi Sant Josep (Sant Boi de Llobregat)
Tutora: Sra. María Isabel Hernández López

Per la presentació que fa del tema tan des del punt de vista formal com dels continguts. Intercala convenientment conceptes i explicacions amb contextualitzacions històriques i continguts matemàtics, i ho fa de manera original. És remarcable el treball pràctic que fa amb Maple.

Treball de recerca

Criptografia

El criptosistema RSA

ÍNDEX

Pàgina

0. INTRODUCCIÓ

Segurament, tothom ha jugat alguna vegada a amagar un missatge i divertir-se mentre l'altre intenta esbrinar el seu significat. Doncs bé, sense adonar-nos, nosaltres també hem utilitzat la criptografia en algun moment de la nostra vida, des del joc de nens més simple fins a fer-ne un veritable ús, com per exemple en el cas d'un *router* inal·làmbric, imposant una clau, per evitar que els veïns es puguin connectar a través del nostre.

L'objecte d'aquest treball, per tant, és la criptografia i concretament un criptosistema anomenat RSA (Rivest, Shamir i Adleman). Quan he treballat amb aquest criptosistema el meu objectiu ha estat sempre aconseguir encriptar i desencriptar de manera correcta un missatge fent servir els números com a eina principal, per a fi de que la desencriptació concordi amb el missatge que es vol enviar.

És un treball que requereix tota una sèrie de coneixements previs per poder entendre'l i seguir-lo amb facilitat, és per això que abans d'endegar-lo m'he preocupat de buscar amb l'objectiu d'ampliar, el coneixement de conceptes matemàtics com ara el de nombres primers, congruències, potències binàries... que trobarem implícitament i explícitament al treball de tal manera que sigui entenedor per a qualsevol públic.

L'elecció d'aquest treball no ha estat cap altra que la d'informar-me sobre el que és i en què consisteix la criptografia i poder descobrir com es treballa i quins mètodes o procediments matemàtics s'utilitzen per dur-la a terme. No he pretès exclusivament definir el món criptogràfic, sinó que també m'he interessat per reflectir la seva part pràctica al llarg de tota la seva història, cosa que és palesa en la part que treballo el criptosistema RSA.

L'RSA és un algorisme, un conjunt d'instruccions que serveixen per executar una tasca o resoldre un problema, que transmet un missatge entre dues persones o institucions tot amagant la informació mitjançant uns números de tal manera que si no es coneix la tècnica emprada és impossible de desxifrar-la. És un algorisme que utilitza una clau pública (que es pot difondre), i una clau privada (críptica) guardada pel propietari.

Al llarg del treball es poden distingir dos grans blocs: una part teòrica on dono cabuda a un conjunt de conceptes com el de criptosistema o esteganografia, els mecanismes de seguretat i els atacs informàtics més freqüents avui dia; i una segona part dedicada a l'RSA, del qual se n'explicarà la història, qui van ser els creadors, el funcionament, es donaran consells a l'hora d'utilitzar-lo o d'escollir-ne les claus i s'esmentaran conceptes relacionats (com la firma digital, entre altres aspectes). Totes aquestes explicacions aniran acompanyades d'exemples pràctics i senzills, il·lustracions o esquemes, per facilitar la comprensió i per tal que el lector pugui dur a terme una lectura més entretinguda.

Treballaré també la seva implementació amb un programa d'ordinador anomenat Maple, que realitza tot tipus d'operació matemàtica i que estarà present en la memòria escrita del treball. Amb aquest programa he fet servir dos mètodes per implementar l'algorisme, no obstant, un d'aquests mètodes té un problema per la qual cosa, tot i que continua estant present en el treball, explico una altra opció.

Al treball hi ha lloc també per un apartat de curiositats on parlaré de l'ús de la criptografia en llibres i els mètodes o mencions que hi apareixen, i en situacions de la vida quotidiana, on sense saber-ho fem ús de la criptografia d'una manera continuada.

L'objectiu principal del treball era treballar més d'un mètode criptogràfic, no obstant vaig interessar-me principalment per l'algorisme RSA, cosa que ha fet que el treball hagi resultat una dedicació exclusiva a aquest criptosistema.

Hem de tenir present durant tot el treball que els avenços tecnològics han evolucionat paral·lelament amb la criptografia, cosa que ha permès que actualment els missatges siguin transmesos de manera més eficaç, però no necessàriament més segura.

El que fa centenars d'anys, es considerava criptografia, ara pot ser només un joc de nens.

1. CRIPTOGRAFIA

1.1. Definició

La criptografia és un terme grec que vol dir *krypto*s, ocult i, *gráphein,* escriure. És considerada com l'art de modificar i amagar un missatge per tal d'aconseguir-ne una aparença nova i irreconeixible. Aquest missatge que assoleix la nova aparença es fa servir per al públic no autoritzat a conèixer el missatge original, ja que per a ells serà il·legible si no coneixen la tècnica utilitzada.

La seguretat d'un missatge no ha de dependre del secret del seu mètode d'ocultació, sinó de la clau utilitzada en aquest mètode[1]. La gent que desconegui aquest mètode no podrà entendre'l.

En un principi, la criptografia era considerada un art, no obstant, actualment es considera una ciència aplicada, una branca de les matemàtiques per la seva relació amb altres ciències com l'estadística, la teoria dels números, la teoria de la informació i la teoria de la complexitat computacional.

El procés de transformació del text original (missatge), en el text xifrat (criptograma), es coneix com xifrat o encriptació, i el seu procés contrari, és a dir, el de la recuperació del text original, és el desxifrat o la desencriptació.

El criptoanàlisi és considerat com la contrapartida de la criptografia. Tots dos han tingut una gran importància en la història i la continuen tenint en l'actualitat. Junts, criptografia i criptoanàlisi, constitueixen la criptologia.

[1] S'ha de saber distingir entre criptografia i codi: un codi assigna una paraula a cada missatge possible, o a cada paraula o grup de paraules, de manera que és necessari un llibre de traducció per poder descodificar i recuperar el missatge. La vertadera criptografia, en canvi, permet xifrar qualsevol missatge, a partir d'una clau establerta, que és la que s'ha de mantenir en secret.

1.2. Història i actualitat

L'home des de sempre ha fet el possible per garantir la seguretat i el secret de les seves comunicacions privades mitjançant codis secrets per aconseguir que un missatge resultés incomprensible.

L' existència de la criptografia apareix en les taules cuneïformes i en els papirs. Des de l'Antic Egipte fins al món actual d'Internet, els criptogrames han estat protagonistes de diversos successos històrics.

Els espartans, a Grècia, van desenvolupar al 400 a.C la Scitala[2]. Juli Cèsar utilitzava un mètode basat en la substitució de cada lletra per una altra que ocupa diversos llocs més enllà a l'alfabet, creant així el conegut xifrat que du el seu nom[3]. Cap dels xifrats anteriors és utilitzat en l'actualitat ja que tots ells es poden trencar fàcilment fent ús de tècniques estadístiques.

La criptografia va ressorgir a l'Edat Mitjana, va ser Grabiele de Lavinde, un servidor del Papa Clement VII qui va escriure el primer manual sobre la matèria.

Al 1446, León Batista Alberti, va crear el sistema polialfabètic, que feia servir diversos abecedaris, saltant d'un a l'altre cada tres o quatre paraules. L'emissor i el destinatari, per tant, s'havien de posar d'acord en certs aspectes per conèixer l'ordre dels salts de l'alfabet.

A partir del segle XIX, van sorgir criptoanalistes tan il·lustres com Charles Babbage[4], Friedrich Kasiski...

Una etapa de la nostra època en què la criptografia va començar a ser considerablement útil i va adquirir fama per arribar a convertir-se en el que és avui dia,

[2] Considerat el primer sistema criptogràfic per transposició.
[3] L'anomenat xifrat Cèsar.
[4] Professor de matemàtiques famós per haber disenyat màquines precursores dels ordinadors actuals.

va ser en la Primera Guerra Mundial. En aquesta època es formaven grups de treball dedicats a idear mètodes per xifrar missatges, de manera que poguessin ser transmesos per ràdio o telègraf sense por que els enemics poguessin escoltar-los, ja que es veien limitats a interceptar col·leccions de caràcters sense sentit.

Al segle xx, Arthur Scherbius va inventar la màquina criptogràfica *Enigma* que va ser utilitzada pels nazis durant tota la Segona Guerra Mundial i que van creure inviolable. No obstant, aquesta màquina va significar la seva derrota, ja que els contrincants van aconseguir desxifrar el seu funcionament[5] al 1942. La invenció d'aquesta màquina criptogràfica va proporcionar mètodes de xifrat més sofisticats i eficients.

Màquina criptogràfica Enigma

A mesura que els alemanys van dissenyar la màquina *Enigma*, els estadounidencs van utilitzar un mètode anomenat *Sigaba*. Aquest va ser l'únic mètode criptogràfic que va conservar intactes tots els seus xifrats durant la guerra, tot i que després el seu funcionament va ser també desxifrat.

Fins als anys 70, la criptografia segura era de domini gairebé exclusiu dels governs. Des de llavors, dos successos han fet que sigui de domini públic: la creació d'un estàndard de xifrat públic (DES[6]) i la invenció de la criptografia asimètrica.

[5] Va ser Marian Rejewski, el primer que va criptoanalitzar la màquina "Enigma".
[6] Correspon a l'abreviatura de Data Encryption Estándar, un mètode per xifrar informació, l'ús del qual s'ha propagat per arreu del món.

Al 1991, Philip Zimmermann, un criptògraf aficionat, va desenvolupar un sistema criptogràfic, el G.P.G i el va distribuir per les reds de comunicació per a que qualsevol el pogués utilitzar[7].

Més tard, Diffie i Hellman van proposar utilitzar criptosistemes, el criptoanàlisi dels quals fos equivalent a la resolució d'un problema computacionalment difícil. Aquest és el principi dels sistemes de clau pública entre els quals troben l'R.S.A.

Avui en dia el que es demana en criptografia és simplement rapidesa, senzillesa de càlcul a l'hora d'encriptar i desencriptar missatges i una complexitat total de càlcul per a qualsevol criptoanalista.

[7] Avui en dia és el més utilitzat en qüestions de correu electrònic.

2. CRIPTOSISTEMES

Un criptosistema es pot definir com els procediments i fonaments que participen en el xifrat i desxifrat d'un missatge.

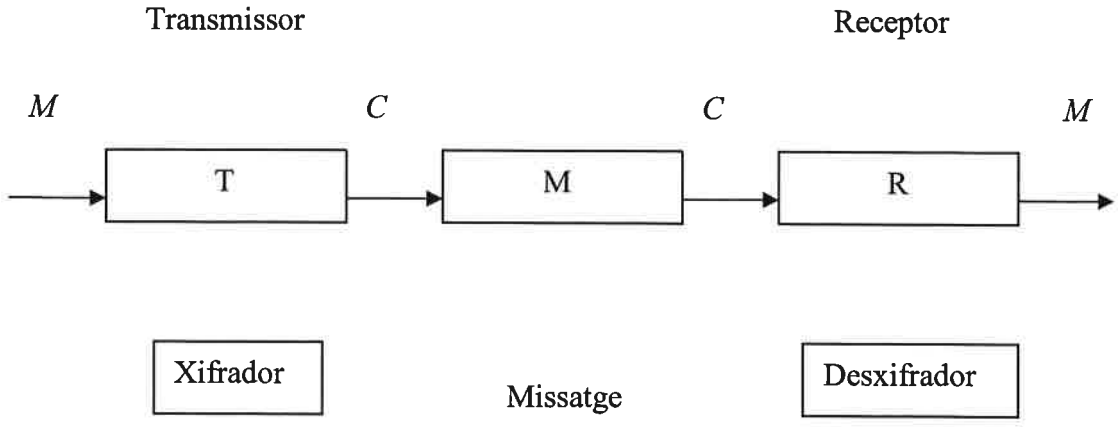
2.1. Components

En tot criptosistema han d'aparèixer cinc components:

- *M*: Conjunt de tots els missatges que es volen transmetre.
- *C*: Conjunt de tots els missatges xifrats.
- *K*: Conjunt de claus a utilitzar.
- *E*: Conjunt de tots els mètodes del xifrat.
- *D*: Conjunt de tots els mètodes del desxifrat.

Un altre element que es pot considerar com a component del criptosistema és el llenguatge del missatge original.

2.2. Visualització - Esquema



2.3. Qualitats

Un bon criptosistema ha de tenir les següents qualitats:

- Seguretat: És la incapacitat per a un criptoanalista de determinar el text original a partir del text xifrat que ha interceptat.

- Autenticitat i integritat: Està considerada com la incapacitat per a un criptoanalista d'improvisar, substituir o modificar un text xifrat per un altre, sense que el receptor ho detecti.

- No rebuig: Això vol dir que l'emissor un cop hagi enviat el missatge no pot afirmar que aquest no sigui seu. Aquí la firma digital té un paper fonamental.

Podem dividir els criptosistemes en dos grups: sistemes de clau pública i sistemes de clau privada.

3. CRIPTOSISTEMES DE CLAU SECRETA

Els criptosistemes de clau secreta es basen que l'emissor i el receptor comparteixen una única clau secreta k, de manera que el procés d'encriptació E és l'invers del de la desencriptació D, i el coneixement d'un d'ells permet el coneixement de l'altre amb facilitat[8].

Tradicionalment, la protecció i la transmissió segura de dades era feina dels criptosistemes de clau secreta, fonamentalment restringits a àmbits militars i diplomàtics.

Al llarg de la història hi han hagut molts criptosistemes de clau secreta i gairebé tots han estat molt utilitzats tot i que podien ser descoberts amb facilitat, ja que la clau utilitzada s'ha de transmetre en algun moment entre l'emissor i el receptor, la qual cosa requereix un canal segur, cosa que a la pràctica és impossible. La seguretat del sistema depèn de la clau i són, generalment, més fàcils de criptoanalitzar que els de clau pública.

3.1. Avantatges i inconvenients

Entre els avantatges més importants de la criptografia de clau secreta destaca la seva eficiència, ja que són molt més ràpids que els criptosistemes de pública, i la curta longitud de les seves claus.

No obstant, entre els inconvenients més comuns dels criptosistemes de clau secreta destaquen els següents:

[8] Per aquesta raó són considerats també sistemes simètrics.

• Distribució de claus. Com hem mencionat anteriorment, dos usuaris han d'escollir una clau en secret i comunicar-se-la mútuament, pel qual hauran de desplaçar-se cap a un lloc comú per efectuar la comunicació d'aquesta, o bé enviar-la mitjançant un canal segur cosa que podria posar en perill la confidencialitat de la clau.

• Manca de firma digital. El destinatari no pot estar segur que qui li envia el missatge sigui realment el remitent, això passa ja que els sistemes de clau secreta no tenen la possibilitat de ser firmats digitalment. Realment, en els criptosistemes de clau secreta no és necessària la firma digital si es restringeix el coneixement de la clau als dos usuaris. El problema sorgeix quan un d'ells ha de convèncer a un tercer (suposem un jutge), que el missatge és autèntic, és a dir, ha estat vertaderament escrit i firmat per la persona que diu ser el seu autor.

3.2. Sistema de Cèsar

Un tipus de criptosistema de clau secreta és el sistema de César, el qual ha estat esmentat anteriorment.

Aquest criptosistema va ser utilitzar per Juli César i els seus generals i és un dels primers criptosistemes documentats històricament. És un sistema de substitució molt simple, en el que cada lletra de l'alfabet es correspon amb una altra que està k llocs més endavant. El número utilitzat és la clau del criptosistema. Aquest criptosistema es pot expressar per aritmètica modular amb la següent fórmula:

$$c = m + k \pmod{N}$$

On c és la lletra ja encriptada, m la lletra que es vol encriptar, k la clau (número utilitzat), i N el nombre de lletres que té l'alfabet utilitzat.

Actualment aquest sistema ha quedat desfasat, ja que té una gran facilitat per endevinar el missatge sense saber la clau privada, només s'han de provar les 27 possibles k per poder trobar el missatge original i comparat amb l'intent de desxifrar altres missatges, aquest seria molt ràpid.

3.3. Exemple

Codificarem el següent missatge: NEN RIC amb la clau $k = 6$

Primer hem de trobar el número que li pertoca a cada lletra segons l'alfabet:

A	B	C	D	E	F	G	H	I	J	K	L	M	N
00	01	02	03	04	05	06	07	08	09	10	11	12	13

Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	[ ]
14	15	16	17	18	19	20	21	22	23	24	25	26	27

N E N [ ] R I C

13 03 13 27 18 08 02

Ara hem de moure cada lletra 6 llocs cap a la dreta amb la fórmula anterior:

$$19 \equiv 13 + 6 \pmod{28}; \quad 19 = S$$

$$9 \equiv 3 + 6 \pmod{28}; \quad 9 = J$$

$$19 \equiv 13 + 6 \pmod{28}; \quad 19 = S$$

$$5 \equiv 27 + 6 \pmod{28}; \qquad 5 = F$$

$$24 \equiv 18 + 6 \pmod{28}; \qquad 24 = X$$

$$4 \equiv 8 + 6 \pmod{28}; \qquad 14 = Ñ$$

$$8 \equiv 2 + 6 \pmod{28}; \qquad 8 = I$$

Un cop passem els números obtinguts a lletres, el missatge que ens queda és: SJSFXÑI

4. CRIPTOSISTEMES DE CLAU PÚBLICA

A mitjans dels anys 70, la criptografia tradicional va experimentar una revolució profunda amb l'aparició dels criptosistemes de clau pública. El desenvolupament i la proliferació d'equips d'electrònica digital molt econòmics va alliberar els processos criptogràfics de les antigues limitacions de la computació mecànica i va permetre un ús general de les tècniques de protecció de dades. Alhora, aquesta va crear la necessitat de nous tipus de sistemes criptogràfics per a que resolguessin els inconvenients dels criptosistemes de clau secreta, fonamentalment, la distribució de les claus d'una manera segura i la possibilitat d'un equivalent digital a la firma d'un missatge.

El desenvolupament de les comunicacions està proporcionant un contacte fàcil i econòmic entre les persones i entitats que tendeixen a substituir els mecanismes tradicionals de correu escrit, a la vegada que introdueixen la seva utilització per a la transmissió de tot gènere de dades, també les que tenen un alt valor per ser confidencials. Resulta, per tant, natural pensar que aquest desenvolupament ha d'anar acompanyat de les mesures de seguretat necessàries per evitar les interferències il·legítimes de tercers no autoritzats.

Els criptosistemes de clau pública a diferència dels de clau privada, es basen en el fet que cada usuari posseeix un parell de claus, la primera de les quals es fa pública[9] mentre que la segona roman privada i només és coneguda pel seu creador.

La seguretat del sistema resideix en la dificultat computacional de descobrir la clau privada a partir de la pública.

[9] Això és possible ja que coneixent només la pública és impossible deduir la privada.

El naixement de la criptografia asimètrica va sorgir quan s'estava buscant una manera més pràctica d'intercanviar les claus simètriques.

4.1. Requisits d'un criptosistema de clau pública

Diffie i Hellman són considerats els iniciadors de la criptografia de clau pública gràcies al seu article al 1976. En aquest article exposen teòricament els requisits de qualsevol sistema d'aquest tipus. Els requisits són els següents:

- El càlcul de claus (públiques i privades) ha de ser computacionalment senzill, és a dir, donat per un algorisme de complexitat polinòmica.

- El procés de xifrat ha de ser computacionalment senzill.

- El procés de desxifrat, coneixent la clau secreta, ha de ser també computacionalment senzill.

- L'obtenció de la clau secreta, a partir de la pública, ha de ser un problema computacionalment impossible, és a dir, donat per un algorisme de complexitat computacional.

- L'obtenció del missatge original, coneixent el missatge xifrat i la clau pública ha de ser també computacionalment impossible.

Tot i que Diffie i Hellman van definir els principis de la criptografia de clau pública, com hem observat més amunt, van ser Ron Rivest, Adi Shamir i Leonard Adleman, investigadors del MIT (Massachusetts Institute of Technology), els primers que, al

1978, van trobar les funcions que satisfeien els requisits anomenats. Va néixer així, l'algorisme RSA (Rivest-Shamir-Adleman).

L'existència d'aquest algorisme no agradava molt a NSA (National Security Agency), la qual els va suggerir que no publiquessin les seves investigacions. No obstant, el seu algorisme va ser publicat a la revista Scientific American. Fruit d'aquests treballs va ser la creació de l'empresa RSA Data Security, Inc.

4.2. Protocol de xifrat amb clau pública

La criptografia de clau pública va néixer, bàsicament, per resoldre els inconvenients de la criptografia de clau secreta. En un criptosistema de clau pública cada usuari escull i domina, en realitat, dos claus íntimament relacionades: una és la clau pública que l'usuari posa a disposició de la resta d'usuaris del sistema i l'altra és la privada, únicament coneguda per ell. Si un usuari *A* vol enviar un missatge a un usuari *B*, mitjançant un protocol de clau pública un missatge xifrat, haurà de seguir els passos següents:

1.- *A* selecciona en el directori de claus públiques la clau pública corresponent a *B*.

2.- *A* xifra el seu missatge aplicant la clau pública de *B* i li envia.

Els passos de l'usuari *B* serien:

1.- *B* rep el missatge xifrat.

2.- *B* desxifra el missatge aplicant la seva clau privada només coneguda per ell.

El sistema basa la seva seguretat en la dificultat que representa per a qualsevol usuari diferent de *B*, que per tant no coneix la seva clau privada, desxifrar el missatge. Com més gran sigui aquesta dificultat, més segur es pot considerar el sistema i més difícil serà el treball del criptoanalista.

5. EL CRIPTOANÀLISI

El criptoanàlisi (del grec *kryptós*, "amagat" i *analýein*, "desfer"), és el terme contrari a la criptografia, si el propòsit d'aquesta és garantir la seguretat i el secret d'un missatge, el criptoanàlisi intenta per tots els mitjans descobrir aquest missatge secret, demostrant la seva vulnerabilitat, és a dir, tracta de desxifrar els criptogrames.

La paraula criptoanàlisi s'utilitza també per referir-se a qualsevol intent de trencar la seguretat d'altres tipus d'algorismes i protocols criptogràfics. No obstant, el criptoanàlisi acostuma a excloure atacs que no tinguin com a objectiu primari els punts dèbils de la criptografia utilitzada, com per exemple el robatori. Avui en dia aquests atacs s'estan fent més efectius que el criptoanàlisi tradicional.

Tot i que l'objectiu ha estat sempre el mateix, els mètodes i tècniques del criptoanàlisi han canviat dràsticament a través de la història de la criptografia, adaptant-se a una creixent complexitat criptogràfica que comprèn des dels mètodes de paper i llapis del passat fins a arribar als sistemes basats en computadores del present.

El criptoanàlisi ha evolucionat conjuntament amb la criptografia, i la competició entre tots dos pot observar-se perfectament al llarg de tota la història de la criptografia.

El primer mètode que es va utilitzar va ser sens dubte l'anomenat *força bruta*[10], és dir, anar provant totes les possibles claus fins trobar la correcta.

Hi han dos tipus de mètodes bàsics:

[10] En molts casos resulta un sistema inútil ja que pot ser que hi hagin infinites claus i això suposaria una pèrdua enorme de temps.

- Actius: El criptoanalista du a terme accions com fer-se passar per un transmissor autoritzat, tracta de substituir o modificar els missatges entre dos usuaris. El fet d'ordenar un altre cop la informació pot tenir efectes devastadors en un sistema. Els principals atacs actius són:

    - Atac a la meitat del camí: El criptoanalista és capaç de col·locar-se al mig de la comunicació entre els dos usuaris que s'intercanvien missatges, de manera que intercepta els missatges que A envia a B i els substitueix pels seus propis missatges, en altres paraules, l'atacant suplanta a l'usuari A.

    - Atac de la temporalització: Es utilitzat per un criptoanalista quan posseeix informació relativa al temps que tarda en efectuar-se el procés de desxifrat i coneix, a més, el nombre d'operacions que aquest requereix. Amb això podria arribar a obtenir informació, com per exemple sobre la mida de la clau privada.

- Passius: El criptoanalista tan sols intenta recuperar la clau i el missatge a partir del text xifrat. No participa, per tant, en la comunicació entre els usuaris del sistema. Els diferents atacs passius, ordenats pel grau de dificultat des del punt de vista d'un criptoanalista, són els següents:

    - Atac al text xifrat: L'adversari intenta deduir la clau de desxifrat o el text, a partir únicament del coneixement del text xifrat o criptograma. Si un criptosistema és vulnerable a aquest tipus d'atacs, es considera que és completament insegur.

    - Atac al text clar conegut: El criptoanalista posseeix una determinada quantitat de textos sense xifrar i els seus xifrats respectius.

- Atac al text clar escollit: L'adversari escull un text clar i n'obté el seu xifrat, de manera que la informació que pugui deduir d'aquest coneixement pot aplicar-se per tractar de desxifrar un nou criptograma o aconseguir-ne la clau.

- Atac al text xifrat escollit: L'adversari selecciona un text xifrat i n'obté el seu desxifrat corresponent. L'objectiu seria el d'obtenir un text clar a partir d'un text xifrat.

5.1. Criptoanàlisi clàssic

Tot i que l'expressió criptoanàlisi és relativament recent[11] , els mètodes per trencar codis i xifrats són molt més antics.

La primera explicació coneguda del criptoanàlisi es deu al savi àrab del segle IX, Yusuf Yaqub ibn Ishaq al-Sabbah Al-Kindi, al seu *manuscrit per desxifrar missatges criptogràfics*[12].

L'anàlisi de freqüències[13] és l'eina bàsica per trencar els xifrats clàssics. En totes les llengües conegudes, lletres puntuals de l'alfabet apareixen més freqüentment que altres; per exemple, a l'Espanyol, les vocals són més freqüents, ja que ocupen gairebé la meitat del text complert, sent la E i la A les que apareixen en més ocasions, mentre

que algunes consonants com són la F, Z, J, X, W i K apareixen en molt poques ocasions.

[11] La va establir William F.Friedman al 1920.
[12] Aquest tractat inclou una descripció del mètode d'anàlisi de freqüències.
[13] Veure annexos (taules).

L'anàlisi de freqüències, per tant, ens dirà el contingut original si el xifrat utilitzat no és capaç d'ocultar aquestes estadístiques. Per exemple en un xifrat simple, en el qual cada lletra és substituïda per una altra, la lletra més freqüent probablement seria una A o una E.

L'anàlisi de freqüències es basa tant en el coneixement lingüístic com en les estadístiques i com que els xifrats són cada cop més complicats, les matemàtiques s'han convertit en una eina bàsica i predominant al criptoanàlisi. Aquest canvi va ser molt evident durant la Segona Guerra Mundial, quan els esforços per trencar els codis van requerir nous nivells de sofisticació matemàtica.

5.2 Criptoanàlisi modern

La criptografia moderna s'ha tornat molt més impenetrable al criptoanalista que els mètodes del passat, i sembla que en l'actualitat porten avantatge sobre els mètodes del criptoanàlisi. Per aquest motiu alguns historiadors com David Kahn mencionen majors possibilitats per a la intercepció com la col·locació de dispositius gravadors, els atacs de canal lateral i la criptografia quàntica.

6. ESTEGANOGRAFIA

L'esteganografia[14] és la branca de la criptografia que estudia els processos que s'han de seguir per dur a terme l'ocultació de missatges, per evitar que es percebi l'existència d'aquests. L'objectiu de l'esteganografia és ocultar el missatge dintre d'un altre sense informació important, de manera que l'atacant ni tant sols s'assabenti de l'existència d'aquesta informació oculta.

És considerat com l'art i la ciència d'escriure missatges secrets de tal manera que ningú, tret de la persona que l'envia i de la persona que el rep, conegui la seva existència. Generalment un missatge d'aquest tipus sembla ser una altra cosa, com un article, una foto, etc, per tal que no hi hagi cap tipus de sospita.

6.1. Tècniques esteganogràfiques

Alguns exemples de tècniques d'esteganografia que han estat utilitzats al llarg de la historia són:

- Missatge ocults en taulells de cera a l'antiga Grècia: La gent escrivia missatges en un taulell de fusta i després el recobria amb cera per a que semblés que no havia estat utilitzada.

- Missatges secrets en paper, escrits amb tintes invisibles entre línies o en les parts en blanc dels missatges.

---

[14] Prové d'un tractat de Johannes Trithemius anomenat "Steganographia", del grec "escriptura secreta", aquest tractat parla sobre la criptografia i l'esteganografia i està disfressat com un llibre de màgia negra.

- Durant la Segona Guerra Mundial: Agents d'espionatge utilitzaven micro-punts per enviar informació, els punts eren extremadament petits comparats amb els d'una lletra d'una màquina d'escriure per la qual cosa, en un punt es podia incloure un missatge sencer.

Amb l'arribada dels ordinadors s'han ampliat i diversificat les tècniques esteganogràfiques. Una de les més comuns consisteix en ocultar un missatge dintre de continguts multimèdia, barrejant els bits del missatge original amb els bits de l'arxiu gràfic o de so. L'arxiu resultant serà una imatge o arxiu de so totalment funcional que, a primera vista, no aixeca cap sospita, però que amb el software adequat es possible extreure'n la informació oculta.

6.2. Diferència entre esteganografia i criptografia

A diferència de la criptografia en la qual s'aprecia que el missatge està xifrat i indueix a la sospita, aquí la informació està immersa en un vehicle, sigui un text, imatge o so, aparentment normals. D'aquesta manera, la informació que conté només podrà revelar-se aplicant el mètode adequat.

Una altra clara diferència és que la criptografia modifica les dades per a que no siguin llegibles mentre que l'esteganografia simplement les oculta entre altres dades.

L'esteganografia es pot utilitzar també per reforçar la seguretat del missatge: primer es xifra i després s'oculta.

7. MECANISMES DE SEGURETAT

Per proporcionar qualsevol tipus de seguretat en la criptografia, és necessari conèixer els dos mecanismes de seguretat més importants:

- Xifrat: El xifrat pot fer-se per mitjà de l'ús de criptosistemes simètrics o asimètrics. El mecanisme de xifrat suporta el servei de confidencialitat de les dades.

- Firma digital: La firma digital es pot definir com un conjunt de dades que s'afegeixen a una unitat de dades de manera que protegeixin a aquesta unitat davant de qualsevol tipus de falsificació, i a més, permetin al receptor comprovar l'origen i l'integritat de les dades. No disposen de firma digital tots els xifrats.

7.1. Seguretat de Shannon

Claude Elwood Shannon (30 d'Abril de 1960 – 24 de Febrer de 2001) va publicar per primer cop el seu article *A mathematical theory of comunication* al 1948. És considerat com el pare de la teoria de la informació i és un dels homes que més han evolucionat individualment el concepte de comunicació humana.

Va ser Shannon qui va escriure els requisits de seguretat de criptosistemes d'una forma explicita: la difusió i la confusió.

- El propòsit de la difusió consisteix en evitar la redundància del text original sobre el text xifrat, i augmentar el desordre. Per això, podem utilitzar la transposició, que evita els criptoanàlisis basats en les freqüències de les *n*-paraules. Una altra manera d'aconseguir-ho és fer que cada lletra del text xifrat

depengui d'un gran número de lletres del text base, amb el qual a més allargaríem la seva longitud.

- La confusió, en canvi, consisteix a fer que la relació entre la clau i el text xifrat sigui el més complexa possible.

Separadament, ni la confusió ni la difusió constitueixen unes bones tècniques de xifrat, no obstant, quan s'uneixen, poden donar lloc a criptosistemes molt segurs com el D.E.S i les seves variants, que s'utilitzen àmpliament a la xarxa.

Shannon va aportar també els conceptes de seguretat incondicional i seguretat computacional:

- Seguretat incondicional: Un sistema criptogràfic és incondicionalment segur si és irrompible, no importen els recursos ni el temps que posseeixi el criptoanalista.

- Seguretat computacional: Un sistema criptogràfic és computacionalment segur si és irrompible, suposant que el criptoanalista posseeix el temps i recursos limitats.

La seguretat incondicional pot arribar a ser inabastable: avui en dia s'han aconseguit crear sistemes de tal complexitat computacional que es suposa que els millors criptoanalistes trigarien anys en trencar-los, la qual cosa és suficient per mantenir la seguretat.

El sistema RSA compleix aquest requisits de complexitat computacional: avui dia un nombre de 150 xifres o més és gairebé computacionalment impossible de factoritzar, amb la qual cosa s'aconsegueix un grau de seguretat molt elevat.

7.2. Principi de Kerckhoffs

El principi de Kerckhoffs[15] diu que la seguretat d'un criptosistema es mesura suposant que el criptoanalista coneix tots dos processos de xifrat i desxifrat. A major quantitat de text xifrat major possibilitat de recuperar el text original. De fet, la seguretat és absoluta quan el missatge xifrat no proporciona cap informació sobre l'original.

El principi de Kerckhoffs es basa que la seguretat del text xifrat ha de residir només en el secret de la clau i no el mecanisme utilitzat per xifrar-lo, per tant aquí la simplicitat o complexitat que pugui tenir la clau és fonamental.

Fins aquí ha estat una petita introducció al que aprofundiré en aquest treball, l'algorisme RSA.

[15] Auguste Kerckhoffs, va ser un holandès autor de "*La criptografia Militar*", 1883.

8. RSA

8.1. Definició i història d'algorisme

L'RSA és un algorisme, paraula que prové del nom del matemàtic anomenat Muhammad ibn Musa al-Jwarizmi que va viure entre els segles VIII i IX. El seu treball va consistir en preservar i difondre el coneixement de l'antiga Grècia i de l'Índia. Els seus llibres eren de fàcil comprensió, ja que el seu principal assoliment no va ser el de crear nous teoremes o corrents de pensaments sinó el de simplificar les matemàtiques a fi que poguessin ser enteses per un major nombre de persones.

Així la paraula *algorismo*, que originàriament feia referència a les regles d'ús de l'aritmètica que utilitzen dígits àrabs, va evolucionar a la paraula llatina, *algobarismus*, que més tard, concretament en el segle XVIII, es transformaria en algorisme.

Un algorisme és un conjunt ordenat i finit d'operacions que permeten trobar la solució d'un problema. Els algorismes són l'objecte d'estudi de d'algorítmica i serveixen per executar una tasca i resoldre problemes matemàtics.

El primer algorisme escrit per a una computadora va sorgir al segle XIX, l'autora del qual va ser Ada Byron, en els escrits de la qual es detallava la màquina analítica[16]. És un sistema pel qual s'arriba a una o diverses solucions, tenint en compte que ha de ser definit, finit, i eficient. Per eficient s'entén que cada pas a seguir té un ordre; finit implica que té un nombre determinat de passos, és a dir, que té un fi; i per definit entenem que si es fa servir el mateix procés més d'una vegada s'arriba sempre al mateix resultat.

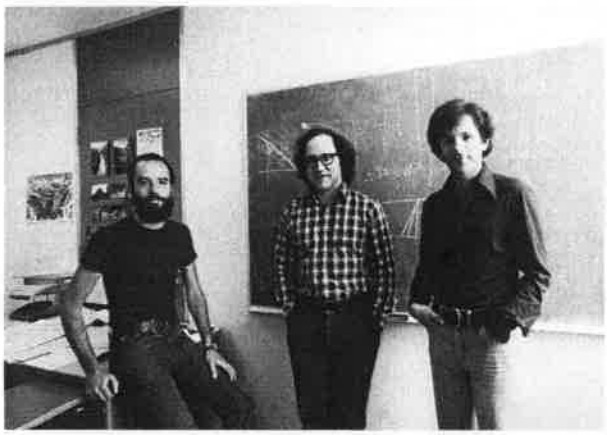
[16] La màquina analítica, és el diseny d'una computadora d'ús general realitzada pel professor britànic de matemàtiques Charles Babbage, que va representar un pas molt important en la història de la computació.

El terme algorisme no està exclusivament relacionat amb les matemàtiques, les ciències de la computació o les matemàtiques, sinó que a la vida quotidiana utilitzem multitud d'algorismes per resoldre diversos problemes, com per exemple l'ús d'una rentadora (es segueixen les instruccions).

8.2. Definició i història de l'RSA

L'algorisme de clau pública RSA va ser creat al 1978 per Ronald Rivest, Adi Shamir i Leonard Adelman, el nom RSA fa referència a les inicials dels cognoms de cadascun dels seus creadors.

Des de llavors, ha mantingut la seva seguretat i avui dia és el sistema criptogràfic asimètric més conegut i utilitzat que es coneix. El fet que s'hagi convertit en un dels sistemes més útils del moment és gracies a que té innumerables avantatges sobre els sistemes de clau secreta.



Ronald Rivest, Adi Shamir i Leonard Adelman

Els seus creadors es van basar en l'article de Diffie-Hellman[17] sobre els sistemes de clau pública, van crear l'algorisme i van fundar l'empresa RSA Data Security Inc., que és actualment una de les més prestigioses en l'entorn de la protecció de dades.

El sistema està basat en la dificultat del problema matemàtic de la factorització d'un nombre compost molt i molt gran i en les potències modulars, de fet, els missatges

[17] Diffie-Hellman és un protocol que permet l'intercanvi secret de claus entre dues parts que no han tingut un contacte previ, utilitzant un canal insegur i de manera anònima.

enviats utilitzant l'algorisme RSA es representen mitjançant números el funcionament dels quals es basa en el producte de dos nombres primers grans.

8.3. Funcionament de l'algorisme RSA

A continuació s'explica d'una manera entenedora el protocol desenvolupat per Rivest, Shamir i Adleman sobre l'RSA.

8.3.1 Generació de claus

1.- El primer que hem de fer per poder dur a terme aquest algorisme és trobar dos números primers molt grans[18] (que els nombres primers tinguin més xifres o menys augmentarà o disminuirà la dificultat d'un intrús alhora d'intentar desxifrar-lo). Aquests nombres els anomenarem p i q, i seran la clau del secret del sistema.

2.- A continuació, mitjançant els dos nombres primers que hem escollit (p i q) trobarem n, que és el resultat del producte entre p i q. Ara n forma part de la clau pública.

3.- Ara hem de trobar *phi_n*, que és primer amb n, per trobar-lo realitzarem la següent operació:

$$phi\_n = (p\text{-}1) \cdot (q\text{-}1)$$

El número que hem anomenat *phi_n* formarà part de la clau secreta.

4.- Procedim a escollir un nombre d, tal que el mínim comú divisor entre aquest nombre i *phi_n* sigui 1:

[18] La longitud dels números es mesura en bits.

$$\text{m.c.d}\ (d, phi\_n) = 1$$

El número *d* és l'altra part de la clau privada.

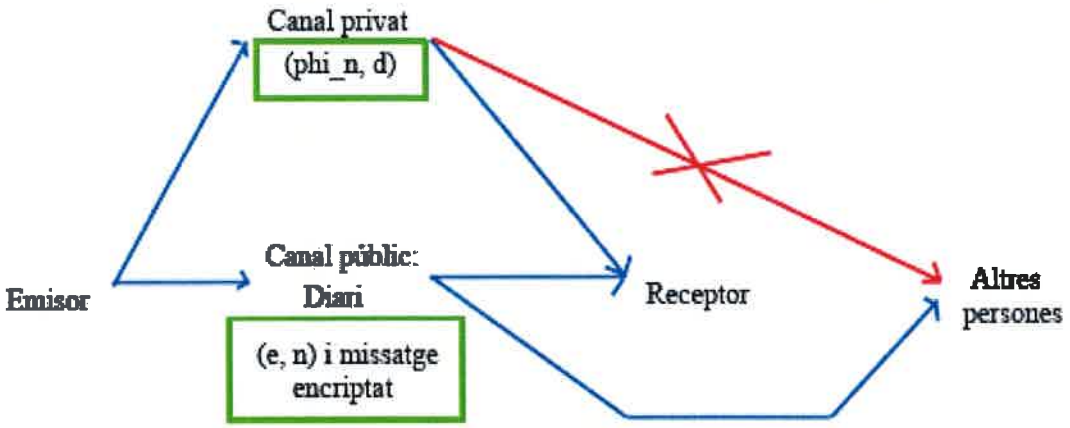
5.- Calculem *e*, de tal manera que: $1 \leq e \leq phi\_n$, mitjançant la fòrmula

$$e \cdot d \equiv 1\ (\text{mod}\ phi\_n)$$

Aquesta operació ens conduirà a una equació diofàntica ja que $e \cdot d \equiv 1$ (mod *phi_n* equival a $e \cdot d = phi\_n \cdot y + 1$. El número *e* és part de la clau pública.

6.- Fem pública la clau de xifrat (*e, n*).

8.3.2 Visualització de les claus



8.3.3 Xifrat de missatges

1.- Per xifrar un text, és necessari prèviament codificar-lo en un sistema numèric, aquest sistema numèric ha de ser conegut tant pels que xifren el missatge com

per als que l'han de desxifrar. Un exemple senzill és el sistema criptogràfic conegut com sistema de *Vigenère*[19].

A més hem de dividir el missatge en blocs M de mida j de tal manera que:

$$10^{j-1} < n < 10^{j}$$

2.- Un cop hem codificat el text en un sistema numèric el xifrem elevant el resultat de codificar el text en el sistema numèric a e, que forma part de la clau pública.

$$(\text{text xifrat en el sistema numèric})^{e}$$

8.3.4. Desxifrat de missatges

Quan el receptor rep el missatge xifrat, ha de realitzar un seguit d'operacions per poder-lo desxifrar, ara bé, hem de recordar que aquesta persona coneix tant les claus públiques (n, e) com les privades (n, d). Les operacions que ha de dur a terme són les següents:

$$\text{text xifrat } (m, \text{ codificat en un sistema numèric}) = x \bmod n$$

Per tant, hem de trobar m mitjançant la congruència anterior. Un cop tinguem m, la elevem a d (clau privada) i fem:

$$m^{d} \pmod{n} = z$$

Trobarem z, i aquest serà el xifrat codificat en el sistema numèric, que el receptor, que el coneix, podrà descodificar.

---

[19] Veure annexos (taules).

8.4. Exemple

8.4.1. Xifrat del missatge

Per facilitar l'explicació anterior utilitzarem un exemple on treballarem amb números petits.

El missatge que encriptarem com a exemple és LA FADA MÀGICA.

1.- Escollim els dos nombres primers $p = 3$ i $q = 5$

2.- Amb ells, trobem *n*:

$$n = p \cdot q = 3 \cdot 5 = 15$$

3.- Trobem *phi_n*:

$$phi\_n = (3-1) \cdot (5-1) = 8$$

4.- Escollim *d*, complint la condició m.c.d (8, *d*) = 1, per exemple 3. Això es pot fer prenent un nombre a l'atzar i calculant el màxim comú divisor amb *phi_n* . Si el resultat és 1, els nombres són primers entre ells i per tant, aquest nombre ens serveix; si aquest no és el resultat haurem d'escollir-ne un altre.

5.- Calculem *e*, seguint la condició $1 \leq e \leq 8$, mitjançant la fórmula

$$e \cdot d \equiv 1 \pmod{phi\_n}$$

$$e \cdot 3 \equiv 1 \pmod 8$$

$$e \cdot 3 = 8 \cdot x + 1$$

$$3e - 8x = 1$$

$3e - 8x = 1$ és una equació diofàntica[20]. Per resoldre aquesta equació la dividim tota entre el m.c.d (3, 8) = 1

$$\frac{3}{1}e - \frac{8}{1}x = 1$$

Trobem els nombres e i x que compleixin l'equació anterior:

- Creem una fracció amb a i b que sigui major que 1

$$\frac{8}{3}$$

- La descomposem fins que el numerador de la darrera sigui 1

$$\frac{8}{3} = 2 + \frac{2}{3} = 2 + \frac{1}{\frac{3}{2}} = 2 + \frac{1}{1 + \frac{1}{2}}$$

- Eliminem l'última fracció $\frac{1}{2}$

- Operem fins a obtenir una fracció impròpia

$$2 + \frac{1}{1} = \frac{3}{1}$$

Això vol dir que 3, 1, -3 o -1 poden ser els nombres e i x que necessitem per fer certa l'equació:

$$3e - 8x = 1$$

$$3\ (3) - 8\ (1) = 1$$

[20] Aquesta equació és del tipus $ax+by=c$, tot i que en la que realitzarem les lletres seran diferents: en comptes de la x serà una e i en comptes de la y una x.

$$9 - 8 = 1$$

Per tant, $e = 3$

6.- Fem pública la clau del xifrat $(e,n) = (3, 15)$.

7.- Escollim[21] j, la mida dels blocs en que xifrarem el missatge, complint la condició $10^{j-1} < n < 10^{j}$, li donarem a j el valor de 2:

$$10^{2-1} < 15 < 10^{2}$$

$$10 < 15 < 100$$

Com que la condició es compleix continuem el procés. El següent pas és codificar el missatge en un sistema numèric, utilitzarem el següent:

[ ]	A	B	C	D	E	F	G	H	I	J	K	L	M
00	01	02	03	04	05	06	07	08	09	10	11	12	13

N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26	27

L A F A D A M À G I C A

12 01 00 06 01 04 01 00 13 01 07 09 03 01

8.- Per codificar el text, hem d'elevar cada bloc a la clau pública $e = 3$.

[21] Pot ser que a l'hora d'escollir j i quan xifrem el missatge hi hagi algún bloc (acostuma a ser l'últim) que no compleixi la condició i que per exemple si j=3, només hi hagin dos nombres, en aquest cas els espais que quedin buits, s'omplirán amb 0.

$$12^3 = 1728$$

$$01^3 = 01$$

$$00^3 = 00$$

$$06^3 = 216$$

$$01^3 = 01$$

$$04^3 = 64$$

$$01^3 = 01$$

$$00^3 = 00$$

$$13^3 = 2197$$

$$01^3 = 01$$

$$07^3 = 343$$

$$09^3 = 729$$

$$03^3 = 27$$

$$01^3 = 01$$

El text xifrat, per tant és 1728 01 00 216 01 64 01 00 2197 01 343 729 27 01

8.4.2. Desxifrat del missatge

Un cop hem rebut el missatge xifrat, procedim a desxifrar-lo. Per desxifrar-lo hem de realitzar la següent operació:

$$\text{text} = m \bmod n$$

I elevem el resultat a d de la clau privada amb mòdul n, $m^d \bmod n = z$, on z és el missatge desxifrat en el sistema numèric:

$1728 = m \bmod 15 \rightarrow m = 3;\ 3^3 \bmod 15 = z \rightarrow z = 12;\ 12 = L$

$01 = m \bmod 15 \rightarrow m = 1;\ 1^3 \bmod 15 = z \rightarrow z = 1;\ 01 = A$

$00 = m \bmod 15 \rightarrow m\ \ 0;\ 0^3 \bmod 15 = z \rightarrow z = 00;\ 00 = [\ ]$

$216 = m \bmod 15 \rightarrow m = 6;\ 6^3 \bmod 15 = z \rightarrow z = 6;\ 06 = F$

$01 = m \bmod 15 \rightarrow m = 1;\ 1^3 \bmod 15 = z \rightarrow z = 1;\ 01 = A$

$64 = m \bmod 15 \rightarrow m = 4;\ 4^3 \bmod 15 = z \rightarrow z = 4;\ 4 = D$

$01 = m \bmod 15 \rightarrow m = 1;\ 1^3 \bmod 15 = z \rightarrow z = 1;\ 01 = A$

$00 = m \bmod 15 \rightarrow m = 0;\ 0^3 \bmod 15 = z \rightarrow z = 00;\ 00 = [\ ]$

$2197 = m \bmod 15 \rightarrow m = 7;\ 7^3 \bmod 15 = z \rightarrow z = 13;\ 13 = M$

$01 = m\ \bmod 15 \rightarrow m = 1;\ 1^3 \bmod 15 = z \rightarrow z = 1;\ 01 = A$

$343 = m \bmod 15 \rightarrow m = 13;\ 13^3 \bmod 15 = z \rightarrow z = 7;\ 07 = G$

$729 = m \bmod 15 \rightarrow m = 9;\ 9^3 \bmod 15 = z \rightarrow z = 9;\ 09 = I$

$27 = m \bmod 15 \rightarrow m = 12;\ 12^3 \bmod 15 = z \rightarrow z = 3;\ 03 = C$

$01 = m \bmod 15 \rightarrow m = 1;\ 1^3 \bmod 15 = z \rightarrow z = 1;\ 01 = A$

Ja hem desxifrat el missatge i el resultat ha estat: LA FADA MÀGICA, el procés d'encriptació i desencriptació ha estat el correcte.

8.5. Principals rècords de factorització

El rècord del mòdul RSA més gran factoritzat el va aconseguir, el 3 de desembre de 2003, un equip d'investigadors alemanys, dirigits per J.Franke, amb l'ajuda d'investigadors d'altres països. El nombre factoritzat és l'RSA-576, de 174 dígits, producte de dos primers de 87 dígits cadascun.

RSA-576 = 188198812920607963838697239461650439807163563379417382700763
356422988859715234665485319060606504743045317388011303396716199692321
2057340318795506569962213051687593076502570 59

Producte dels nombres:

P = 3980750864240649373971255005503864911990643623425267084063851895759463889572617685833 17

Q = 4727721461074353025362230719730482246329146953020971164598521711305207112563635903975 27

Al llarg del temps s'han anat superant els rècords de factorització[22], ja sigui pels avenços tecnològics o pels mètodes utilitzats.

8.6. Consells d'elecció dels nombres que formen l'algorisme

8.6.1. Elecció dels nombres primers *p* i *q*

En general, els nombres primers p i q, factors del mòdul n, han d'escollir-se de manera que factoritzar $n = p \cdot q$ sigui computacionalment molt difícil, ja que si un criptoanalista coneixés els valors de p i q mitjançant la factorització de n, també coneixeria el de *phi_n*, a partir del valor de *phi_n* el de e i a partir del de e, la clau privada d.

Per evitar que això passi, un dels requisits que haurem de seguir es que p i q han de tenir la mateixa longitud. Els nombres primers p i q han de ser nombres grans i tenir, aproximadament, la mateixa longitud, donat que si un d'ells és molt més petit que l'altre serà més fàcil d'obtenir, i si un d'ells està determinat es divideix el mòdul n entre el nombre primer ja calculat i n'obtindrem el segon nombre primer.

En l'actualitat es recomana que p i q tinguin com a mínim 512 bits, i per tant, que el producte n, tingui com a mínim 1024 bits (al voltant d'uns 309 dígits).

---

[22] Veure annexos (taules).

8.6.2. Elecció de l'exponent de xifrat *e*

En general, un cop que s'ha determinat un valor gran per al mòdul n, es recomana seleccionar un exponent de xifrat e petit, ja que qui ha de xifrar el missatge, m, ha de dur a terme l'operació $m^e \pmod n$, i d'aquesta manera facilitaríem la tasca del xifrat. Per aquesta raó es recomana que l'exponent de xifrat sigui 3 o bé 65537. Aquesta recomanació es basa en dos fets:

- Tant $e = 3$ com $e = 65537$ són nombres primers.

- L'expressió binària d'aquests nombres és molt senzilla, cosa que facilitarà molt la tasca de calcular els valors de $m^e \pmod n$. L'expressió binaria de $e = 3$ és 11 i només es requereixen dues multiplicacions modulars. L'expressió binaria de $e = 65537$ és 10000000000000001 i per calcular m^{65537} només fan falta 17 multiplicacions modulars.

D'altra banda dos usuaris poden tenir el mateix exponent de xifrat, però en aquest cas, han de tenir diferent el producte n, si no fos així, el coneixement que un usuari té del seu parell (e,d) li permetrà conèixer l'exponent de desxifrat de tots els altres, ja que tots tindrien la mateixa clau.

8.6.3. Elecció de l'exponent de desxifrat d

A l'hora d'escollir l'exponent de desxifrat d, hem de tenir en compte que ha de tenir una longitud aproximadament igual a la de n. Si

$$\text{longitud en bits } (d) \leq \frac{1}{4} \text{ longitud en bits } (n)$$

aleshores existeix un algorisme eficient per calcular d.

9. FIRMA DIGITAL

Un dels problemes que es planteja en la criptografia de clau pública és que qualsevol usuari amb accés a un directori públic de claus té també accés a les claus públiques d'altres usuaris, per la qual cosa pot enviar un missatge suplantant a una altra persona. Aquest problema es coneix com autentificació del remitent. No obstant, com ja s'havia mencionat, la criptografia de clau pública permet que tot missatge porti la seva pròpia firma digital. Aquesta propietat fa que el destinatari d'un missatge pugui verificar que el missatge rebut procedeix de qui diu ser el remitent. A més, la firma digital assegura que cap part del missatge s'ha modificat.

Per firmar un missatge, el remitent ha de dur a terme determinats càlculs amb el missatge que desitja enviar i la seva clau privada. D'aquesta manera, cada missatge porta la seva pròpia firma i es pot comprovar que el remitent és l'únic que ha pogut signar, donat que és l'únic que posseeix la seva clau privada.

9.1. Classificació de la firma digital

Les firmes digitals es classifiquen de diferents maneres:

- Implícites: Si es troben en el propi missatge.

- Explícites: Si són afegides com a una marca inseparable del missatge.

- Privades: Si permeten identificar al remitent només per algú que comparteixi un secret amb el remitent.

- Públiques: Si permeten identificar al remitent davant de qualsevol persona a partir de la informació públicament disponible.

- Revocables: Si el remitent pot, posteriorment, negar que la firma digital en qüestió no li pertany.

- Irrevocables: Si el receptor pot provar que el remitent va escriure el missatge.

9.2. Requisits d'una firma digital

Una bona firma digital ha de ser:

- Única

- Infalsificable, o el que és el mateix, computacionalment segura.

- Verificable pel receptor d'aquesta.

- Viable, és a dir, fàcil de crear.

9.3. Protocol per fer una firma digital

Fins ara només utilitzàvem les claus d'un usuari ja que era ell només qui transmetia el missatge i no rebia cap resposta, ara bé, amb la firma digital es necessiten les claus de tots dos usuaris ja que sinó no seria possible la creació d'aquesta.

El protocol de firma digital consta de dues parts. La primera és el procés de firma en sí i el du a terme el remitent del missatge, la segona part del procés és la verificació de la firma i és executada pel destinatari.

Suposem que les claus públiques de l'usuari B són (n_B, e_B) i la clau privada és d_B i que les claus de l'usuari A, amb qui es vol comunicar, siguin (n_A, e_A) les públiques i d_A la privada. Si l'usuari B vol enviar un missatge a l'usuari A juntament amb una firma digital, l'haurà d'elaborar segons el següent protocol:

1.- B calcula el valor de la seva firma r [23]:

$$r \equiv m^{d_B} (\mathrm{mod}\, n_B)$$

2.- Després xifra el valor anterior amb la clau pública de A:

$$s \equiv r^{e_A} (\mathrm{mod}\, n_A)$$

El missatge que envia B a A, està format per dos elements, c que correspon al missatge xifrat i s que correspon a la firma digital (c,s). Està clar que només B pot signar el missatge anterior, ja que és l'únic que coneix la clau privada d_B.

Un cop A rep la firma, la verifica per assegurar-se que és de B. Per fer-ho ha d'executar els següents passos:

1.- A recupera la firma de B calculant:

$$s^{d_A} (\mathrm{mod}\, n_A) \equiv r$$

2.- Posteriorment A comprova que el que ha rebut s, correspon amb el missatge m, que un cop desxifrada amb el mètode que els dos usuaris coneixent, correspondrà amb la firma del seu company:

$$r^{e_B} (\mathrm{mod}\, n_B) \equiv m$$

Si el resultat obtingut no coincideix amb m, el missatge serà rebutjat ja que això vol dir que ha estat manipulat al llarg de la transmissió o per que el remitent no és qui diu ser.

[23] r és la firma xifrada que B vol transmetre a A.

9.4. Exemple firma digital RSA

Per a dur a terme aquest exemple suposarem que la firma codificada que l'usuari B vol enviar juntament amb el missatge a A és 911. Suposarem també les següents claus:

$p_B = 307$

$q_B = 659$

$n_B = 307 \cdot 659 = 202313$

$phi\_n_B = 306 \cdot 658 = 201348$

$e_B = 65537$

$d_B = 138185$

$n_A = 199543$

$e_A = 3$

$d_A = 132427$

A continuació procedim a l'elaboració de la firma pel missatge ja codificat, el qual corresponia a la xifra 911:

1.- B calcula la seva firma:

$$r \equiv m^{d_B} \pmod{n_B}$$

$$r \equiv 911^{138185} \pmod{202313}$$

$$r \equiv 97598$$

2.- A continuació la determina amb la clau pública de l'usuari A:

$$s \equiv r^{e_A} (\mathrm{mod}\, n_A)$$

$$s \equiv 97598^3 (\mathrm{mod}\, 199543)$$

$$s \equiv 172625$$

Un cop feta la firma el parell de valors que ha d'enviar B a A quedarien de la següent manera (c, 172625)

9.4.1. Verificació de la firma

Per a que A s'asseguri de que és l'usuari B qui li envia el missatge, utilitza la seva clau privada $n_A = 132427$ i segueix el protocol:

1.- A recupera la firma de B calculant:

$$r \equiv s^{d_A} (\mathrm{mod}\, n_A)$$

$$r \equiv 172625^{132427} (\mathrm{mod}\, 199543)$$

$$r \equiv 97958$$

2.- Per acabar, A comprova si la firma xifrada amb la clau pública de B coincideix amb el missatge rebut m:

$$m \equiv r^{e_B} (\mathrm{mod}\, n_B)$$

$$m \equiv 97958^{65537} (202313)$$

$$m \equiv 911$$

Donat que A ha obtingut el missatge que va rebre, pot estar segur de que el missatge transmès va ser enviat per l'usuari B.

10. ATACS A L'ALGORISME RSA

10.1. Mitjançant la descomposició de factors

En cas que existís un atacant i aquest interceptés el misstage encriptat, no podría executar l'operació de desencriptació ja que no coneix *d* (recordem que per desencriptar el missatge hem d'elevar-lo a *d* i fer mòdul *n*). L'única forma que aquest la pot conèixer és calculant la inversa de *e* (que és pública) en mòdul *phi_n*. Tot i així per realitzar aquesta operació s'ha de conèixer *phi_n*, i *phi_n* recordem que és el resultat de calcular (*p*-1)*(*q*-1), per tant, hauria de conèixer *p* i *q*. No obstant, coneixem *n*, que és pública i el resultat del producte entre *p* i *q*, pel que el lector pot pensar que com *p* i *q* són primers només fa falta descomposar *n*, ja que n'obtindrem els dos primers pels que està formada. El problema està que per descomposar un nombre de 200 xifres es trigaria gairebé un milió d'anys fins i tot amb l'ordinador més potent.

De fet la clau de tot aquest algorisme està aquí, en el fet que calcular els factors d'un nombre molt gran és molt costós.

10.2. Criptoanàlisi Wiener-Boneh

Aquest atac basat en el criptoanàlisi Wiener-Boneh està basat en la debilitat que suposa utilitzar exponents de desxifrat petits.

Els exponents de xifrat més habituals per aquest algorisme acostumen a ser el 3 i el 65537, que fan que la seva curta expressió en binari faci que la potència per al xifrat sigui molt més ràpida. No obstant, l'elecció d'aquest nombre, *e*, determina la mida del seu invers, *d*, per la qual cosa aquest ha de ser analitzat amb l'objectiu d'evitar possibles debilitats (al ser *d*, l'invers de *e*, o viceversa, si tenim un exponent *e* gran,

tindrem un exponent *d* petit; pel contrari, si escollim *e* petit, *d* serà gran, però haurem de protegir-nos dels atacs quan *e* sigui petit).

11. DIFICULTATS MATEMÀTIQUES DE L'ALGORISME RSA

11.1. Congruències

L'encriptació i desencriptació de l'algorisme RSA està basada en operacions de congruències, per aquesta raó dedicaré aquest apartat al seu funcionament.

Les congruències van ser introduïdes formalment per K.F.Gauss a la seva obra "*Disquisitiones Arithmeticae*" per estudiar problemes aritmètics relacionats amb la divisibilitat, tot i que, posteriorment, s'han aplicat a molts dels problemes de la teoria dels números.

El terme congruència s'utilitza per designar que dos nombres enters, a i b, tenen el mateix residu quan els dividim per un número natural anomenat m. Per expressar això, utilitzem l'anotació: $a \equiv b(\mathrm{mod}\, m)$

Exemple senzill:

$$35 \bmod 4 = 3$$

En aquest exemple el 35 correspon a la a, el 4 a la m i el 3 a la b, de fet també es pot expressar com $35 \equiv 3(\mathrm{mod}\, 4)$

Per comprovar que el resultat és correcte realitzarem les divisions:

$$\begin{array}{r|l} 35 & 4 \\ 3 & 8 \end{array} \qquad \begin{array}{r|l} 3 & 4 \\ 3 & 0 \end{array}$$

Com es pot comprovar, la divisió de 35/4 i de 3/4 tenen el mateix residu.

A partir de l'anotació $a \equiv b(\text{mod}\, m)$, les següents expressions són equivalents:

- a és congruent amb b mòdul m
- El residu de a entre m és el residu de b entre m
- m divideix exactament a la diferencia de a i b.
- a es pot escriure com a $b + km$, on k sigui un enter.

11.1.1. Propietats de les congruències

- Reflexivitat: $a \equiv a$ (mod m)

  $35 \equiv 35$ (mod 4)

- Simetria: si $a \equiv b$ (mod m), llavors també $b \equiv a$ (mod m)

  $35 \equiv 3$ (mod 4) $\qquad$ $3 \equiv 35$ (mod 4)

- Transitivitat: si $a \equiv b$ (mod m) i $b \equiv c$ (mod m), llavors també $a \equiv c$ (mod m)

  $35 \equiv 3$ (mod 4) $\quad 3 \equiv 3$ (mod 4) $\qquad$ $35 \equiv 3$ (mod 4)

- Si a és coprimer amb m i $a \equiv b$ (mod m), llavors b també és coprimer amb m.

- Si $a \equiv b$ (mod m) i k és un enter, es compleix també que $a+k \equiv b+k$(mod m) i que $ka \equiv kb$(mod m).

  $35+2 \equiv 3+2$ (mod 4) = $37 \equiv 5$ (mod 4)

  $35 \cdot 2 \equiv 3 \cdot 2$ (mod 4) = $70 \equiv 6$ (mod 4)

- Si a més, k és coprimer amb m, llavors podem trobar un enter k^{-1}, tal que

$$kk^{-1} \equiv 1 (\text{mod } m)$$

a més, amb aquesta k podem parlar de la divisió

$$\frac{a}{k} \equiv \frac{b}{k} (\text{mod } m)$$

- Com a conseqüència de l'anterior, si tenim dues congruències amb mateix mòdul: $a \equiv b$ (mod m) i $c \equiv d$ (mod m), podem sumar-les, restar-les o multiplicar-les de forma que també es verifiquen les congruències $a+c \equiv b+d$ (mod m) i $ad \equiv bc$ (mod m).

11.2 Nombres binaris i potències binàries

Una dels principals dificultats de l'encriptació i desencriptació utilitzant l'algorisme RSA són les potències binàries.

Per exemple, imaginem que en el procés d'encriptació hem de fer la següent operació $5^{19} \text{ mod } 53^{24}$, el 5 seria el codi que hem d'encriptar, el 19 és la e, que pertany a la clau pública i el 53 és la n.

Doncs bé, elevar 5 a la 19 i treballar amb aquest número podria resultar impossible si fem servir una calculadora normal i li hauríem de dedicar molt de temps, és a dir, o disposem del temps i material necessari o intentem buscar una altra forma de trobar el resultat. Aquesta forma de la que parlem és l'anomenada potència binària.

[24] Tot i que aquest nombre continuaria sent petit, ja que per que l'algoritme sigui segur hem de treballar amb números de més de mil xifres.

Per dur a terme aquesta potència binària el primer que hem de fer és passar el 19, que és al número que està elevat el 5, al sistema binari.

11.2.1. Com passar decimals al sistema binari.

Els passos per passar un nombre al sistema binari són els següents:

1.- Primer hem de dividir el nombre entre 2 fins que el coeficient sigui menor que el residu:

$\frac{19}{2}$ = 9, el coeficient és 9 i el residu 1. Tornem a dividir el coeficient entre 2.

$\frac{9}{2}$ = 4, el coeficient és 4 i el residu 1. Tornem a dividir el coeficient entre 2.

$\frac{4}{2}$ = 2, el coeficient és 2 i el residu 0. Tornem a dividir el coeficient entre 2.

$\frac{2}{2}$ = 1, el coeficient és 1 i el residu 0. Tornem a dividir el coeficient entre 2.

$\frac{1}{2}$ = 0, el coeficient es 0 i el residu 1. No dividim més ja que el coeficient és menor que el residu.

2.- Un cop hem finalitzat les divisions, ens quedem amb els residus de cadascuna d'elles: 11001.

3.- Finalment ordenem els residus al revés, és a dir, 10011.

11.2.2. Com passar del sistema binari al decimal.

El nombre 19 en el sistema binari correspon amb el 10011.

L'operació inversa, és a dir, passar del nombre binari al nombre enter és molt més fàcil, només hem multiplicar cada xifra del nombre binari per 2 i elevar-la a la posició en que es troba començant per la dreta i, per acabar sumar el resultat:

$$10011 = 1\cdot2^4 + 0\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 = 16+0+0+2+1 = 19$$

11.3. Potències binàries

Un cop ja sabem com passar un nombre al sistema binari i ho hem realitzat, procedim al que s'anomena potència binària ja que ara no treballem amb un 5^{19} sinó que ho fem amb un 5^{10011}.

El primer que hem de fer és multiplicar individualment cada xifra del sistema binari per 2, ho elevem a la posició que ocupa i ho sumem, tal i com ho faríem per passar-ho a número:

$$5^{10011} = 5^{1\cdot2^4+0\cdot2^3+0\cdot2^2+1\cdot2^1+1\cdot2^0}$$

La primera xifra del 19 en sistema binari és un 1, per tant, hem multiplicat 1 per 2 i ho hem elevat a 4, ja que és la posició que ocupa començant a comptar des de la dreta i tenint amb compte el 0. Així successivament.

El resultat d'aquesta operació és el següent:

$$5^{2^4+2^1+2^0} = 5^{2^4+2+1}$$

A partir d'aquí operem:

Si tenim que $5^2 = 25$, tindrem que $5^2 \equiv 25 \bmod 53$. Per tant, si sabem el resultat d'aquesta congruència, com el que ens demanen és 5^{16}, el trobarem a partir d'aquesta i sense la necessitat de treballar amb números grans. Si sabem que 25 mod 53 = 25, sabrem que $5^4 = 25^2 = 625 \equiv 42 \bmod 53$, repetim el mateix procés dues vegades $5^8 = 42^2 = 1764 \equiv 15 \bmod 53$ i $5^{16} = 15^2 = 225 \equiv 13 \bmod 53$ i finalment tenim que $5^{16} \equiv 13 \bmod 53$. Hem realitzat aquesta operació sense la necessitat de treballar amb el número 152587890625 que és el resultat d'elevar 5 a la 16.

Per fer-ho més entenedor ho ordenaré tot en una columna:

$$5^2 \equiv 25 \bmod 53$$

$$5^4 = 25^2 = 625 \equiv 42 \bmod 53$$

$$5^8 = 42^2 = 1764 \equiv 15 \bmod 53$$

$$5^{16} = 15^2 = 225 \equiv 13 \bmod 53$$

Ara ja hem resolt la congruència $5^{2\hat{}4}$ mod 53. Doncs ara hem de resoldre les dues restants 5^2 mod 53 i 5 mod 53:

$$25 \bmod 53 = 25$$

$$5 \bmod 53 = 5$$

Ara que també hem resol les que ens faltaven multipliquem el seu resultat i tornem a fer la congruència:

$$13 \cdot 25 \cdot 5 \bmod 53 = 1625 \bmod 53 = 35$$

Per tant, 35 és el resultat de l'operació 5^{19} mod 53.

11.4. Nombres primers

El conjunt dels nombres primers és un subconjunt dels nombres naturals que engloba a tots els elements d'aquest conjunt majors que 1 que són divisibles únicament per sí mateixos i per la unitat.

Existeixen infinits nombres primers i Euclides en va fer una demostració al voltant del 300 a.C. Més endavant molts altres matemàtics han aportat la seva demostració mitjançant diversos mètodes.

La recerca de nombres primers és la segona dificultat que trobarem a l'algorisme RSA, ja que és un algorisme basat en el fet de que no existeix un algorisme suficientment eficient com per a factoritzar grans nombres que siguin producte de dos números primers.

De fet, quan van inventar l'RSA, els seus creadors van afirmar que farien falta 40.000 billons d'anys per desxifrar-lo, no obstant, això es va aconseguir al 1996. Les computadores més potents del món tardarien moltíssims anys en desxifrar una clau basada en la factorització, no obstant les noves tecnologies han fet que això sigui possible.

11.4.1 Recerca de nombres primers

11.4.1.1. Criba d'Eratòstenes

Tot i que no sigui un bon recurs perquè pot resultar un mecanisme llarg, podem esmentar la Criba d' Eratòstenes per trobar números primers. La Criba d' Eratòstenes és un algorisme que funciona de la següent manera:

1.- Es crea una taula amb números compresos entre el 2 i un número N que podem escollir nosaltres. Jo escolliré el 50:

2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21
22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41
42	43	44	45	46	47	48	49	50	

2.- Un cop tinguem feta la taula, hem d'eliminar els múltiples del 2, els del 3, els del 5, etc i així successivament, fins que arribem al 50, sense eliminar el primer que escollim ja que aquest el classifiquem com a primer:

Eliminem els múltiples del 2:

2	3		5		7		9		11
	13		15		17		19		21
	23		25		27		29		31
	33		35		37		39		41
	43		45		47		49		

Eliminem els múltiples del 3:

2	3		5		7				11
	13				17		19		
	23		25				29		31

			35		37				41
	43				47		49		

Eliminem els múltiples del 5:

2	3		5		7				11
	13				17		19		
	23						29		31
					37				41
	43				47		49		

Eliminem els múltiples del 7:

2	3		5		7				11
	13				17		19		
	23						29		31
					37				41
	43				47				

Hem escollit el primer número, el 2, i hem eliminat els seus múltiples, després amb el 3, amb el 5, etc. i aquest ha estat el resultat, és a dir, el 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43 i 47 són els nombres prims més propers a la unitat, concretament fins al 50 que hem escollit com a *N*.

Tot i que com ja he dit, per trobar números de moltes xifres és un sistema molt llarg, és efectiu.

11.4.1.2. Mètode de Fermat

Un altre mètode per esbrinar si un nombre és primer o no, és el mètode de Fermat, que es va descobrir al voltant del 1636.

Aquest mètode diu que si a és un enter i p un nombre primer es compleix la següent operació:

$$a^p \equiv a(\text{mod } p)$$

Per exemple, sabem que el número 7 és un nombre primer i utilitzem el 6 com a qualsevol enter, per tant:

$$6^7 \equiv 6(\text{mod } 7)$$

El resultat d'elevar 6 a la 7 és 279936 i si el dividim entre 7, el residu és efectivament 6, l'operació es correcta i podem afirmar que 7 és primer.

Realitzarem un altre exemple, però més complicat, ja que el realitzarem amb nombres grans. Primer busquem un nombre primer gran, com és el 5333 i ara utilitzem un enter qualsevol com el 14, realitzem l'operació:[25]

$$14^{5333} \equiv 14(\text{mod } 5333)$$

Com que l'operació és correcta, també podem afirmar que 5333 és primer.

Ara bé, en els exemples anteriors, els nombres primers han estat buscats, realitzarem el mateix amb un nombre qualsevol (sense saber si és o no és primer): escollim el 678

[25] Totes aquestes operacions amb nombres grans han estat realitzades amb Maple V Release 4, programa del qual parlaré més endevant, ja que amb una calculadora normal és impossible.

com a nombre p i el 56 com a nombre a, a continuació, realitzem l'operació amb l'ordinador i ho comprovem:

$$56^{678} \equiv 56 (\text{mod}\, 678)$$

és a dir,

$$56^{678} \mid 678$$

$$56 /$$

El resultat d'aquesta operació és 196 pel que comprovem que 678 no és primer ja que es divisible per 1, 2, 3 i 113.

Actualment existeixen els nombres primers més grans del moment[26].

[26] Veure annexos (taules).

12. MAPLE

Maple és un programa de computació de propòsit general, capaç de realitzar càlculs simbòlics, algebraics i d'àlgebra computacional. Va ser desenvolupat al 1981 pel Grup de Càlcul Simbòlic de la Universitat de Waterloo (Canadà). El seu nom prové de la frase MAthematical PLEasure (plaer matemàtic). Des d'aquell moment ha estat millorat.

El programa conté milers de procediments matemàtics i permet definir-ne de nous utilitzant el seu propi llenguatge de programació. A més, inclou eines suficientment flexibles per ajustar-se a totes les necessitats de càlculs: des de la resolució de sistemes d'equacions fins a problemes d'enginyeria.

El programa Maple té una sintaxis fàcil d'aprendre ja que les mateixes ordres recorden les operacions matemàtiques que s'executen. A més conté una gran quantitat d'explicacions i exemples.

Sigui quina sigui l'àrea científica en què s'està treballant, Maple és un entorn ideal que cobreix tots els aspectes necessaris.

12.1. Funcions que incorpora

Maple incorpora més de 3000 funcions entre les quals s'inclouen funcions de:

- Àlgebra: Factorització, expansió, combinació i simplificació d'expressions algèbriques i polinomis, seqüències, sèries i aritmètica simbòlica amb nombres reals i complexes o polinomis.

- Càlcul: Derivades, integrals i límits.

- Àlgebra lineal: Més de 100 funcions per construir, resoldre i programar en àlgebra lineal, construcció de matrius de Hankel, Hilbert, identitat, Toeplitz, Vandermonde, Bezout i la matriu Silvester de dos polinomis.

- Càlcul Vectorial: Derivades direccionals, gradients, matriu Hessiana, Laplacianes, rotacionals i divergències d'un camp vectorial, matrius Jacobianes i Wronskian, productes escalars, vectorials i externs de vectors i operadors diferencials.

- Altres funcions: Àlgebra d'operadors lineals, corbes algebraiques, funcions i estructures combinatòries, variables complexes, àlgebra diferencial, matemàtica financera, programació lineal, lògica, estadística, etc.

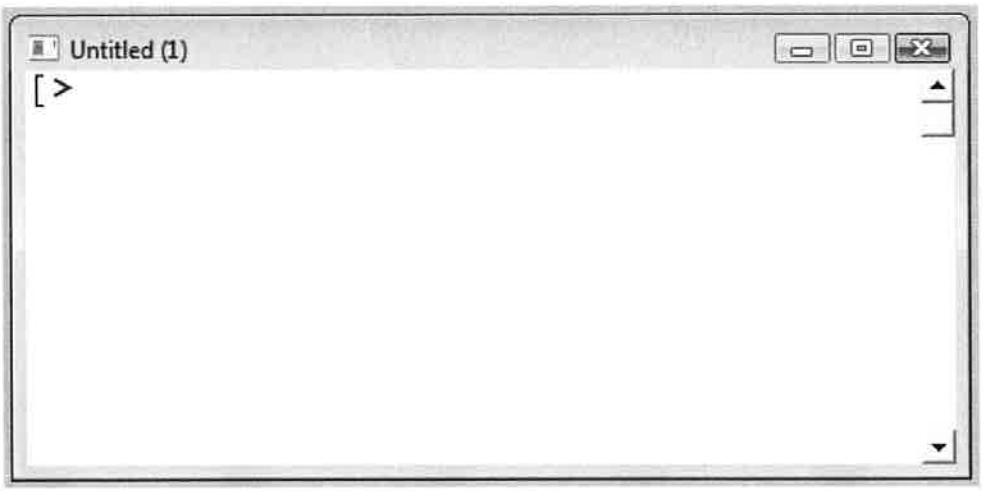
- Programació: Maple posseeix un llenguatge de programació avançat que inclou programació funcional, sobrecàrrega d'operadors, manipulació d'excepcions, eines de depuració, etc.

- Visualització: Inclou un ampli conjunt d'eines de visualització amb gràfics típics predefinits, gràfics 2D i 3D, animacions 2D i 3D, una amplia varietat de tipus de coordenades, gràfics implícits 2D i 3D, gràfics vectorials, contorns, gràfics complexes, gràfics ODSs i PDEs, rotació en temps real i objectes geomètrics.

12.2. Parts i descripció de Maple

12.2.1. Full de treball

Quan obrim Maple, apareix una finestra la qual anomenem fulla de treball (worksheet) En qualsevol fulla de treball hem de distingir entre les regions d'entrada, les regions de sortida i el text. En alguns casos, pot haver-hi una altra regió, la de gràfics. A mesura que s'executen comandaments a la fulla de treball, Maple va creant variables, emmagatzemant resultats intermedis, etc. L'estat del programa en un determinat moment del treball s'anomena estat intern del programa.



En Maple, l'usuari pot moure's per tota la fulla de treball situant el cursor en qualsevol línea, executant comandaments en qualsevol ordre, editant i tornant a executar sentencies anteriors, insertant de noves, etc.

És evident que tot això pot modificar l'estat intern del programa, i per tant, afectar el resultat d'alguna d'aquestes sentencies que depengui d'aquell estat[27].

[27] Una recomanació seria que un cop canviada una ordre, es tornés a clicar intro en totes les donades anteriorment o posteriorment a la modificada.

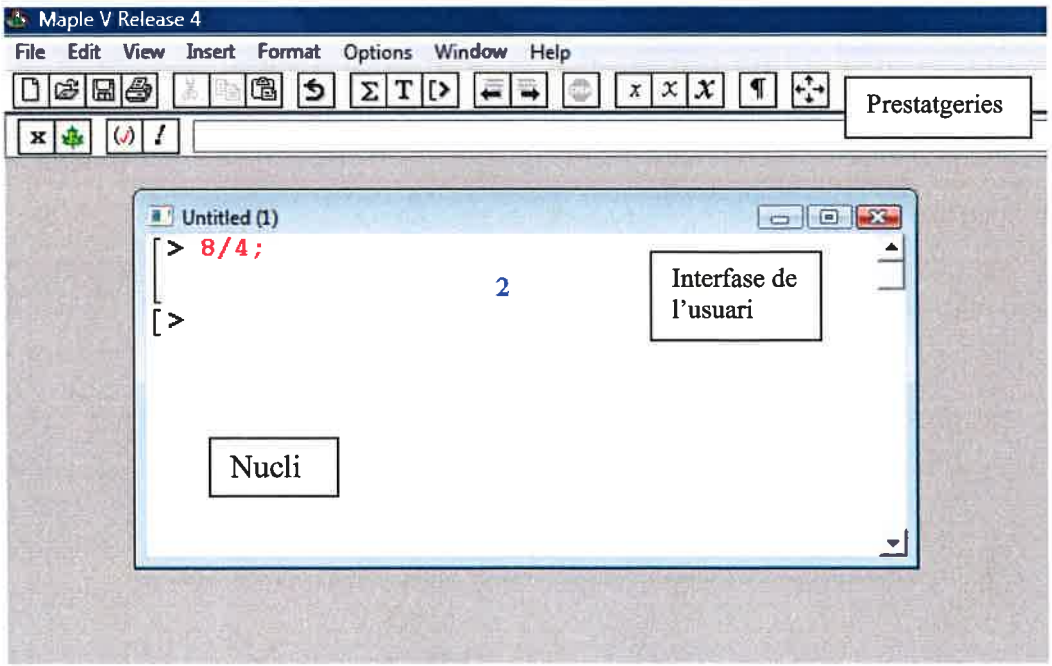
En Maple, no es poden utilitzar les famoses fletxes per recuperar ordres anteriors, no obstant, mitjançant la barra de desplaçament vertical es poden editar ordres que seran canviades en el moment que polsem l'intro.

Podem diferenciar en el programa tres parts principals:

- Nucli: Que és la part central del programa encarregada de realitzar les operacions matemàtiques fonamentals.

- Prestatgeries: Són un conjunt de funcions relacionades que resideixen en el disc. Maple disposa de més de 2000 comandaments. Només els més importants es carreguen quan el programa comença a executar-se. La major part dels comandaments estan agrupats en diverses prestatgeries temàtiques, que es troben al disc de l'ordinador. Un usuari mateix pot crear les seves pròpies prestatgeries.

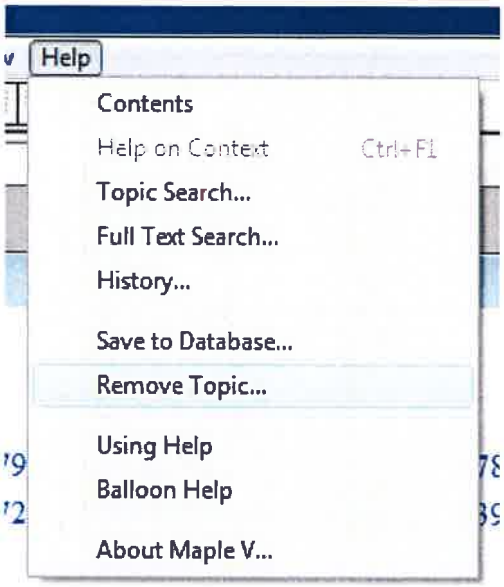
- Interfase de l'usuari: S'encarrega de totes les operacions d'entrada i de sortida, i en general, de la comunicació amb l'exterior.



12.2.2. Apartat Help

Maple, com moltes altres aplicacions de Windows, poseeix l'apartat d'ajuda (help). Aquest menú té dues opcions principals que són:

- Contents: Et mostra tot el que pots realitzar amb el programa d'una forma ordenada i amb exemples.



- Topic Search: Et permet buscar com és du a terme una operació mitjançant el seu nom en anglès, és a dir, busques una operació mitjançant el nom, per exemple, *prime* i el programa t'ofereix tota una sèrie de paraules que contenen el mot *prime* per a que tu puguis escollir la que més t'interessa. Un cop escollida, si cliquem a dalt, s'obrirà una pantalla amb un o un parell d'exemples d'aquesta funció.

12.2.3. Finestres de Maple

Maple té cinc tipus diferents de finestres:

- Finestra principal: O la també anomenada fulla de treball.

- Finestra d'ajuda: S'activa quan sol·licitem l'ajuda del Help.

- Finestra gràfica 2-D: S'obra una finestra nova cada cop que s'executa una ordre de dibuix 2-D. Aquestes finestres tenen els seus propis menús que permeten modificar de mode interactiu algunes característiques del dibuix: tipus de línea, d'eixos, etc.

- Finestra gràfica 3-D: S'obra també una finestra nova cada cop que s'executa una ordre de dibuix 3-D. També són finestres que tenen els seus propis menús, però diferents als de la finestra gràfica 2-D.

- Finestra d'animació: Aquest és l'últim tipus de finestra gràfica, diferent també a les dues anteriors. Disposen de comandaments similars als d'un equip de vídeo, per a poder veure de diverses formes l'animació produïda.

12.3. Instruccions

Per realitzar aquest treball he escollit el programa Maple versió Maple V Release 4 per fer la implementació amb l'algorisme RSA.
Com he dit anteriorment és un programa fàcil d'utilitzar, no obstant, has de saber el llenguatge i el codi que has d'aplicar pel que desitgis fer.

A l'hora d'iniciar una funció hem de començar escrivint *restart;* i cliquem intro. Això es necessari sobretot si es vol realitzar més d'una funció en una mateixa fulla ja que cada cop que vas implementant dades seguidament, aquestes valen per a tota la funció. Un cop que has escrit el codi del que vols calcular, has de finalitzar-lo amb *";"* i clicant intro, ja que això vol dir que has acabat d'escriure l'ordre i que vols que el programa realitzi el càlcul. Si no posem el punt i coma, quan cliquem la tecla intro ens sortirà un error i ens dirà que el codi està incomplert. Anem a visualitzar el següent exemple:

> restart;

> 5^2*7;

175

> ifactor(15);

(3) (5)

El primer cas és una operació senzilla i no cal posar cap més codi que el que volem fer, en canvi la segona si que necessita un codi específic: *ifactor*, ja que és el codi que necessitem per demanar-li que ens descompongui en factors prims el nombre que desitgem.

En aquest cas, posar *restart;* entre totes dues operacions no cal ja que cap de les dues no té res a veure amb l'altre. Anem a veure un altre exemple:

> restart;

> a:=6;

a := 6

> b:=3;

b := 3

> a*b;

18

> a/b;

2

> gcd(a,b);

3

En aquest segon exemple hem assignat a la lletra *a* i a la lletra *b*, mitjançant els signes :=, dos nombres diferents, d'aquesta manera si haguéssim de treballar amb nombres molt grans només caldria treballar amb les lletres. En l'exemple he multiplicat els nombres *(a*b)*, els he dividit *(a/b)* i he trobat el seu mínim comú divisor *(gcd)*. En aquest cas si volguéssim assignar altres nombres a les mateixes lletres en la mateixa fulla si que faria falta un *restart*, ja que un cop fet, tornem a començar:

> restart;

> a:=6;

a := 6

> b:=3;

b := 3

> a*b;

18

> a/b;

2

> gcd(a,b);

3

> restart;

> a*b;

a b

12.4. Nombres primers

Fent servir el Maple, trobar números primers molt grans no costa esforç ja que és el propi programa qui treballa per tu. Podem trobar números primers de diverses maneres:

- *Get Prime*: Mitjançant la funció *getprime* (traduït a l'anglès *"aconsegueix un primer"*) podem trobar qualsevol nombre primer amb el nombre de xifres que nosaltres li demanem.

  Per dur a terme aquesta funció hem d'escriure a la fulla de treball el següent codi:

```
> getprime:=proc(a) local pnumero,seed_alea,num_alea,Seed;
if (op(0,a)<>integer) then
ERROR(`Invalid parameter type. Must be an integer!`);
fi;
Seed:=readlib(randomize)():
seed_alea:=rand(10^a):
num_alea:=seed_alea():
pnumero:=nextprime(num_alea):
RETURN(pnumero):
end:
```

Posteriorment cliquem intro i realitzem la recerca, si per exemple volem un nombre primer de 250 xifres hem d'escriure a continuació el següent:

> getprime(250);

109646663843094052954060382219658125248993000007196511560888562048112639477400796460901449540169096891149384550838641197515332029902177677938919870743374986655733601674586306912936296890853913767574119569559800895707384893737287094316723503658240501

Amb aquest mètode podem saber amb seguretat que el nombre:
109646663843094052954060382219658125248993000007196511560888562048112639477400796460901449540169096891149384550838641197515332029902177677938919870743374986655733601674586306912936296890853913767574119569559800895707384893737287094316723503658240501, és un nombre primer. Això si, cada cop que demanem a Maple un nombre primer de 250 xifres el més probable és que ens doni cada cop un nombre primer de 250 xifres diferents.

Aquí si que faria falta el que hem explicat abans, és a dir, assignar-li una lletra o paraula a aquest nombre per a que resulti més còmode treballar amb ell:

> p:=getprime(250);

p :=
799038829277411466675252802327657994408062948384473704779639882991923961048062248106215489204593140781792491150264236496350

728294316053738979092409330119979017271114143791087729731302469895086037021059057321242550144981055203462904811106538797737209 9

> p;

799038829277411466675252802327657994408062948384473704779639882991923961048062248106215489204593140781792491150264236496350728294316053738979092409330119979017271114143791087729731302469895086037021059057321242550144981055203462904811106538797737209 9

Aquest cop li hem dit que volem assignar la lletra *p* a un nombre primer de 250 xifres, que com he afirmat abans, és un nombre diferent al que li hem demanat anteriorment. Per veure el funcionament d'aquesta assignació, li hem imposat la següent ordre al programa "*p*;" i efectivament, ens ha donat el nombre prim al que hem anomenat "*p*".

- *Next Prime: Nextprime* és una altra funció que traduïda de l'anglès i que vol dir "*prierm següent*". Aquesta funció consisteix en posar un número qualsevol i el programa et torna el número primer que es troba més a prop d'ell per damunt. Per dur a terme aquest procés no fa falta cap codi previ:

> restart;

>nextprime(9676597836895762387420897589246597823658926597862895689659816389265986598946769658396328983965983658365982365892368756734165467430260545054254652445921504271630927465827887024768465360164087632870);

9676597836895762387420897589246597823658926597862895689659816389265986598946769658396328983965983658365982365892368756734165467430260545054254652445921504271630927465827887024768465360164087632870

Ara hem trobat el següent nombre primer al número

67659783689576238742089758924659782365892659786289568965981638926598659894676965839632898396598365836598236589236875673416546743026054505425465244592150427163092746582788702476846536016408763287**0**

Per veure l'eficiència d'aquest mètode ho provarem amb els nombres primers més usuals. Visualitzem la Criba d' Eratòstenes:

2	3		5		7				11
	13				17		19		
	23						29		31
					37				41
	43				47				

Segons la taula el número primer després del 31 és el 37:

restart;

> nextprime(31);

37

Efectivament, el programa ha trobat el número primer després del 31, tot i que per trobar un número primer després d'un número, aquest no fa falta que sigui primer.

```
restart;
> nextprime(32);

                    37
```

Ara hem posat el 32 (que no és primer) i observem el mateix.

13. RSA EN MAPLE

Podem implementar l'algorisme en Maple de diverses maneres. Jo ho he fet de dues maneres que m'han semblat fàcils i entenedores i que a continuació explicaré. Pot ser no segueixen els mateixos passos que si ho féssim a mà però funciona.

13.1. Primer mètode

El primer que hem de fer es obrir Maple i escriure *restart*; i clicar intro. A continuació començarem introduint les ordres. Recordem que després del final de cada instrucció hem de clicar la tecla intro i que per fer la continuació d'una instrucció fem clic a la fletxa que trobem sota la tecla d'intro i sense deixar de clicar intro.
Primer la que codificarà l'alfabet en números:

```
>`crypt/alphabet` :=
`abcdefghijklmnopqrstuvwxyz`
.`ABCDEFGHIJKLMNOPQRSTUVWXYZ`
.```1234567890-=~!@#$£%^&*()_+`
.` ,./<>?;':"[]{}|`:
```

Això ens permetrà codificar les lletres en números i els números en lletres, no obstant no ho podem fer sense donar-li les instruccions:

```
> to_number := proc(st, string) local ll, nn, ss, ii;
 global `crypt/alphabet`; ll := length(st);
 if ll = 0 then RETURN(0) fi; nn := 1;
 for ii to ll do ss := SearchText(substring(st, ii .. ii),
```

```
`crypt/alphabet`);
 if not type(ss, numeric) or ss = 0 then
 ERROR(`the letter `.(substring(st, ii .. ii))
.` is not in the alphabet`)
 fi; nn := 100*nn + ss od;
 nn - 10^(2*ll)
 end:
```

Aquestes instruccions són per passar de lletres a números, per exemple, volem que ens codifiqui en números la frase: "*Estic escoltant la pluja*". Només cal que escrivim *to_number* i entre parèntesi allò que vulguem codificar:

```
> to_number(`Estic escoltant la pluja`);
```

3119200903800519031512200114208012018016122110 01

Ja tenim la frase codificada, però ara li afegirem les instruccions necessàries per a que ens torni la frase en lletres:

```
>from_number := proc(nn, integer)
  local ss, mm, ll, pp, ii, ans;
  global `crypt/alphabet`; mm := nn;
  ll := floor(1/2*trunc(evalf(log10(mm))))+1;
  ans := ``; for ii to ll do mm := mm/100;
  pp := 100*frac(mm);
  if pp > length(`crypt/alphabet`) then
  ERROR
  #(`there is no letter in the alphabet corresponding `
```

```
 #.`to the number `.(convert(pp, string))) fi;
(`there is NO meaningful text corresponding to the number you have inputted`)
fi;
ss := substring(`crypt/alphabet`, pp..pp);
ans := cat(ss, ans); mm := trunc(mm)
od;
ans
end:
```

Un cop establim les instruccions la tornem a passar a lletres:

```
> from_number(311920090380051903151220011420801201801612211001);
```

Estic escoltant la pluja

Ja tornem a tenir el missatge inicial. Amb el codi que hem escrit per passar de nombres a lletres i de lletres a nombres també s'inclouen signes de puntuació (,.!/?, etc) no obstant els accents no s'inclouen ja que està programat per ser traduït en anglès:

```
> to_number(`Escolto la pluja!!`);
```

311903151220158012018016122110016767

```
> to_number(`Escolto música`);
```

Error, (in to_number) the letter ú is not in the alphabet

Un cop hem introduït les ordres anteriors ens centrem en els passos de l'algorisme que recordarem a mesura que els portem a terme amb Maple.

Primer havíem d'escollir dos nombres primers als que anomenarem p i q. En un dels exemples que he dut a terme a mà eren dos nombres petits, no obstant, ara Maple pot treballar amb nombres grans.

```
> p:=nextprime(567828949975925269465969659995659484747344646 44);
```

p := 567828949975925269465969659995659484747344646 99

```
> q:=nextprime(752965876285682658282844856255474943843737735744 47);
```

q := 752965876285682658282844856255474943843737735744 59

Utilitzem la funció *nextprime*, explicada anteriorment per a que ens trobi el següent nombre primer a un aleatori i considerablement gran que hem introduït. Un cop tenim p i q, trobem n que és el seu producte, *phi_n* i *e*:

```
> n:=p*q;
```

n := 427555822899001633313058761952096208562220728374102746600 7483
76488618819159285087033742518352284 1

```
> phi_n:=(p - 1)*(q - 1);
```

phi_n := 427555822899001633313058761952096208562220728366567409548

39179065309571090013038100045766754836 84

> e:=nextprime(75859825925924926296969);

e := 7585982592592492629703 3

Per comprovar que anem per bon camí, el m.c.d entre *phi_n* i *e* hauria de ser 1:

> igcd(e, phi_n);

1

És correcte. El següent pas és codificar el missatge que vulguem xifrar, per exemple *Quedem al costat del port a les 10 del mati*[28] i al que anomenarem *numerical form of the text*:

> numerical_form_of_text:=to_number(`Quedem al costat del port a les 10 del mati`);

numerical_form_of_text := 432105040513800112800315192001208004051 2

80161518208001801205198054638004051280130120 09

Ja tenim el nostre missatge codificat parcialment ja que per poder passar-li a un receptor hem de fer una darrera operació, elevar-lo a *e* amb mòdul *n*. L'anomenarem *numerical form of the chiper text*:

>numerical_form_of_cipher_text:=numerical_form_of_text&^e mod n;

[28] No li posem accent a la *i*, ja que, com hem vist anteriorment, aquest sistema per passar de lletres a números no reconeix els accents.

numerical_form_of_cipher_text := 92594841339186757832466065654795
603334638209121040196951417528326057209029189237006231665024875
5

Aquest missatge és el missatge ja xifrat que nosaltres volem enviar:
92594841339186757832466065654795760333463820912104019695141752832605720902918923700623166502487.

Un cop el receptor hagi rebut el missatge i posseeixi les claus. Per poder desxifrar el missatge amb Maple, es fa de la mateixa forma que com si ho féssim a mà: $x^d \mod n = z$, no obstant Maple treballa com si el receptor no conegués d i el trobarem mitjançant el mètode de l'algorisme estès d'Euclides[29]:

```
>igcdex(e, phi_n, x, y);
```

1

```
>d:=x mod phi_n;
```

d := 42574893333247988439423343158596154075316884287164940062483085
31486916830882004049206391715393258 9

Un cop tenim *d*, podem desxifrar correctament el text:

```
>numerical_form_of_decoded_text:=numerical_form_of_cipher_text&^d mod n;
```

[29] L'algoritme d'Euclides ens permetrà trobar el m.c.d entre els números que vulguem.

numerical_form_of_decoded_text := 432105040513800112800315192001201

8004051280161518208001801205198054638004051280130120091

Ha de coincidir amb el text que l'emissor ha xifrat prèviament en el sistema numèric.

Ja tenim el text desxifrat però en sistema numèric. Per desxifrar-lo només tenim que realitzar el procediment de passar-ho amb números del qual hem fet un parell d'exemples anteriorment:

>original_text:=from_number(numerical_form_of_decoded_text);

original_text := Quedem al costat del port a les 10 del mati

Observem com les operacions i els passos han estat els correctes i el receptor ha rebut satisfactòriament el missatge encriptat.

Aquest mètode té un inconvenient i és que per alguna raó que desconec i que m'ha estat impossible descobrir, només treballa amb els números p i q de l'exemple. Per aquesta raó, realitzaré una altra implementació en la que posem el número que posem funciona, no obstant, aquests han de ser mínimament grans.

13.2. Segon mètode

Per realitzar aquesta implementació, el primer que hem de fer es crear un fitxer amb el programa *Bloc de Notes* i guardar-lo preferiblement en el directori *c*, tot i que jo utilitzaré un fitxer procedent del directori *d* i al que anomenaré prova.txt.

El primer que farem, com en qualsevol arxiu de Maple on vulguem treballar, serà reiniciar-lo:

```
> restart:
```

A continuació hem d'escriure-hi totes les instruccions per a que realitzi correctament el procés de xifrat i desxifrat. Comencem amb les instruccions per trobar un número primer aleatori amb el número de xifres que nosaltres li demanem, per això utilitzarem el mètode *get prime*, explicat anteriorment. Aquesta funció la farem servir per trobar *p* i *q*.

```
> getprime:=proc(a) local pnumero,seed_alea,num_alea,Seed;
 if (op(0,a)<>integer) then
 ERROR(`Invalid parameter type. Must be an integer!`);
 fi;
 Seed:=readlib(randomize)():
 seed_alea:=rand(10^a):
 num_alea:=seed_alea():
 pnumero:=nextprime(num_alea):
 RETURN(pnumero):
 end:
```

A continuació es defineixen els algorismes per obtenir els exponents d'encriptació i desencriptació *e* i *d*.

La primera de les funcions, *decryptkey* accepta un número enter que ha de ser *phi_n* i torna *d*. Com que per calcular *e*, necessitem conèixer *d*, la funció *encryptkey* requereix tant *phi_n* com *d*.

```
> decryptkey:=proc(f) local mcd,r,d,semilla;
 if (op(0,f)<>integer) then
 ERROR(`Invalid parameter type. Must be an integer!`);
 fi;
 mcd:=0:
 r:=0:
 while (mcd<>1) do
 semilla:=rand(f):
 d:=semilla();
 mcd:=gcd(d,f);
 od:
 RETURN(d);
 end:

> encryptkey:=proc(d,f) local aux,enc,e;
  if (op(0,d)<>integer) or (op(0,f)<>integer) then
 ERROR(`Invalid parameter type. Must be an integer!`);
 fi;
 enc:=msolve(aux*d=1,f):
 e:=rhs(enc[1]);
 RETURN(e);
 end:
```

La potència necessària per al càlcul del text xifrat i per al seu posterior desxiframent requereix un algorisme que accepti números grans tant en la base i exponent com per al mòdul. Per aquesta raó utilitzarem l'algorisme de potència ràpida *fastexp*.

```
> fastexp := proc (a,z,n) local a1,z1,x;
 if (op(0,a)<>integer) or (op(0,z)<>integer)
```

```
or (op(0,n)<>integer) then
ERROR(`Invalid parameter type. Must be an integer!`);
fi;
# return x=a^z mod n
a1:=a;z1:=z;
x:=1;
while (z1<>0) do
while ((z1 mod 2)=0) do
z1:=iquo(z1,2);
a1:=(a1*a1) mod n;
od;
z1:=z1-1;
x:=(x*a1) mod n
od;
RETURN(x);
end:
```

Quan Maple llegeixi el que hi ha escrit en l'arxiu que nosaltres demanem, ho convertirà tot en un número enter. Amb la intenció que aquest número no tingui una longitud superior a la del producte *n,* fem servir la funció *file2declist.*

```
> file2declist :=proc(archivo,n)
local fd, numero, cont, EOF,i,dimension:
global LISTA;
numero := 0:
fd := fopen(archivo,READ):
EOF := filepos(fd,infinity):
filepos(fd,0):
```

```
i:=1;numero[i]:=0;
for cont from 1 to EOF do
numero[i] := numero[i] * 256 + readbytes(fd,1):
if length(numero[i])>=(n-1) then
i:=i+1;
numero[i]:=0;
fi;
od:
fclose(fd):
LISTA:=numero;
dimension:=i;
RETURN(dimension):
end:
```

La funció *dec2word* és la inversa de l'anterior, accepta un enter i torna la cadena de caràcters equivalent.

```
> dec2word:=proc(l) local a,r,s,t,num,p,q,i;
r:=0;
num:=l;
while (num<>0) do
r:=r+1;
s[r]:=(num mod 256);
num:=iquo(num,256);
od;
for a from 1 to r do
t[r-a+1]:=s[a];
od;
```

```
p:=[seq(t[i],i=1..r)];
q:=convert(p,bytes);
RETURN(q);
end:
```

Ara introduïm les funcions *encrypt* i *decrypt* que eleven el número obtingut anteriorment a l'exponent respectiu (*e* o *d*, segons si xifren o desxifren) mòdul *n*.

```
> encrypt:=proc(l,e,n) local a,enc;
 for a from 1 to depth do
 enc:=power(l,e)mod n;
 od;
 RETURN(enc);
 end:

> decrypt:=proc(l,d,n) local a,dec;
 for a from 1 to depth do
 dec:=power(l,d) mod n;
 od;
 RETURN(dec);
 end:
```

A partir d'aquí ja comencem a introduir dades de l'algorisme RSA. Primer, mitjançant la funció *get prime* crearem *p* i *q* de 100 i 103 xifres respectivament:

```
> p:=getprime(100);
 q:=getprime(103);
```

p := 685399165074561164289452239836334598045897816622512801966919 4
1736060704701170237501900578854608101 11

q := 429502663387213216233016066980331185826759509508366310251317 6
8552606701949142874696984336040940720377 27

Ara obtenim el producte *n* i *phi_n*:

```
> n:=p*q;
 ffi:=(p-1)*(q-1);
```

n := 294380766882896228730644965791520566126501425125571873730873 8
0041298048216435777222238253688456348878781794146898275265789 8
3852936309176501691142776075149184460239759289460731964601440 6
753217541975057697

ffi := 294380766882896228730644965791520566126501425125571873730873 087
3800412980482164357772222382536884563488744799135213753974918 1
6783337161416560768857354510863261311756348158618341935241520 9
581295555624422098 60

Obtenim els exponents d'encriptació i desencriptació mitjançant la funció definida anteriorment.

```
> d:=decryptkey(ffi);
  e:=encryptkey(d,ffi);
```

d := 290945391145270317958764550421913645502447131489904989548291O
819359861877118238603884778587863234300072512634213431688145 98
988409613483934517459561252491978608615197057785077688104637 06
417753718718056099

e := 130823445546840402337581655420368005970604722665804911717 6330
379817774742579622053754269787117849254410986031180348970159 58
418067625243931700446364937084072947833789901803892450437078 22
085036976132253399

Realitzem un parell d'operacions per assegurar-nos de que tot és correcte:

> is(d>max(p,q)+1);
is(d<n-1);

true

true

> is(e>log[2](n));

true

Com que ha superat les proves recomanades per Rivest, Shamir i Adleman continuem. Ara treballem amb el fitxer creat al directori, que serà el missatge que vulguem transmetre. Primerament l'obrim:

> fichero:=`d:\prova.TXT`;

fichero := d:prova.TXT

Treiem la informació del fitxer i la convertim en una llista de números enters, la longitud de cada enter serà igual a la de *n* menys una unitat.

```
> depth:=file2declist(fichero,length(n)-1);
```

depth := 1

Comprovem que la llista existeix i té els nivells que indica *depth:*

```
> seq(LISTA[i],i=1..depth);
```

[527151239021539250337434855487545940402228442774603114553263093359930476]

Encriptem el contingut de *LISTA:*

```
> for i from 1 to depth do
 DataEncrypt[i]:=encrypt(LISTA[i][1],e,n);
od;
```

DataEncrypt[1] := 159526840572293491057875260430985996142457029072
208241850983831337948446015509925907962682097733164449406286l6
001655154702116426954134809328961282202077680324860183279108291
300408690706282019823140227866 8

Aquest seria el missatge ocult que hauríem d'enviar en cas de que vulguem transmetre alguna cosa a algú. Procedim a desxifrar-lo aplicant la funció inversa:

```
> for i from 1 to depth do
 DataDecrypt[i]:=decrypt(DataEncrypt[i],d,n);
 od;
```

DataDecrypt[1] := 527151239021539250337434855487545940402228442774
60311455326309335993047 6

Apliquem *dec2word* amb la intenció de que aparegui un text llegible:

```
> for i from 1 to depth do
 texto[i]:=dec2word(DataDecrypt[i]);
 od:
```

```
> cat(seq(texto[i],i=1..depth));
```

La Miriam m'ha demanat un full

Sembla que sí, per comprovar que és el text utilitzem una funció que llegeix el contingut del fitxer utilitzat.

```
> readbytes(fichero, infinity, TEXT);
```

La Miriam m'ha demanat un full

Coincideix, perfecte. Ja tenim el text desxifrat.

14. SITUACIÓ DE TRANSMISSIÓ DE MISSTAGES CODIFICATS AMB MAPLE

Imaginem una situació en la que dues amigues, l'Anna i la Laia es volen enviar un missatge encriptat utilitzant el Maple.

El primer que faran serà crear les claus. Ho poden fer a mà, però com es transmetran el missatge amb el Maple, serà més fàcil i ràpid crear-la amb el programa.

Per fer-ho, donen les instruccions necessàries al programa i procedeixen a escollir les claus:

>p:=getprime(6);

p := 254747

> q:=getprime(7);

q := 2254757

> n:=p*q;

n := 574392581479

> ffi:=(p-1)*(q-1);

ffi := 574390071976

> d:=decryptkey(ffi);

d := 375212704269

> e:=encryptkey(d,ffi);

e := 471799971477

La clau *p := 254747* i *q := 2254757* només serviran per poder obtenir la *n := 574392581479, phi_n := 574390071976, d := 375212704269 i e := 471799971477,* que constitueixen les quatre claus importants i que coneixeran totes dues.

No obstant la *n* i la *e*, poden ser conegudes per altres persones, o bé que una d'elles la rebi per un canal públic com per exemple un diari, ja que la coneixença d'aquestes claus per terceres persones no bastarà per a resoldre el missatge. *Phi_n* i *d* en canvi, només poden ser conegudes per elles.

Un cop cadascuna posseeix les quatre claus poden passar a l'encriptació del missatge. Suposem que aquestes noies es troben separades i l'Anna li envia el missatge per *e-mail* a la Laia.

Els passos que durà a terme l'Anna per codificar el missatge amb el Maple seran els següents:

- Primer introduirà a Maple l'adreça d'on es troba el fitxer que vol amagar amb la criptografia.

- Un cop Maple li ha tornat el missatge codificat:

369347450695605303254032712217530004545901577524142706462895631445770622362160757281492155815332942514787631950608334152355323401023646120256715704901593828199093345530758710378 6

El copiarà a l'*e-mail* que li vol enviar i li transmetrà les claus si es que encara la Laia no les coneix.

Quan la Laia rebi el missatge, introduirà les claus i el missatge codificat al Maple, el qual després de donar-li les instruccions adequades li tornarà el missatge descodificat:

`La clau del problema la trobarem a l'estable`

D'aquesta manera l'Anna i la Laia s'hauran transmet un missatge encriptat mitjançant un servei públic (els *e-mails*) i l'hauran descodificat amb facilitat utilitzant el Maple.

15. CURIOSITATS

15.1. "El Código Da Vinci"

"El Código Da Vinci" [30] és un llibre actual basat en el cas d'un assassinat que és l'inici de la recerca d'un tresor, les limitacions del qual són totalment criptogràfiques.

Entre elles trobem anagrames, que és una paraula o frase que resulta de la transposició de les lletres de la mateixa paraula o frase. En l'actualitat és un típic passatemps que apareix a les pàgines d'un diari o revista però de fet, consta d'una llarga història de simbolisme sagrat. Els ensenyaments místics de la Càbala es basen fonamentalment en anagrames en els quals mitjançant l'alteració de l'ordre de paraules d'origen hebreu s'obtenien nous significats. Els reis francesos del Renaixement estaven tan convençuts de que els anagrames tenien propietats màgiques que contaven amb anagramistes reals que els ajudaven a prendre les decisions més encertades mitjançant l'anàlisi de les paraules dels documents importants. Els romans donaven a l'estudi dels anagrames la categoria *ars magna* (art major).

L'exemple que ens ofereix el llibre és el següent:

¡Diavole in Dracon! ¡Leonardo da Vinci!

¡Límala, asno! ¡La *Mona Lisa!*

Al llibre també apareix un instrument anomenat *criptex* i creat per Leonardo Da Vinci que no és una manera d'encriptar un missatge, sinó una manera d'amagar-lo i així evitar que l'atacant pugui assabentar-se del seu contingut.

[30] BROWN, Dan. *El Código Da Vinci,* Barcelona, Ediciones Urano, S.A.

Un *criptex* és un recipient portàtil que pot contenir cartes, mapes, diagrames o qualsevol tipus de document. Un cop la informació queda segellada en el *criptex* només la persona que coneix la contrasenya pot accedir a ella.

El *criptex* que s'esmenta al llibre està compost de cinc discs que giren canviant de lletra cada cop. Per aconseguir obrir-lo, s'ha d'escriure la paraula correcta, en aquest cas de cinc lletres. Un cop el cilindre s'obra, es possible accedir a un compartiment interior que pot contenir un paper enrotllat amb la informació que s'ha volgut mantenir en secret.

Les possibilitats d'obrir el cilindre sense conèixer la clau són mínimes ja que encertar la paraula seria molt costós, per això la millor manera és trencar-lo. No obstant, el *criptex* ja està dissenyat per evitar aquesta situació: tota informació que es transmet dintre del cilindre ha d'estar escrita en un papir i abans d'introduir el papir dintre del cilindre, aquest ha d'envoltar un tub de cristall molt delicat que conté vinagre. El vinagre es capaç de dissoldre el papir, per aquesta raó, si el *criptex* rep un cop, el tub amb vinagre es trenca i deixa anar el líquid que s'encarrega de destruir la informació.

15.2. La criptografia en situacions quotidianes

15.2.1. NIF (Número d'identificació personal)

El NIF és el sistema d'identificació tributària utilitzada a Espanya. Per calcular el NIF hem de fer la següent operació:

DNI mod 23

Al resultat d'aquesta operació li correspon una de les lletres de la taula següent:

0 = T	8 = P	13 = J
1 = R	9 = D	14 = Z
2 = W	20 = C	15 = S
3 = A	21 = K	16 = Q
4 = G	22 = E	17 = V
5 = M	10 = X	18 = H
6 = Y	11 = B	19 = L
7 = F	12 = N	

Per tant, cada persona té un NIF determinat depenent de les xifres del seu DNI (Document Nacional d'identitat).

15.2.2. Número d'un compte corrent

El número d'un compte corrent consta de 20 dígits:

Entitat	Sucursal		N° de compta
ABCD	EFGH	00	ABCDEFGHIJ

Es calculen de la següent manera:

- Els primers vuit dígits:

$$(7A + 3B + 6C + D + 2E + 4F + 8G + 5H) \bmod 11$$

- Els deu últims dígits:

$$(10A + 9B + 7C + 3D + 6E + F + 2G + 4H + 8I + 5J) \bmod 11$$

Això demostra que per garantir la seguretat de qualsevol tipus d'informació, de manera implícita apareix també la criptografia.

16. CONCLUSIÓ

Poder encriptar i desencriptar missatges a través dels números ha estat la meva finalitat durant tot el treball, així com adquirir els conceptes necessaris per poder entendre i saber explicar l'RSA.

Pel que fa a aquesta part, encriptar i desencriptar missatges amb l'RSA, he assolit els meus objectius tant amb un programa d'ordinador com treballant-ho a mà.

La criptografia és un món molt complex en el qual distingim una bona quantitat de termes que s'han de tenir en compte a l'hora de realitzar qualsevol pràctica criptogràfica.

Com s'ha observat, l'RSA és un algorisme que si s'utilitza formalment i es vol que el missatge encriptat arribi en bones condicions al receptor i que ningú el pugui arribar a desxifrar, s'ha de treballar amb números molt i molt grans. L'exemple que ofereixo al treball és molt simple per la qual cosa no serviria de res a l'hora de voler transmetre informació important. Per tant, qualsevol persona que vulgui xifrar o desxifrar un text mitjançant aquest algorisme, ha de disposar de les eines necessàries per poder realitzar tot el seguit d'operacions que aquest algorisme suposa.

És un algorisme que treballat a mà és molt vulnerable, ja que treballar d'aquesta manera amb números grans seria molt costós, per la qual cosa, xifrar un missatge en bones condicions requereix noves tecnologies.

Davant d'aquesta afirmació s'observa amb claredat que la criptografia ha estat influenciada d'una manera directa per l'evolució de les noves tecnologies, sent elles qui executin les tasques més dures i ofereixin les seves limitacions.

Per tal de poder posar un exemple del procés d'encriptació i desencriptació amb xifres relativament grans he utilitzat un programa d'ordinador anomenat Maple, que en qüestió de segons realitza operacions amb números de més d'un centenar de xifres

Al llarg de gairebé tot el treball, s'ha mencionat o parlat sobre la firma digital, ja que és una part fonamental per saber que l'emissor del missatge és la persona desitjada, ja que actua com una firma normal i que cada persona posseeix i amb la qual s'identifica.

Trobo que aquest aspecte és importantíssim per a garantir la seguretat i la màxima tranquil·litat entre els destinataris, pel fet de que si a un d'ells li arriba una firma que no es correspon amb la de l'altra persona o lleugerament canviada, ja pot desfer-se'n de la informació rebuda.

Com qualsevol algorisme, l'RSA posseeix unes dificultats, majoritàriament matemàtiques, que el caracteritzen. D'aquesta manera, si un algorisme d'aquest tipus no presentés cap mena de dificultat, la seva transmissió no seria cap obstacle, doncs aquí trobem la clau de la criptografia i que fa que actualment no hagi cap mètode veritablement segur de transmissió d'informació.

Tot i que aquesta no era la meva previsió de treball, treballar només un algorisme ha estat més bona idea que anomenar uns quants sense extreure'n tota la seva informació i funcionament. La dedicació a l'RSA ha estat un avantatge per introduir-me plenament en el món de la criptografia.

17. BIBLIOGRAFIA

1. ADDLINK [en línia] < http://www.addlink.es/productos.asp?pid=41 > [consulta: 26.07.07] Maple

2. ADEPTSCIENCE [en línia] < http://www.adeptscience.co.uk/products/mathsim/maple/powertools/cryptography/HTML/RSA.html > [consulta: 19.07.07] Using RSA

3. BROWN, Dan. *El Código Da Vinci*, Barcelona, Ediciones Urano, S.A.

4. CABALLERO GIL, Pino. *Introducción a la criptografía*, Ra-Ma.

5. CIBERPUNK [en línia] < http://www.ciberpunk.com/indias/enredadera_02.html > [consulta: 08.05.07] Como una enredadera y no como un árbol.

6. CREANGEL.COM [en línia] < http://www.creangel.com/drupal/?q=node/115&PHPSESSID=62e7eeb009e6af2f038744c6bf3480b8 > [consulta: 07.08.07] ¿Qué es una firma digital?

7. CRIPTOZONA [en línia] < httphttp://www.dat.etsit.upm.es/~mmonjas/cripto/00.html > [consulta: 10.07.07] Criptografia

8. DANIEL LERCH [en línia] < http://daniellerch.com/sources/doc/algoritmo_rsa.html > [consulta: 06.07.07] El algoritmo RSA y la factorización de números grandes

9. DELITOS INFORMÁTICOS [en línia] < http://www.delitosinformaticos.com/seguridad/criptografia.shtml > [consulta: 18.07.07] Introducción a la criptografia

10. DEPARTAMENTO DE DIDÁCTICA GENERAL Y DIDÁCTICAS ESPECÍFICAS [en línia] < http://www.dgde.ua.es/congresotic/public_doc/pdf/6283.pdf > [consulta: 16.07.07] Algoritmos matemáticos y criptográficos

11. DEPARTAMENTO DE MATEMÁTICA APLICADA [en línia] < http://www.dma.fi.upm.es/java/matematicadiscreta/aritmeticamodular/congruencias.html > [consulta: 27.06.97] Congruencias

12. DURÁN DÍAZ, Raúl; HERNÁNDEZ ENCINAS, Luís; MUÑOZ MASQUÉ, Jaime. *El criptosistema RSA*, Madrid, Ra-Ma.

13. EL HACKER [en línia] <http://foro.elhacker.net/index.php/topic,67109.0.html> [consulta: 15.04.2007]

14. EL PARALELEPIPEDO [en línia] <http://www.elparalelepipedo.org.ar/matematica/teoria-de-numeros.html > [consulta: 27.04.07] Teoría de números enteros

15. EL RINCÓN DE QUEVEDO [en línia] < http://rinconquevedo.iespana.es/rinconquevedo/criptografia/rsa2.htm > [consulta: 03.07.07] Introducción a la criptografia

16. GAUSSIANOS [en línia] < http://gaussianos.com/category/numeros-enteros/ > [consulta: 09.05.07]

17. KRIPTÓPOLIS [en línia] < http://www.kriptopolis.org/node/4550 > [consulta: 08.08.07]

18. KRIPTÓPOLIS [en línia] < http://www.kriptopolis.org/algoritmo-para-calcular-numeros-primos-muy-grandes > [consulta: 13.05.07] Algoritmo para calcular números primos muy grandes

19. LORDEPSYLON.NET [en línia] < http://www.lordepsylon.net/descargas/esteganografia.pdf > [consulta: 10.07.07] Esteganografía

20. LYCOS [en línia] < http://usuarios.lycos.es/teoriadenumeros/modular.html > [consulta: 16.06.07] Aritmètica modular. Congruències

21. MANKIEWICZ, RICHARD. Historia de las matemáticas. Del cálculo al caos, Edición Paidós.

22. MATEMÁTICAS [en línia] < http://www.matematicas.net/paraiso/cripto.php?id=rsa1 > [consulta: 15.07.07] Cifrado RSA (I)

23. MICROTECNOLOGIAS [en línia] < http://www.microteknologias.cl/bib_estegano.html > [consulta: 01.09.07] Esteganografia

24. MONDRAGON [no disponible] < http://mondragon.angeltowns.net/paradiso/Criba Eratostenes.html > [consulta: 13.05.07]

25. NEUMANN [en línia] < http://neumann.dma.fi.upm.es/docencia/primerciclo/matdiscreta/11M-1/Aplicaciones%20AritmMod.pdf > [consulta: 29.11.07] Aplicaciones de la aritmética modular

26. PITÁGORAS [en línia] < http://pitagoras.usach.cl/~crodriguez/manuales/programacion%20en%20maple.pdf > [consulta: 18.07.07] Matemática discreta

27. SEGURIDAD INFORMÁTICA [en línia] < http http://seguinfo.blogspot.com/2007/03/rsa-lanza-la-campaa-10-das-contra-el.html > [consulta: 21.08.07] Noticias de seguridad informática

28. UNIVERSIDAD AUTÓNOMA DE MADRID [en línia] < http://www.uam.es/proyectosinv/estalmat/Estalmat/susipablo02.pdf > [consulta: 13.09.07] El sistema RSA

29. UNIVERSIDAD DE BURGOS [en línia] < http://www.ubu.es/investig/aulavirtual/trabajos_04/Criptografia.pdf > [consulta: 29.05.07] Una introducción a la criptografia. El criptosistema RSA

30. UNIVERSIDAD DE NAVARRA [en línia] < ttp://mat21.etsii.upm.es/ayudainf/aprendainf/MapleV/mapleVr3.pdf > [consulta: 26.07.07] Maple V

31. UNIVERSITAT DE VALÈNCIA [en línia] < http://www.uv.es/asepuma/XII/comunica/bernal_martinez_sanchez_2.pdf > [consulta: 11.06.07] Encriptación en la comunicación de información electrònica

32. UNIVERSIDADES DE VIGO [en línia] < http://webs.uvigo.es/martapr/Docencia/MD/Guia/A_numeros.pdf > [consulta: 26.07.07] Matemática discreta

33. URJC CAMPUS DE MÓSTOLES [en línia] < http://www.escet.urjc.es/~matemati/md_iti/practicas/Libromaple03/LibroMaple031.html > [consulta: 05.07.07] Prácticas y problemas resueltos en Maple V

34. WIKIPEDIA [en línia] <http://es.wikipedia.org/wiki/Algoritmo> [consulta: 17.04.2007]

35. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/Exponenciaci%C3%B3n_binaria > [consulta: 21.05.07] Potència binària

36. WIKIPEDIA [en línia] < http http://es.wikipedia.org/wiki/Cifrado_C%C3%A9sar > [consulta: 11.05.07] Cifrado César

37. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/Claude_Elwood_Shannon > [consulta: 05.07.07] Claude Elwood Shannon

38. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/Criptoan%C3%A1lisis > [consulta: 03.07.07] Criptoanálisis

39. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/Congruencia > [consulta: 30.07.07] Congruencia

40. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/Maple> [consulta: 06.07.07] Maple (software)

41. WIKIPEDIA [en línia] < http:// http://es.wikipedia.org/wiki/N%C3%BAmero_de_identificaci%C3%B3n_fiscal > [consulta: 29.11.07] Número de Identificación Fiscal

42. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/Firma_digital > [consulta: 26.08.07] Firma digital

43. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/Historia_de_la_criptograf %C3%ADa > [consulta: 01.09.07] Historia de la criptografia

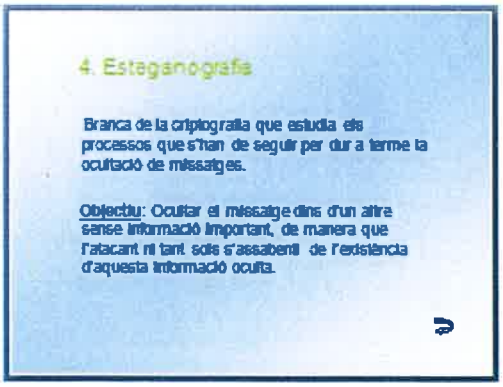
44. WIKIPEDIA [en línia] < http://es.wikipedia.org/wiki/RSA#Velocidad > [consulta: 06.08.07] RSA

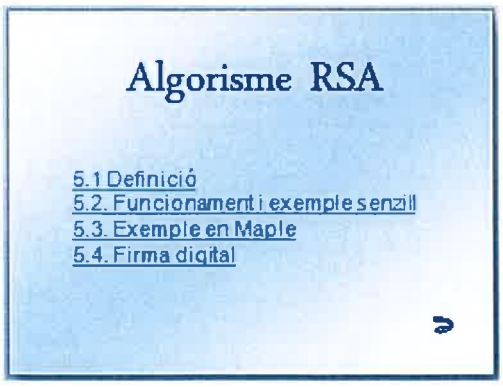
45. ZONAVIRUS [en línia] < http://www.zonavirus.com/datos/articulos/44/Firma_ Digital_Certificados_Digitales.asp > [consulta: 04.09.07] Firma digital y certificados digitales

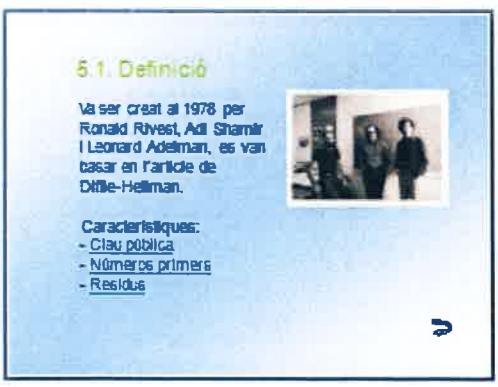
18. ANNEXOS

18.1. Fitxers de Maple

18.3. Diapositives de l'exposició







Clau pública

Números primers

p = 25290649873662354539

q = 7223636076479215820887

p x q = 182690450824991906030995570354761154560 93

Residus

Per calcular el NIF hem de fer la següent operació: DNI mod 23

Al resultat d'aquesta operació li correspon una de les lletres de la taula següent:

0 = T	8 = P	15 = S
1 = R	9 = D	16 = C
2 = W	10 = X	17 = V
3 = A	11 = B	18 = H
4 = G	12 = N	19 = L
5 = M	13 = J	
6 = Y	14 = Z	
7 = F		

5.2 Funcionament i exemple senzill

Generació de clau

- Triem dos nombres primers molt grans *p* i *q*:
  p = 3 q = 5
- Fem el producte dels nombres primers *n*:
  n forma part de la clau pública
  n = 16
- Calculem *phi_n* = (p-1) · (q-1):
  phi_n forma part de la clau privada
  phi_n = 8

- Escollim *d* tal que m.c.d (*d*, *phi_n*) = 1
  d forma part de la clau privada
  d = 3
- Calculem *e*, de tal manera que: $1 \leq e \leq phi\_n$, mitjançant la fórmula $e \cdot d \equiv 1 \pmod{phi\_n}$. Aquesta operació ens condueix a una equació diofàntica ja que $e \cdot d \equiv 1 \pmod{phi\_n}$ equival a $e \cdot d = phi\_n \cdot y + 1$.
  e forma part de la clau pública
  e = 3

Clau pública: n=15 e=3

Clau privada: phi_n=8 d=3

Xifrat de missatges

- Per xifrar un text, és necessari prèviament codificar-lo en un sistema numèric

  L A F A D A

  12 01 00 05 01 04 01

Clau pública: n=15 e=3

Clau privada: phi_n=8 d=3

- Un cop hem codificat el text en un sistema numèric el xifrem elevant el resultat de codificar el text en el sistema numèric a *e*

$12^3 = 1728$ $01^3 = 01$ $00^3 = 00$ $05^3 = 125$ $01^3 = 01$ $04^3 = 64$ $01^3 = 01$

Missatge a enviar: 1728 01 00 2160164 01

Desxifrat de missatges

El receptor del missatge per tant, rebrà la següent informació:

1728 01 00 2160164 01

Clau pública: n=15 e=3

Clau privada: phi_n=8 d=3

- Per desxifrar un text, hem de dur a terme la següent operació: text xifrat (*m*) = *x* mod n

$1728 = x \bmod 15 \rightarrow m = 3$ $01 = x \bmod 15 \rightarrow m = 1$ $00 = x \bmod 15 \rightarrow m = 0$ $2160 = x \bmod 15 \rightarrow m = 6$ $01 = x \bmod 15 \rightarrow m = 1$ $64 = x \bmod 15 \rightarrow m = 4$ $01 = x \bmod 15 \rightarrow m = 1$

- Una vegada tenim el valor de *x*, l'obtenim a dl realitzem una última operació: *m* mod n = text

$3^3 \bmod 15 \rightarrow 27 \bmod 15 = 12 \rightarrow$ L
$1^3 \bmod 15 \rightarrow 1 \bmod 15 = 1 \rightarrow$ A
$2^3 \bmod 15 \rightarrow 8 \bmod 15 = 8 \rightarrow$ F
$6^3 \bmod 15 \rightarrow 216 \bmod 15 = 6 \rightarrow$ F
$1^3 \bmod 15 \rightarrow 1 \bmod 15 = 1 \rightarrow$ A
$4^3 \bmod 15 \rightarrow 64 \bmod 15 = 4 \rightarrow$ D
$1^3 \bmod 15 \rightarrow 1 \bmod 15 = 1 \rightarrow$ A

Clau pública: n=15 e=3

Clau privada: phi_n=8 d=3

- Per últim, associem el número del text amb la lletra del sistema numèric que havíem fixat.

5.3. Exemple en Maple

Clau

Missatge xifrat

159526840572293491057875260430985996142
457029072208241850983813137948446015509
925907952682097733164449405236160016551
547021164269541348093289612822020776803
248601832791082930040869070628201982314
02278668

Missatge desxifrat

La Miriam m'ha demanat un full

5.4. Firma digital

La firma digital és un mecanisme mitjançant el qual el destinatari s'assegura de que qui li envia el missatge és verdaderament la persona que ell creu que és. Aquesta firma està composta per dígits més petits formats a través d'unes claus i uns procediments determinats.

(text xifrat, firma)

6. Curiositats

Màquina criptogràfica *Enigma*

Creada al segle XX per Arthur Scherbius

Xifrat en Maple

- Missatge xifrat

- Missatge desxifrat

Hola

18.4. Taules

Anàlisi de freqüències

Francès	E	S	A	R	N	U	T	L	I	O	Total
Freqüència %	15	8	6	5,5	5,4	4,8	4,7	4,6	4,5	4	54,5%

Anglès	E	T	A	O	I	N	S	H	R		Total
Freqüència %	10	8,2	7	6,5	6,4	6,3	6	4	3,6		58%

Espanyol			
Altes	Mitjanes	Baixes	Molt baixes
E – 16,78%	R – 4,94%	Y – 1,54%	J – 0,30%
A – 11,96%	U – 4,8%	Q – 1,53%	Ñ – 0,29%
O – 8,69 %	I – 4,15%	B – 0,92%	Z – 0,15%
L – 8,37%	T – 3,31%	H – 0,89%	X – 0,06%
S – 7,88%	C – 2,92%	G – 0,73%	K – 0,00%
N – 7,01%	P – 2,776%	F – 0,52%	W – 0,00%
D – 6,87%	M – 2,12%	V – 0,39%	

Sistema de Vigènere

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
...																										
25	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Rècords de factorització

Any	Dígits	Número	Autor(s)	Mètode	Hardmare
1970	39	$2^{128}+1$	Brillhart/Morrison	CFRAC	IBM Mainframe
1978	45	$2^{223}-1$	Wunderlich	CFRAC	IBM Mainframe
1981	47	$3^{225}-1$	Gerver	QS	HP-3000
1982	51	$5^{91}-1$	Wagstaff	CFRAC	IBM Mainframe
1983	63	$11^{93}+1$	Davis/Holdridge	QS	Cray
1984	71	$10^{71}-1$	Davis/Holdridge	QS	Cray
1986	87	$5^{128}+1$	Silverman	MPQS	LAN Sun-3's
1987	90	$5^{160}+1$	Silverman	MPQS	LAN Sun-3's
1988	100	$11^{104}+1$	Internet	MPQS	Comp. distribuida
1990	111	$2^{484}+1$	Lenstra/Manasse	MPQS	Comp. Distribuïda
1991	116	$10^{142}+1$	Lenstra/Manasse	MPQS	Comp. Distribuïda
1992	129	RSA-129	Atkins	MPQS	Comp. Distribuïda
1996	130	RSA-130	Montgomery	GNFS	Comp. Distribuïda
1998	140	RSA-140	Montgomery	GNFS	Comp. Distribuïda
1999	155	RSA-155	Montgomery	GNFS	Comp. Distribuïda
2003	160	RSA-160	BSI	GNFS	Comp. Distribuïda
2003	174	RSA-576	BSI	GNFS	Comp. distribuida

Nombres primers més grans actualment

Nombre prim	Nombre de dígits	Any del seu descobriment
$2^{13466917}-1$	4053946	2001
$2^{6972593}-1$	2098960	1999
$2^{3021377}-1$	909526	1998
$2^{2976221}-1$	895932	1997
$2^{1398269}-1$	420921	1996
$1361846^{65536}+1$	402007	2002
$1266062^{65536}+1$	399931	2002
$5 \cdot 2^{1320487}+1$	397507	2002
$1057476^{65536}+1$	394807	2002
$857678^{65536}+1$	388847	2002

19. ÍNDEX D' AUTORS RELACIONATS AMB LA CRIPTOGRAFIA

Autor **Pàgina**