ENSAM
Ecole Nationale Supérieure d'Arts et Métiers
Aix-en-Provence

Introduction à la cryptographie

Version 1.0
Juin 2003

Jean-Louis POSS

Table des matières

1 Aperçu historique **4**
1.1 Science du secret pendant 2500 ans... 4
1.1.1 Âge artisanal 4
1.1.2 Âge technique 7
1.1.3 DES et AES 7
1.1.4 Chiffrement en continu 8
1.2 ... science de la confiance depuis 25 ans. 8
1.2.1 Principe du chiffrement asymétrique 8
1.2.2 Fonctions de hachage 9

2 Rappels mathématiques **9**
2.1 Arithmétique sur les entiers 9
2.1.1 Divisibilité 9
2.1.2 PGCD 10
2.1.3 Congruences 11
2.2 Groupes finis 12
2.2.1 Définitions 12
2.2.2 Groupes cycliques 13
2.2.3 Logarithme discret 13
2.2.4 Indicatrice d'EULER 14
2.2.5 Résidus quadratiques 15
2.3 Corps finis 16
2.4 Courbes elliptiques 16

3 Chiffrement symétrique **16**
3.1 Chiffrement par blocs 16
3.2 DES (*Data Encryption Standard*) 17
3.3 AES (*Advanced Encryption Standard*) : Rijndael 19

4 Chiffrement asymétrique **20**
4.1 RSA 20
4.1.1 Description 20
4.1.2 Comment ça marche ? 20
4.1.3 Exemple 21
4.1.4 Choix des paramètres 21
4.1.5 Failles de RSA 22
4.2 EL GAMAL 22
4.2.1 Description 22
4.2.2 Comment ça marche ? 22
4.2.3 Exemple 23
4.2.4 Variantes 23
4.3 Échange de clefs 23
4.3.1 Attaque « man-in-the-middle » 23
4.3.2 Empreinte 23

4.3.3 Certificat de clef publique 24

5 Quelques protocoles **24**
5.1 DIFFIE-HELLMAN : échange public de clefs secrètes 24
5.1.1 Échange entre deux correspondants 24
5.1.2 Échange entre trois correspondants et plus 25
5.2 Authentification 25
5.2.1 Défi-réponse 25
5.2.2 Identification à divulgation nulle 25
5.3 SHA : *Secure Hash Algorithm* 27
5.4 DSA : *Digital Signature Algorithm* 28
5.4.1 Génération des clefs 28
5.4.2 Génération de signature 29
5.4.3 Vérification de signature 29
5.4.4 Pourquoi ça marche ? 29
5.5 SSL : *Secure Socket Layer* 29
5.5.1 SSL *Record protocol* 30
5.5.2 SSL *Handshake protocol* 30
5.6 Comparaison des systèmes symétriques et asymétriques 32

6 Sécurité des cartes à puce **32**

7 Législation **32**

8 PGP **34**
8.1 Contexte 34
8.2 Fonctionnement 34
8.3 Distribution des clefs 34

9 Quelques pointeurs **35**
9.1 Sites Web 35
9.2 Newsgroups 35

1 Aperçu historique

1.1 Science du secret pendant 2500 ans...

1.1.1 Âge artisanal

– Au V^{e} siècle avant J.-C. la scytale spartiate est un bâton de bois autour duquel on enroule une lanière de cuir ou une bande de papyrus ; le message est écrit parallèlement à l'axe. La lanière, une fois déroulée, semble couverte d'une suite de lettres incohérente ; le message est lu par le destinataire en enroulant la lanière sur une scytale de même diamètre. C'est un procédé de *transposition* (assez rudimentaire) : un tel procédé consiste à modifier l'ordre des caractères composant un message.

– Au I^{er} siècle avant J.-C. Jules CÉSAR utilise un chiffre de *substitution* : les caractères du message sont remplacés par d'autres. Le chiffre de CÉSAR consiste à décaler l'alphabet de trois rangs vers la droite :

Alphabet clair	a	b	c	...	x	y	z
Alphabet chiffré	D	E	F	...	A	B	C

– Ces techniques *cryptographiques* destinées à cacher le sens du message doivent être distinguées des techniques *stéganographiques* qui consistent à dissimuler le message lui-même. Au V^{e} siècle avant J.-C. un grec prévient Sparte de l'invasion de XERXÈS en inscrivant un message sur une tablette de bois, puis en le masquant avec de la cire. À la même époque HISTAÏAEUS écrivit à ARISTAGORAS de MILET sur le crâne rasé d'un esclave ; il attendit la repousse des cheveux avant d'envoyer le messager au destinataire. Les encres sympathiques, les micro-points... sont aussi des techniques stéganographiques.

– Pour garantir le secret des communications on a beaucoup utilisé les *codes*. Dans un code une lettre, un mot ou une phrase est remplacé par une lettre, un symbole ou un groupe de lettres ou de chiffres. Les tables de correspondance sont inscrites dans des livres qui peuvent être très volumineux ; les correspondants qui doivent partager le secret possèdent chacun un exemplaire. En cas de perte ou de vol il faut créer un nouveau code et redistribuer les documents. On parle aussi de *code* dans un autre contexte : code ASCII, alphabet MORSE, code BAUDOT... Il s'agit de représenter conventionnellement une information, sans préoccupation de secret.

– Le chiffre de CÉSAR est un peu amélioré si le décalage est variable. Cependant il n'y a que 26 possibilités, ce qui n'arrêtera pas longtemps les indiscrets.

– Ce chiffre est grandement amélioré en définissant une permutation des lettres de l'alphabet : *substitution monoalphabétique*. Cette permutation est généralement définie à l'aide d'une clef :

Alphabet clair	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
Alphabet chiffré	E	N	S	A	M	O	P	Q	R	T	U	V	W	X	Y	Z	B	C	D	F	G	H	I	J	K	L

Ce procédé a résisté plusieurs siècles à la perspicacité des cryptanalystes. Il a été cassé au IX^{e} siècle par un cryptanalyste arabe, AL-KINDI, grâce à l'analyse des fréquences : la fréquence des lettres permet de retrouver la substitution employée dès que le message a une longueur dépassant quelques dizaines de caractères. (Remarque : en 1969 Georges PEREC a écrit un roman de 200 pages, La Disparition, sans employer la lettre *e* ; ce roman a été traduit en anglais, A Void, par Gilbert ADAIR...)

– En 1586 Blaise de VIGENÈRE proposa un chiffre basé sur la *substitution polyalphabétique*. Un

carré de VIGENÈRE est formé de 26 chiffres de CÉSAR avec des décalages croissants.

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
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
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
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
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
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
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	F
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	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
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
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

Une clef permet de sélectionner la ligne utilisée pour chiffrer chaque caractère ; par exemple choisissons ENSAM pour clef.

clef	E	N	S	A	M	E	N	S	A	M	E	N	S	A	M	E	N	S	A	M	E	N	S
Texte clair	u	n	m	e	s	s	a	g	e	i	n	d	e	c	h	i	f	f	r	a	b	l	e
Texte chiffré	Y	A	E	E	E	W	N	Y	E	U	R	Q	W	C	T	M	S	X	R	M	F	Y	W

Le chiffre de VIGENÈRE ne peut s'attaquer par l'analyse des fréquences. Le nombre de clefs utilisables n'est pas limité. Ce chiffre a résisté plus de deux siècles et a été nommé « le chiffre indéchiffrable ». Il a finalement été brisé en 1854 par Charles BABBAGE ; la méthode a été publiée en 1863 par Friedrich KASISKI à qui on attribue parfois la découverte. BABBAGE a remarqué que certaines suites de lettres pouvaient apparaître à différents endroits du texte chiffré et qu'il était probable que ces répétitions soient dues au chiffrement du même texte clair avec la même partie de la clef. Ceci permet de déterminer la longueur de la clef ; on est alors ramené à l'analyse de plusieurs substitutions monoalphabétiques.

– C'est également en 1854 que le baron PLAYFAIR de SAINT-ANDREWS présenta le chiffre qui porte son nom et qui avait été inventé par son ami Charles WHEATSTONE (également inventeur

de « ponts » célèbres…). Il s'agit d'un chiffre bigrammatique, c'est-à-dire que l'on chiffre deux lettres simultanément. On construit un carré (ou un rectangle) à partir d'une clef :

E	N	S	A	M
B	C	D	F	G
H	I/J	K	L	O
P	Q	R	T	U
V	W	X	Y	Z

On divise le texte à chiffrer en groupes de deux lettres (bigrammes) ; si, *dans un même bigramme*, se présentent deux lettres identiques on intercale entre elles une lettre rare, *x* par exemple. On obtiendra, par exemple : *as-sa-sx-si-na-ts*. Pour chaque bigramme trois cas peuvent se présenter :

- Les deux lettres sont dans la même ligne : on les remplace par les lettres à leur droite. Par exemple *pu* est chiffré QP.
- Les deux lettres sont dans la même colonne : on les remplace par les lettres en dessous. Par exemple *la* est chiffré TF.
- Les deux lettres sont dans des lignes et colonnes différentes : on les remplace par les lettres qui forment avec elles le sommet d'un rectangle, la première étant située sur la même ligne que la première lettre du bigramme. Par exemple *et* est chiffré AP.

Le déchiffrement est aisé… si l'on connaît la clef. Le procédé est cryptanalysé par analyse des fréquences des bigrammes : la méthode a été publiée pour la première fois en 1914 par Joseph MAUBORGNE.

- Il existe un chiffre indéchiffrable : il a été conçu par Gilbert VERNAM en 1926 ; Claude SHANNON a prouvé en 1949 qu'il assure une confidentialité parfaite, c'est-à-dire que la connaissance du texte chiffré n'apporte *aucune information* sur le texte clair. Il suffit d'appliquer le chiffre de VIGENÈRE avec une clef aussi longue que le texte à chiffrer. Tout texte chiffré correspond à un texte quelconque en clair pour une clef bien choisie. La sécurité est absolue si :
  - la clef est aléatoire ;
  - elle est utilisée une fois seulement.

  Ce procédé (« clef aléatoire une fois » ou « one-time pad ») est encore en usage actuellement sous une forme légèrement différente : on réalise un XOR bit à bit entre le texte clair et la clef aléatoire. C'est ainsi que fonctionnerait le « téléphone rouge ». Le problème fondamental est le partage de la clef :
  - soit elle est vraiment aléatoire et elle doit être communiquée aux correspondants : risque de divulgation…
  - soit elle est pseudo-aléatoire et on peut l'attaquer par des méthodes statistiques.

  Le problème de la gestion des clefs est le plus délicat de la cryptographie.
- En 1881 Auguste KERCKHOFFS publie un ouvrage fondamental : *La cryptographie militaire*. En 64 pages il rassemble les connaissances cryptologiques de l'époque et énonce quelques principes qui guident encore les cryptologues. En particulier : « La sécurité d'un code cryptographique ne doit pas reposer sur le secret de l'algorithme, mais sur celui des clefs utilisées. ». Au contraire l'algorithme doit être largement diffusé ; il sera étudié par des spécialistes, ses faiblesses seront mises en évidence et il pourra être amélioré… ou abandonné.

1.1.2 Âge technique

Quel que soit le chiffre utilisé le chiffrement est fastidieux et les erreurs sont fréquentes. On a donc cherché très tôt à mécaniser les opérations. Leon Battista ALBERTI proposa en 1466 un cadran chiffrant : un disque fixe porte l'alphabet clair dans l'ordre et un disque mobile porte l'alphabet désordonné ; on peut ainsi réaliser aisément une substitution monoalphabétique. Au fil des années de nombreuses améliorations ont vu le jour : cylindre de JEFFERSON vers 1800, cylindre de BAZERIES en 1891…
La forme la plus achevée de ce type de machines à chiffrer est l'ENIGMA allemande utilisée pendant la dernière guerre mondiale. C'est un dispositif électro-mécanique qui réalise une substitution polyalphabétique. Un tambour chiffrant est formé de rotors (disques) en matériau isolant portant sur chaque face des contacts électriques ; chaque contact d'une face est relié à un contact de l'autre. Les liaisons sont différentes pour chaque rotor. Le nombre de rotors est passé de trois rotors choisis parmi cinq à quatre parmi huit. Ces rotors tournent à des vitesses différentes après chaque caractère chiffré, ce qui donne quelques millions d'alphabets de substitution. De plus la machine était munie d'un tableau de connexions à fiches, ce qui portait le nombre de clefs à 159.10^{18} environ. La cryptanalyse de l'ENIGMA a été réalisée par les polonais puis, surtout, par les anglais. Sous la direction d'Alan TURING le gouvernement britannique regroupa à Bletchley Park de jeunes universitaires : mathématiciens, mais aussi scientifiques, historiens, linguistes, cruciverbistes… À la fin du conflit plus de sept mille personnes travaillaient pour l'opération Ultra. C'est pour « casser » l'ENIGMA que furent fabriqués les COLOSSUS qui sont, avec l'ENIAC, les premiers ordinateurs. Mais la technologie n'aurait pas permis le succès sans le génie de TURING.

1.1.3 DES et AES

Après guerre la recherche en cryptologie s'est limitée au domaine militaire. Au début des années 70, avec le développement de l'informatique et des réseaux, le besoin apparut de sécuriser le stockage et le transfert de données dans le domaine civil. En 1973 le NBS (*National Bureau of Standards*), devenu depuis le NIST (*National Institute of Standards and Technology*) a lancé un appel d'offre pour un algorithme de chiffrement offrant un niveau de sécurité élevé. IBM proposa *Lucifer*, algorithme développé au début des années 70, qui fut évalué par la NSA (*National Security Agency*) et publié en 1976 sous le nom de *Data Encryption Standard* (voir § 3.2).
C'est un algorithme de chiffrement par blocs ; la plupart des algorithmes de chiffrement par blocs utilisent les principes de confusion et de diffusion :
– *confusion* : en mixant opérations linéaires et non linéaires on cache les relations entre le clair, le chiffré et la clef ;
– *diffusion* : par des transpositions on fait dépendre chaque bit du chiffré de tous les bits du clair et de la clef.
Les progrès de la cryptanalyse et de la vitesse d'exécution des ordinateurs ont permis de casser le DES ; il est devenu nécessaire de le remplacer. Le NIST a lancé le 12 septembre 1997 un appel à contribution mondial pour la définition d'un nouvel algorithme, l'AES (*Advanced Encryption Standard*) :
– blocs de 128 bits ;
– taille de clef variable dont 128, 192 et 256 bits ;
– libre de droits ;
– simple et rapide ;
– efficace : doit pouvoir être implémenté sur carte à puce.

Cinq propositions ont été retenues en août 1999 parmi quinze reçues ; le vainqueur, une équipe belge dirigée par Joan DAEMEN et Vincent RIJMEN, a été désigné le 2 octobre 2000 : l'algorithme se note Rijndael (voir § 3.3). Après une dernière période d'appel à commentaires il a été officialisé le 26 mai 2002 comme le standard FIPS-197 (*Federal Information Processing Standard*). Il est utilisé par le gouvernement américain et par beaucoup d'autres organisations dans le monde pour protéger les informations sensibles ; il remplacera progressivement le DES.
D'autres algorithmes de chiffrement par blocs sont aussi utilisés : FEAL, IDEA, CAST, RC5, Blowfish…

1.1.4 Chiffrement en continu

Pour des usages militaires ou gouvernementaux essentiellement on utilise des algorithmes de chiffrement en continu qui opèrent sur des flots de données qu'ils chiffrent (ou déchiffrent) bit à bit (ou parfois octet par octet) : chaque bit du clair est « xoré » avec un bit produit par un générateur pseudo-aléatoire (GPA). Le déchiffrement est réalisé avec la même suite-clef. Le chiffrement est synchrone. La sécurité dépend essentiellement du GPA.

1.2 … science de la confiance depuis 25 ans.

Tous les algorithmes évoqués jusqu'à présent sont *symétriques* en ce sens que la même clé est utilisée pour le chiffrement et le déchiffrement. Le problème essentiel de la cryptographie symétrique est la distribution des clefs : pour que n personnes puissent communiquer de manière confidentielle il faut $n(n-1)/2$ clefs.
En 1976 Whitfield DIFFIE et Martin HELLMAN[1] présentèrent à la *National Computer Conference*, puis publièrent dans leur article fondateur « *New Directions in Cryptography* », le concept de *clef publique* et d'*algorithme asymétrique* : il n'est pas nécessaire que la clef utilisée pour le chiffrement soit la même que celle utilisée pour le déchiffrement. La clef de chiffrement peut être publiée largement (« clef publique »), la clef de déchiffrement (« clef privée ») restant secrète et connue de son seul propriétaire : le problème de l'échange confidentiel de clef disparaît.
Un système asymétrique est basé sur l'utilisation de *fonction à sens unique avec trappe* :
– connaissant x il est facile de calculer $f(x)$;
– connaissant $f(x)$ il est très difficile de calculer x ;
– connaissant $f(x)$ et une information supplémentaire (« trappe ») il est facile de calculer x.
Malgré tous leurs efforts DIFFIE et HELLMAN ne purent présenter un tel algorithme et le premier système à clef publique fut proposé en 1978 par Ronald RIVEST, Adi SHAMIR et Leonard ADLEMAN ; il est connu sous l'acronyme RSA. Sa compréhension nécessite quelques connaissances mathématiques : il sera présenté ultérieurement, de même que les autres systèmes du même type.

1.2.1 Principe du chiffrement asymétrique

Schématiquement un système asymétrique est composé de deux algorithmes : l'algorithme de chiffrement, $\mathcal{C}$, est public et l'algorithme de déchiffrement, $\mathcal{D}$, est privé. Ils vérifient :
– pour tout message m, $\mathcal{D}(\mathcal{C}(m)) = m$ et aussi $\mathcal{C}(\mathcal{D}(m)) = m$: $\mathcal{C}$ et $\mathcal{D}$ commutent ;

[1] D'après DIFFIE le premier protocole d'échange public de clef a été imaginé par Ralph MERKLE. Il semble que la cryptographie à clef publique ait été inventée antérieurement et indépendamment au *Government Communications Headquarters* de Cheltenham par James ELLIS, Clifford COCKS et Malcom WILLIAMSON, mais que le secret militaire ait interdit sa publication.

– connaissant $\mathcal{C}$ il est (virtuellement) impossible de déterminer $\mathcal{D}$;
– il est facile de générer des couples $(\mathcal{C}, \mathcal{D})$.

Le concept de chiffrement asymétrique a considérablement élargi le domaine d'application de la cryptographie : le chiffrement symétrique permet de garantir la *confidentialité* (le « secret »). Le chiffrement asymétrique permet également de garantir la *confidentialité* : le message est chiffré par l'expéditeur en utilisant la clé *publique* du destinataire et celui-ci procède au déchiffrement à l'aide de sa clef *privée*. Mais il permet en outre l'*authenticité*, l'*intégrité* et la *non-répudiation* (la « confiance »).

– *Authenticité* et *non-répudiation* : pour authentifier un message m, Alice le chiffre *avec sa clef privée*, $\mathcal{D}_A$, et expédie à Bob le couple $(m, \mathcal{D}_A(m))$. Bob connaît la clef publique d'Alice et peut donc calculer $\mathcal{C}_A(\mathcal{D}_A(m))$; il retrouve m qu'il peut comparer au clair. Il est certain que le message émane d'Alice car *elle seule* connaît $\mathcal{D}_A$: authenticité. De plus Alice ne peut répudier ce message car elle seule a pu le signer.
– *Intégrité* : Bob est en outre assuré que le message n'a pas été modifié. Si Ève intercepte la communication elle peut modifier le message m, mais elle ne peut modifier en conséquence $\mathcal{D}_A(m)$: son tripatouillage sera démasqué par Bob.
– Il est possible par ailleurs d'assurer la confidentialité en chiffrant le couple $(m, \mathcal{D}_A(m))$ avec la clef publique de Bob, $\mathcal{C}_B$.

1.2.2 Fonctions de hachage

En pratique cette technique serait très onéreuse car elle double la taille des messages à transmettre. On peut limiter le surcoût par l'emploi de *fonctions de hachage* ; une telle fonction associe à un message de longueur quelconque une *empreinte* de longueur fixe (160 bits par exemple pour SHA) : $m \mapsto \mathcal{H}(m)$. Une telle fonction $\mathcal{H}$ ne peut donc être injective, c'est-à-dire que deux messages différents peuvent avoir la même empreinte : il y a une *collision*. Une bonne fonction de hachage est telle qu'il est « pratiquement » impossible de former deux documents ayant la même empreinte, donc de modifier un document sans modifier son empreinte. Pour garantir l'authenticité et l'intégrité d'un message Alice signe, c'est-à-dire chiffre avec $\mathcal{D}_A$, l'empreinte du message et transmet à Bob le couple $(m, \mathcal{D}_A(\mathcal{H}(m)))$, éventuellement après l'avoir chiffré avec $\mathcal{C}_B$ pour garantir la confidentialité. Bob déchiffre avec $\mathcal{D}_B$ si besoin, applique $\mathcal{C}_A$ à la signature $\mathcal{D}_A(\mathcal{H}(m))$ et compare le résultat avec $\mathcal{H}(m)$ calculé à partir de m.

Le chiffrement asymétrique est malheureusement beaucoup plus lent que le chiffrement symétrique : RSA est environ mille fois plus lent que le DES. On utilise donc souvent des systèmes hybrides où le chiffrement asymétrique sert uniquement à sécuriser l'échange de la clef symétrique qui est choisie aléatoirement pour chaque message.

2 Rappels mathématiques

2.1 Arithmétique sur les entiers

2.1.1 Divisibilité

Définition 1. *Soient a et b deux entiers relatifs ; on dit que b* divise *a et on note $b|a$ s'il existe un entier c tel que $a = b\,c$.*

Définition 2. *Un entier positif est* premier *s'il n'admet pas d'autres diviseurs que 1 et lui-même.*

2, 3, 5, 7, 11… sont premiers. 1 n'est pas premier.

Théorème 1. *Il existe une infinité de nombres premiers.*

Remarque : on ne connaît pas de méthode pour déterminer le n-ième nombre premier sans avoir calculé ceux qui le précèdent.

Théorème 2. *Tout nombre entier naturel s'écrit comme un produit de nombres premiers ; cette décomposition en facteurs premiers est unique (si l'on range les facteurs par ordre croissant).*

La décomposition en facteurs premiers est aisée pour de petits nombres, mais elle est très longue pour de grands nombres, même lorsqu'il n'y a que deux facteurs premiers ; le dernier record date du mois d'août 1999 : un nombre de 155 chiffres (512 bits) a été factorisé en produit de deux nombres premiers de 75 chiffres par 300 ordinateurs en parallèle pendant 7 mois.
La sécurité du système asymétrique RSA (cf. § 4.1) est basée sur la difficulté de la factorisation.
Fonctions *Mathematica* associées à la divisibilité : `Divisors`, `PrimeQ`, `Prime`, `FactorInteger`, ainsi que la bibliothèque `NumberTheory'PrimeQ'`.

2.1.2 PGCD

Définition 3. *Le* PCGD *de deux entiers naturels a et b est le plus grand élément de l'ensemble de leurs diviseurs communs ; on le note $a \wedge b$.*
Si $a \wedge b = 1$ a et b sont premiers entre eux.

Le PGCD de deux entiers relatifs est celui de leurs valeurs absolues.

Théorème 3. *Les diviseurs communs à deux entiers naturels sont les diviseurs de leur PGCD.*

Théorème 4 (Identité de BACHET-BÉZOUT). *Soient a et b deux entiers relatifs. Il existe deux entiers relatifs u et v tels que :*

$$au + bv = a \wedge b.$$

Le PGCD de deux nombres s'obtient aisément à partir de leurs décompositions en facteurs premiers ; mais cette décomposition peut être très difficile si les nombres sont grands. Heureusement il existe un algorithme efficace.

Algorithme 1 (EUCLIDE). *Soient a et b deux entiers naturels avec $a \geq b$.*
while $b \neq 0$ ***do***
$r \leftarrow a \mod b$; $a \leftarrow b$; $b \leftarrow r$.
end while
return(a).

Cet algorithme peut être étendu pour fournir, outre le PGCD de a et b, deux entiers premiers entre eux u et v tels que $au + bv = a \wedge b$.

Algorithme 2 (EUCLIDE étendu). *Soient a et b deux entiers tels que $a \geq b \geq 1$.*
$x_2 \leftarrow 1$; $x_1 \leftarrow 0$; $y_2 \leftarrow 0$; $y_1 \leftarrow 1$.
while $b > 0$ ***do***
$q \leftarrow \lfloor a/b \rfloor$; $r \leftarrow a - qb$; $x \leftarrow x_2 - qx_1$; $y \leftarrow y_2 - qy_1$.
$a \leftarrow b$; $b \leftarrow r$; $x_2 \leftarrow x_1$; $x_1 \leftarrow x$; $y_2 \leftarrow y_1$; $y_1 \leftarrow y$.
end while
$d \leftarrow a$; $u \leftarrow x_2$; $v \leftarrow y_2$; *return* (d, u, v).

Fonctions *Mathematica* : `GCD`, `ExtendedGCD`.

2.1.3 Congruences

Définition 4. *Soient a et b deux entiers relatifs ; si $a-b$ est multiple de $n \in \mathbb{N}$ on dit que a est* congru à b modulo n *et on écrit*

$$a \equiv b \pmod{n}.$$

C'est une relation d'équivalence sur $\mathbb{Z}$; l'ensemble des classes d'équivalence $\{\overline{0}, \overline{1}, \ldots, \overline{n-1}\}$ est noté $\mathbb{Z}/n\mathbb{Z}$ ou $\mathbb{Z}_n$.

Théorème 5.

$$\begin{cases} a_1 \equiv a_2 \pmod{n} \\ b_1 \equiv b_2 \pmod{n} \end{cases} \Longrightarrow \begin{cases} a_1 + b_1 \equiv a_2 + b_2 \pmod{n} \\ a_1\, b_1 \equiv a_2\, b_2 \pmod{n} \end{cases}$$

On peut donc munir $\mathbb{Z}_n$ d'une addition et d'une multiplication en posant

$$\overline{a} + \overline{b} = \overline{a+b}, \qquad \overline{a} \times \overline{b} = \overline{a\,b}.$$

Théorème 6. *L'addition dans $\mathbb{Z}_n$ est commutative et associative ; $\overline{0}$ est un élément neutre et tout élément a un opposé.*
La multiplication dans $\mathbb{Z}_n$ est commutative et associative ; $\overline{1}$ est un élément neutre.
La multiplication est distributive par rapport à l'addition.

Ces propriétés s'énoncent : $(\mathbb{Z}_n, +, \times)$ a une structure d'anneau commutatif et unitaire.
Certains éléments de $\mathbb{Z}_n$ n'ont pas d'inverse ; il existe même des éléments non nuls dont le produit est nul (diviseurs de zéro).
Exemple : dans $\mathbb{Z}_6$ on a $\overline{3} \times \overline{4} = \overline{0}$.

Théorème 7. *Conditions d'inversibilité :*
– *$\overline{k}$ est inversible dans $\mathbb{Z}_n \Longleftrightarrow k \wedge n = 1$.*
– *Si $k \wedge n \neq 1$ il existe $p \in \mathbb{Z}$ tel que $\overline{k} \times \overline{p} = \overline{0}$.*
– *Si n est un nombre premier tous les éléments de $\mathbb{Z}_n$ sont inversibles sauf $\overline{0}$.*

La détermination de l'inverse de $\overline{k}$ dans $\mathbb{Z}_n$ s'effectue grâce à l'algorithme d'EUCLIDE étendu. Puisque $k \wedge n = 1$ il existe u et v tels que $k\,u + n\,v = 1$; d'où $\overline{k} \times \overline{u} = \overline{1}$, c'est-à-dire que $\overline{u}$ est l'inverse de $\overline{k}$.

Théorème 8 (Théorème chinois des restes). *Soient n_1 et n_2 deux entiers naturels premiers entre eux. Les entiers x qui vérifient les congruences simultanées :*

$$\begin{cases} x \equiv r_1 \pmod{n_1} \\ x \equiv r_2 \pmod{n_2} \end{cases}$$

sont de la forme $x \equiv r_2(u_1\, n_1) + r_1(u_2\, n_2) \pmod{n_1\, n_2}$ où u_1 et u_2 vérifient l'identité de BÉZOUT $n_1\, u_1 + n_2\, u_2 = 1$.

Fonctions *Mathematica* : `Mod`, `NumberTheory'NumberTheoryFunctions'ChineseRemainder`.

2.2 Groupes finis

2.2.1 Définitions

Définition 5. *Un* groupe *est un couple $(G, *)$ formé d'un ensemble G et d'une loi $*$ de composition interne sur G, $(a, b) \to a * b$, qui vérifie les propriétés suivantes.*
– *Associativité :* $\forall (a, b, c) \in G^3,\ a * (b * c) = (a * b) * c$.
– *Existence d'un élément neutre :* $\exists e \in G,\ \forall a \in G,\ a * e = e * a = a$.
– *Tout élément a un symétrique :* $\forall a \in G,\ \exists a' \in G,\ a * a' = a' * a = e$.

Si la loi est commutative ($\forall (a, b) \in G^2,\ a * b = b * a$) le groupe est dit *commutatif* ou *abélien*.
La loi du groupe est souvent notée additivement (l'élément neutre est noté 0 et le symétrique de a est son opposé $(-a)$) ou multiplicativement (l'élément neutre est noté 1 et le symétrique de a est son inverse a^{-1}).
Exemples :
– $(\mathbb{Z}, +)$, $(\mathbb{R}, +)$, $(\mathbb{R}^*, .)$, $(\mathbb{Z}_n, +)$ sont des groupes abéliens.
– $(S_n, \circ)$, où S_n représente les permutations des n premiers entiers, est un groupe non commutatif.
– L'ensemble des éléments inversibles de $\mathbb{Z}_n$ est un groupe multiplicatif, noté $\mathbb{Z}_n^\times$.

$\times$	1	3	9	7
1	1	3	9	7
3	3	9	7	1
9	9	7	1	3
7	7	1	3	9

$\mathbb{Z}_{10}^\times$

$\times$	1	3	5	7
1	1	3	5	7
3	3	1	7	5
5	5	7	1	3
7	7	5	3	1

$\mathbb{Z}_8^\times$

Définition 6. *Le nombre d'éléments (ou cardinal) d'un groupe G s'appelle son* ordre *et se note $|G|$.*

S'il est fini on parle de *groupe fini* ; c'est ce que nous supposerons par la suite.

Définition 7. *Deux groupes $(G, *)$ et $(G', \odot)$ sont* isomorphes *s'il existe une bijection $f : G \mapsto G'$ telle que*

$$\forall (x, y) \in G^2,\ f(x * y) = f(x) \odot f(y).$$

Les deux groupes donnés par les tables suivantes ne sont pas isomorphes :

+	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

$\mathbb{Z}_4$

$\odot$	e	a	b	c
e	e	a	b	c
a	a	e	c	b
b	b	c	e	a
c	c	b	a	e

Viergruppe

Définition 8. *Un* sous-groupe *est une partie d'un groupe qui possède une structure de groupe pour la loi induite.*

Théorème 9. *Une partie H d'un groupe G est un sous-groupe si et seulement si*

$$\forall (a,b) \in H^2,\ a * b' \in H.$$

Théorème 10 (LAGRANGE). *L'ordre de tout sous-groupe d'un groupe fini divise l'ordre du groupe.*

L'exponentiation dans un groupe multiplicatif étant très utilisée en cryptographie, nous donnons un algorithme efficace pour le calcul de g^e :

Algorithme 3 (Exponentiation). *Soient g un élément du groupe multiplicatif G et $e \in \mathbb{N}$.*
$S \leftarrow g$; $k \leftarrow e$; $A \leftarrow 1$.
while $k \neq 0$ ***do***
  if k *est impair* ***then***
    $A \leftarrow A\,S$
  end if
  $k \leftarrow \lfloor k/2 \rfloor$; $S \leftarrow S^2$.
end while
return(A).

Fonction *Mathematica* : `PowerMod`.

2.2.2 Groupes cycliques

Théorème 11. *Soient $(G,.)$ un groupe fini, noté multiplicativement, et $a \in G$.*
L'ensemble $\langle a \rangle = \{a^k \mid k \in \mathbb{N}^*\}$ des puissances de a est un sous-groupe de G, le sous-groupe engendré *par a.*

Définition 9. *L'*ordre *de $a \in G$ est l'ordre du sous-groupe $\langle a \rangle$; il divise l'ordre du groupe.*

Théorème 12. *Soit $(G,.)$ un groupe fini : $\forall a \in G,\ a^{|G|} = e$.*

Théorème 13. *Si p est un nombre premier il existe un seul groupe d'ordre p (à un isomorphisme près) : $(\mathbb{Z}_p, +)$. Tout élément différent de l'élément neutre est générateur.*

Définition 10. *Un groupe est* cyclique *s'il est fini et engendré par un de ses éléments.*

Théorème 14. *Soit $(G,.)$ un groupe cyclique d'ordre $n = |G|$ et g un générateur de G.*
- *Tous les sous-groupes de G sont cycliques.*
- *Si d divise n, G possède un unique sous-groupe d'ordre d : il est formé des éléments de la forme g^k où k est divisible par n/d.*
- *L'ordre de g^k est $n/(k \wedge n)$.*

2.2.3 Logarithme discret

Définition 11. *Soit $(G,.)$ un groupe noté multiplicativement, a et b deux éléments de G. Un* logarithme discret *de b dans la base a est un élément $x \in \mathbb{N}$ tel que $a^x = b$.*

On n'est assuré, en général, ni de l'existence, ni de l'unicité du logarithme discret.

Théorème 15. *Soit $(G,.)$ un groupe cyclique et g un générateur de G ; quel que soit $b \in G$ il existe un unique $x \in \{0, 1, \ldots, |G| - 1\}$ tel que $g^x = b$: c'est le logarithme de b dans la base g.*

La détermination du logarithme discret est un problème difficile (si l'ordre de G est grand, bien sûr) : cette difficulté est à la base de certaines techniques cryptographiques.
Fonction *Mathematica* : `MultiplicativeOrder`.

2.2.4 Indicatrice d'EULER

Définition 12. *L'*indicatrice d'EULER *est l'application* $\varphi : \mathbb{N}^* \mapsto \mathbb{N}$ *telle que* $\varphi(n)$ *est le nombre d'éléments inversibles de* $\mathbb{Z}_n$.

$\varphi(n)$ est aussi le nombre d'entiers positifs inférieurs à n et premiers avec n.
En appliquant le théorème 12 il vient :

Théorème 16 (EULER). *Si* a *et* n *sont deux entiers premiers entre eux alors*

$$a^{\varphi(n)} \equiv 1 \pmod{n}.$$

Théorème 17. *Si* p *est un nombre premier et* $u \in \mathbb{N}^*$ *alors*

$$\varphi(p^u) = p^{u-1}(p-1).$$

En particulier, si p est premier on a : $\varphi(p) = p - 1$. Le théorème 16 donne alors :

Théorème 18 (FERMAT). *Si* p *est premier et* a *non divisible par* p *alors*

$$a^{p-1} \equiv 1 \pmod{p}.$$

Ce théorème permet de montrer qu'un nombre n'est pas premier : puisque

$$2 \wedge 254371345897 = 1$$

et

$$2^{254371345896} \equiv 148773232990 \pmod{254371345897}$$

on peut conclure que 254371345897 n'est pas premier.
Mais il ne peut permettre de montrer qu'un nombre est premier : il existe des nombres composés (non premiers) n tels que, pour tout nombre k premier avec n, on ait $k^{n-1} \equiv 1 \pmod{n}$; ce sont les nombres de CARMICHAEL. Par exemple : $n = 561$.

Théorème 19 (Wilson). p *est premier si et seulement si* $(p-1)! \equiv -1 \pmod{p}$.

Cette caractérisation des nombres premiers est numériquement inefficace lorsque p est grand.

Théorème 20. *Si* m *et* n *sont premiers entre eux l'application* h *de* $\mathbb{Z}_{mn}$ *dans* $\mathbb{Z}_m \times \mathbb{Z}_n$ *définie par*

$$h(\overline{a}) = (\overline{a \pmod{m}}, \overline{a \pmod{n}})$$

est bijective.

Si $h(\overline{a}) = h(\overline{b})$ alors $a - b$ est divisible par m et par n, donc par mn, c'est-à-dire que $\overline{a} = \overline{b}$: h est injective, donc bijective. Le théorème 8 permet de déterminer effectivement h^{-1}.

Théorème 21. *La restriction de* h *à* $\mathbb{Z}_{mn}^\times$ *est une bijection de* $\mathbb{Z}_{mn}^\times$ *sur* $\mathbb{Z}_m^\times \times \mathbb{Z}_n^\times$.

Corollaire 1.

$$m \wedge n = 1 \Longrightarrow \varphi(m\,n) = \varphi(m)\,\varphi(n).$$

Théorème 22. *Si $n = {p_1}^{r_1}\,{p_2}^{r_2}\ldots{p_k}^{r_k}$ est la décomposition de n en facteurs premiers alors*

$$\varphi(n) = {p_1}^{r_1-1}\,(p_1-1)\,{p_2}^{r_2-1}\,(p_2-1)\ldots{p_k}^{r_k-1}\,(p_k-1),$$

qui s'écrit aussi

$$\frac{\varphi(n)}{n} = \prod_{i=1}^{k}\Big(1-\frac{1}{p_i}\Big).$$

Remarques :

1. $\varphi(n)/n$ est la probabilité de choisir au hasard dans $\{1,2,\ldots,n\}$ un nombre qui est premier avec n. Cette probabilité est voisine de 1 si les facteurs premiers de n sont grands.
2. Le calcul de $\varphi(n)$ est aisé…lorsqu'on sait décomposer n en facteurs premiers, ce qui peut être très difficile lorsque n est grand. Peut-on espérer trouver un meilleur algorithme ? Sans doute pas : le calcul de $\varphi(n)$ est de même difficulté que la factorisation. En effet supposons que $n = p\,q$ avec p et q premiers ; on a : $\varphi(n) = \varphi(p)\,\varphi(q) = (p-1)(q-1) = n+1-(p+q)$. Si l'on connaît $\varphi(n)$ on a $p+q = \varphi(n)-(n+1)$ et $p\,q = n$, donc on sait calculer p et q, c'est-à-dire factoriser n.

Table des premières valeurs de $\varphi(n)$:

$\varphi(n)$	n					
1	1	2				
2	3	4	6			
4	5	8	10	12		
6	7	9	14	18		
8	15	16	20	24	30	
10	11	22				
12	13	21	26	28	36	42
16	17	32	34	40	48	60
18	19	27	38	54		
20	25	33	44	50	66	

Fonctions *Mathematica* : `EulerPhi`, `CarmichaelLambda`.

2.2.5 Résidus quadratiques

Définition 13. *Soit $n \in \mathbb{N}^*$; $a \in \mathbb{Z}$ est un* résidu quadratique modulo n *s'il existe $x \in \mathbb{Z}$ tel que $x^2 \equiv a \pmod{n}$.*

Il est aisé de savoir si $a \in \mathbb{Z}$ est un résidu quadratique modulo n.
Il est aisé de déterminer $x \in \mathbb{Z}$ tel que $x^2 \equiv a \pmod{n}$ lorsque n est un nombre premier impair ; il est donc également aisé, grâce au théorème chinois (voir th. 8, page 11), de déterminer $x \in \mathbb{Z}$ tel que $x^2 \equiv a \pmod{n}$ lorsqu'on sait factoriser n en produit de facteurs premiers impairs.
Plus précisément, on montre que la difficulté de la recherche d'une racine carrée d'un résidu quadratique modulo n est équivalente à celle de la factorisation de n. Ce problème est à la base de différents protocoles cryptographiques, en particulier l'identification à divulgation nulle de FEIGE-FIAT-SHAMIR (voir § 5.2.2).

Fonctions *Mathematica* associées aux résidus quadratiques : `JacobiSymbol` et `SqrtMod` dans la bibliothèque `NumberTheory‘NumberTheoryFunctions‘`.

2.3 Corps finis

Les corps finis interviennent en cryptographie dans deux domaines.
– Les polynômes sur un corps fini sont utilisés dans l'algorithme Rijndael (voir § 3.3) ainsi que dans l'étude des registres à décalage produisant des suites pseudo-aléatoires.
– Les courbes elliptiques sur un corps fini permettent d'utiliser des clefs moins longues pour une même sécurité.

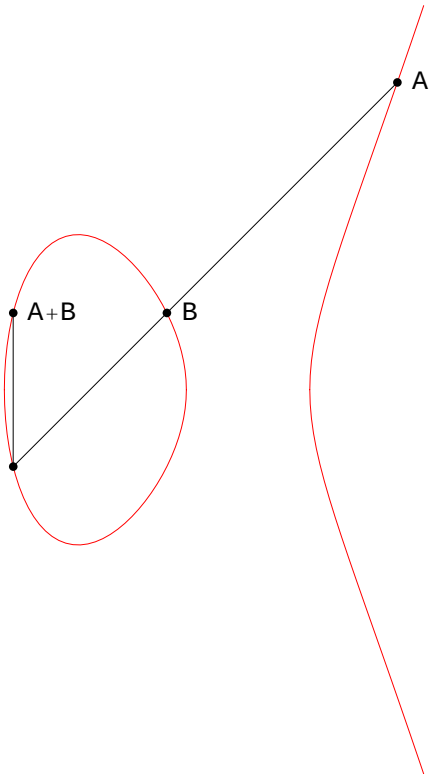
2.4 Courbes elliptiques

Définition 14. *Une* courbe elliptique *sur un corps K est l'ensemble des points $(x, y) \in K^2$ qui vérifient l'équation*

$$y^2 = x^3 + ax + b, \qquad (a, b) \in K^2.$$

On peut définir une « addition » sur une courbe elliptique comme le montre schématiquement la figure ci-contre ; la courbe elliptique est ainsi munie d'une structure de groupe. On pourra donc calculer des logarithmes discrets…
Les méthodes cryptographiques définies dans ce cadre sont plus efficaces que celles utilisant les groupes classiques.

Bibliothèque *Mathematica* utilisant les courbes elliptiques :
`NumberTheory‘FactorIntegerECM‘`.



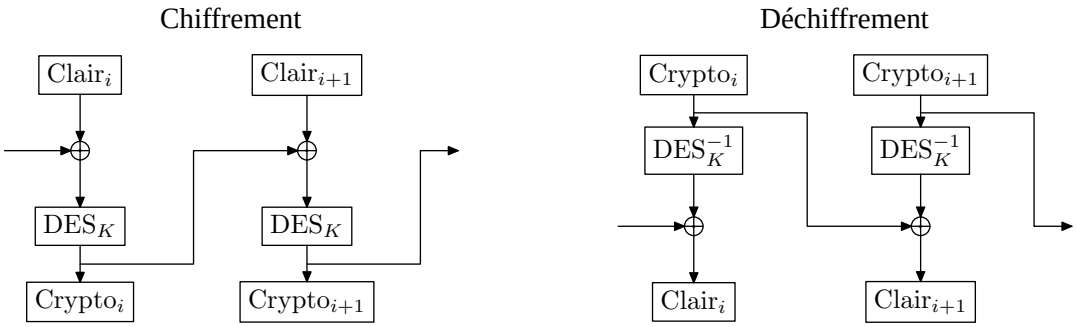
$y^2 = x^3 - 4x + 1$

3 Chiffrement symétrique

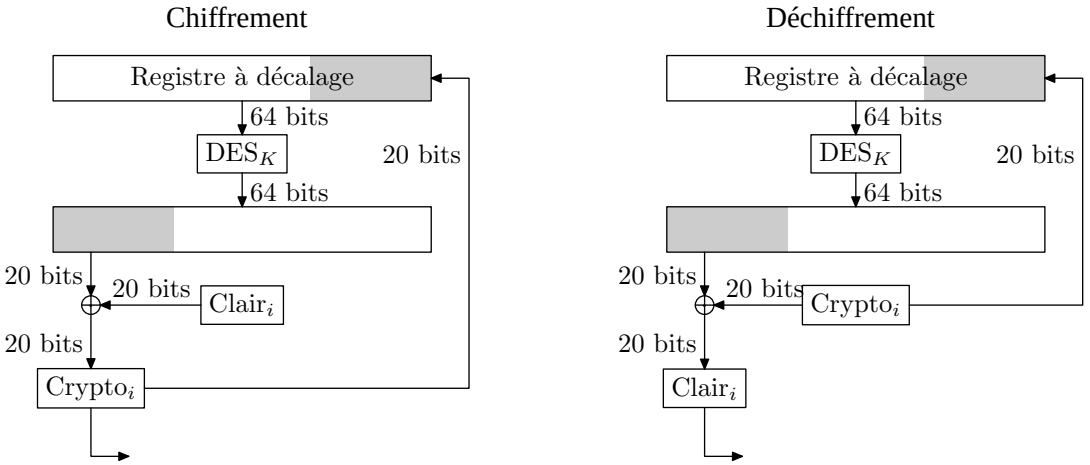
3.1 Chiffrement par blocs

Les algorithmes de chiffrement par blocs peuvent être utilisés dans quatre modes (dans les exemples les tailles indiquées correspondent au DES) :
– ECB (*Electronic CodeBook*)

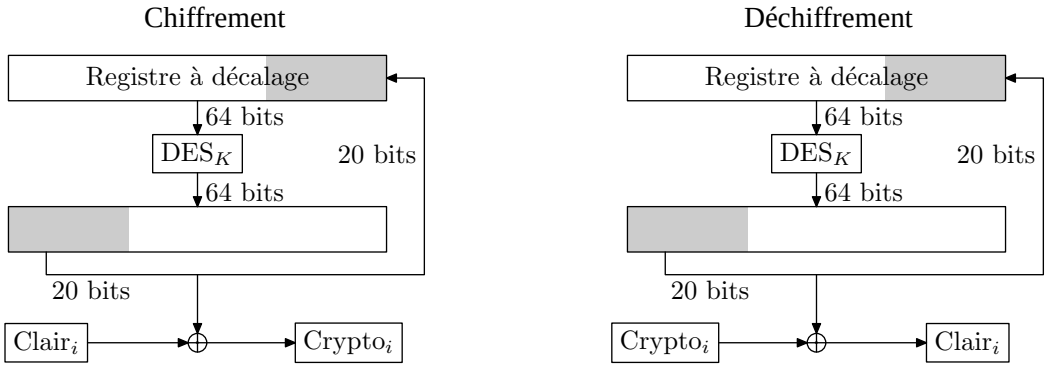
– CBC (*Cypher Block Chaining*)



– CFB (*Cypher Feed Back*) Ce mode permet de chiffrer des blocs de taille inférieure à 64 bits (20 bits sur l'exemple).



– OFB (*Output Feed Back*) Ce mode permet également de chiffrer des blocs de taille inférieure à 64 bits.



3.2 DES (*Data Encryption Standard*)

Le message est découpé en blocs de 64 bits (8 octets). Le chiffrement du bloc se décompose en 16 « tours » ; chaque tour utilise une clef de 48 bits obtenue à partir de la clef principale de 56 bits et différentes transformations dont une est non-linéaire (voir page 18).
L'algorithme est facilement implanté par logiciel et, surtout, par matériel ; la vitesse de chiffrement est élevée.

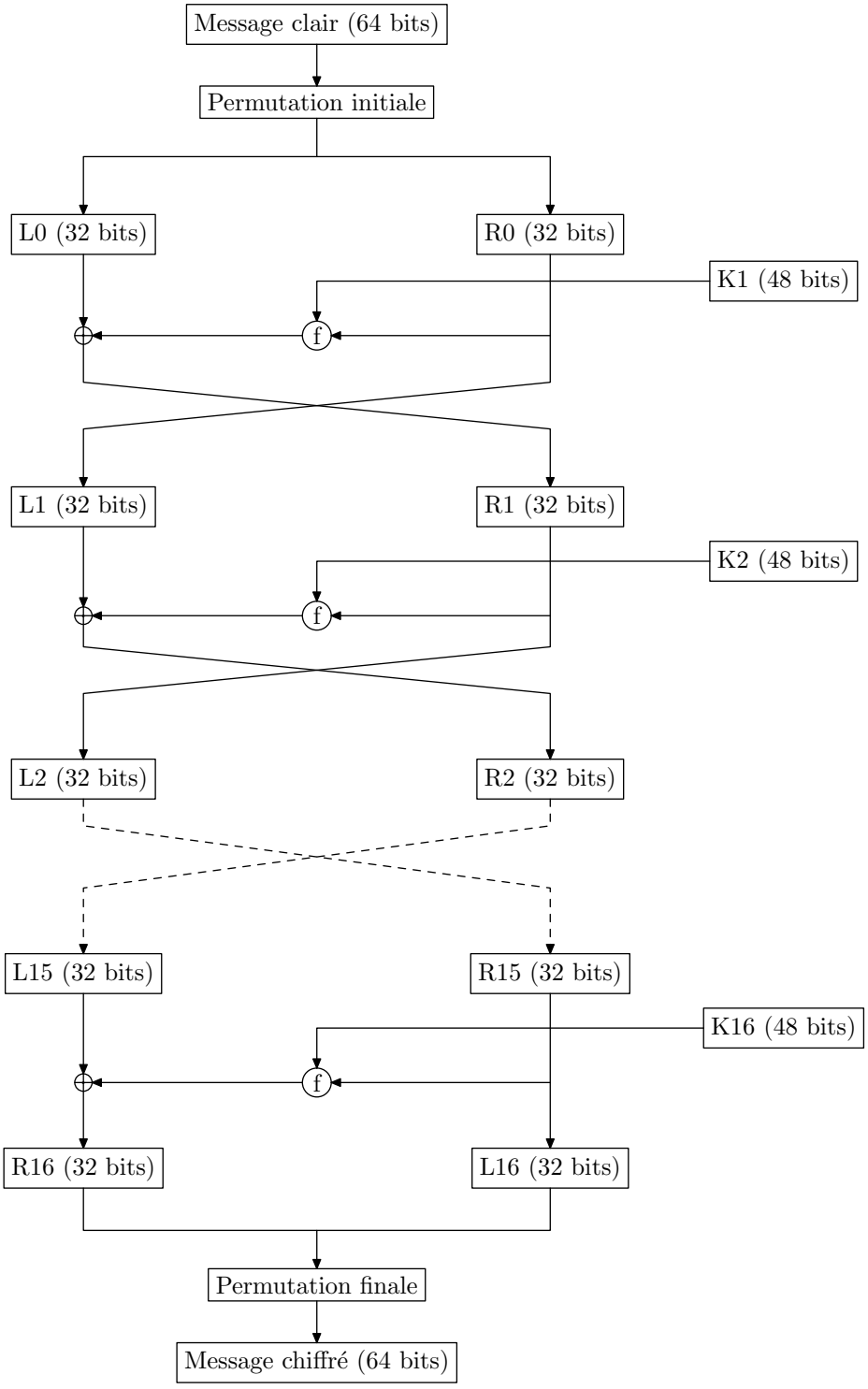


FIG. 1 – Architecture du DES

– Logiciel sur DEC Alpha 4000/610 : 12 Mb/s.
– Matériel (en 1999, composant SNL) : 6,7 Gb/s.

Le déchiffrement est réalisé par le même algorithme que le chiffrement en modifiant le plan de génération des clefs.

La cryptanalyse du DES est étudiée depuis sa création avec des succès relatifs.

– Attaque exhaustive (force brute) : attaque à clair connu, recherche de la clef. En 1999 la clef a été trouvée après avoir testé 22% de l'espace des clefs en 22h15 sur un ordinateur dédié (Deep Crack, plus de 200 000 $) et un réseau de plusieurs milliers d'ordinateurs.
– Cryptanalyse différentielle (1990, BIHAM et SHAMIR) : on examine des paires de cryptos (chiffrés avec la même clef…) dont les clairs présentent des différences particulières fixées. Cette attaque nécessite 2^{47} clairs choisis. Elle est assez théorique…
– Cryptanalyse linéaire (1993, MATSUI) : elle étudie les relations linéaires probables entre certains bits de clair, certains bits de crypto et certains bits de clefs et permet de prédire la valeur de certains bits de clef quand on connaît suffisamment de couples clair-crypto ; on réduit ainsi l'espace de clefs à attaquer par force brute. En 1994 MATSUI a réalisé une attaque à l'aide de 2^{43} couples clair-chiffré sur 12 stations (HP9735/PA RISC 99MHz) en 50 jours.

Il semble donc que le DES peut être considéré comme relativement sûr encore quelque temps. On peut améliorer sa sécurité grâce à des variantes ; la plus utilisée est le triple DES :

$$C = DES_{K3}(DES^{-1}_{K2}(DES_{K1}(M))).$$

La compatibilité est assurée avec le simple DES en prenant $K_1 = K_2 = K_3$. La clef est trois fois plus grande, le temps de chiffrement trois fois plus long.

3.3 AES (*Advanced Encryption Standard*) : Rijndael

Nous donnons de cet algorithme une description succincte[2].

Comme le DES, Rinjdael est une méthode de chiffrement par blocs ; chaque bloc est représenté par une matrice d'octets (« *State* ») ayant 4 lignes et N_b colonnes où N_b est égal à 4, 6 ou 8 suivant que les blocs ont une longueur de 128, 192 ou 256 bits. La clef secrète est également représentée par un matrice d'octets ayant 4 lignes et N_k colonnes où N_k est égal à 4, 6 ou 8.

Le chiffrement d'un bloc est réalisé en N_r « tours » où N_r est égal à 10, 12 ou 14 suivant les valeurs de N_b et N_k.

Chaque tour comprend 4 transformations.

– Substitution non-linéaire sur chaque octet du *State* :
  – inversion dans $GF(2^8)$, corps de GALOIS ayant 256 éléments ;
  – transformation affine dans $GF(2)$.
– Décalage cyclique de chaque ligne.
– Transformation de chaque colonne par multiplication par un polynôme à coefficients dans $GF(2^8)$.
– XOR (ou exclusif) avec une « clef de tour ».

La clef de tour est obtenue à partir de la clef secrète par :

– expansion de la clef secrète,
– extraction de la clef de tour.

Cet algorithme est implémenté efficacement sur des processeurs 8 bits (cartes à puce) ou 32 bits.

[2] Une présentation détaillée est disponible à : http://www.nist.gov/aes/

4 Chiffrement asymétrique

4.1 RSA

Le système cryptographique RSA est basé sur la factorisation d'entier en produit de deux grands facteurs premiers.

4.1.1 Description

Alice crée une clef publique qu'elle diffuse à ses correspondants et une clef privée qu'elle cache soigneusement. Pour cela elle choisit au hasard deux grands nombres premiers distincts p et q ; leur produit $n = p\,q$ a pour indicatrice d'EULER $\varphi(n) = (p-1)(q-1)$ que l'on notera Φ. Elle choisit ensuite au hasard un entier positif e premier avec Φ et détermine l'unique entier d compris entre 2 et $\Phi - 1$ tel que $e\,d \equiv 1 \pmod{\Phi}$. La clef publique d'Alice est (n, e) : elle est transmise à ses correspondants ou à un serveur de clefs. La clef privée est d.
Remarque : p, q, Φ et d ne doivent jamais être communiqués.

Lorsque Bob veut envoyer un message confidentiel à Alice :

1. Il représente le message par un nombre m compris entre 0 et $n-1$; si besoin il découpe auparavant le message en blocs.
2. Il se procure la clef publique (n, e) d'Alice ; il doit s'assurer qu'il s'agit effectivement de la clef publique d'Alice.
3. Il calcule $c = m^e \pmod{n}$ qui est le texte chiffré.
4. Il transmet c à Alice.

Lorsqu'Alice reçoit c elle calcule le texte clair en utilisant sa clef privée d : $m = c^d \pmod{n}$.

4.1.2 Comment ça marche ?

Puisque $e\,d \equiv 1 \pmod{\Phi}$ il existe un entier k tel que $e\,d = 1 + k\,\Phi$.
Si m est premier avec p, d'après le théorème de FERMAT

$$m^{p-1} \equiv 1 \pmod{p}.$$

On élève les deux membres à la puissance $k(q-1)$:

$$m^{k(p-1)(q-1)} \equiv 1 \pmod{p}.$$

En multipliant les deux membres par m on obtient

$$m^{ed} = m^{1+k(p-1)(q-1)} \equiv m \pmod{p}.$$

Si m n'est pas premier avec p alors m est un multiple de p et la congruence précédente est encore valide puisque les deux membres sont congrus à 0 modulo p.
On montre de même

$$m^{ed} \equiv m \pmod{q}.$$

Puisque p et q sont premiers et distincts

$$m^{ed} \equiv m \pmod{n}.$$

Enfin

$$c^d \equiv (m^e)^d \equiv m^{ed} \equiv m \pmod{n}.$$

Remarques
– Puisque e est premier avec $\varphi(n)$ l'application $m \mapsto c$ est une bijection sur $\mathbb{Z}_n$.
– On peut calculer $c^d \pmod{p}$ et $c^d \pmod{q}$ et en déduire $c^d \pmod{pq}$ à l'aide du théorème 8.
– Un raisonnement hâtif se rencontre fréquemment dans la littérature. Puisque $m^{\varphi(n)} \equiv 1 \pmod{n}$ on a $m^{e\,d} = m^{1+k\,\varphi(n)} \equiv m \pmod{n}$... sans se préoccuper de vérifier que m est premier avec n. On peut prendre pour contre-exemple $6^{20} \equiv 12 \pmod{33}$.

L'exponentiation dans $\mathbb{Z}_n^*$ est effectuée grâce à l'algorithme 3 qui se présente sous la forme :

Algorithme 4 (Exponentiation modulaire). *Soient $m \in \mathbb{Z}_n^*$ et $e \in \mathbb{N}$.*
$S \leftarrow m$; $k \leftarrow e$; $A \leftarrow 1$.
while $k \neq 0$ ***do***
  if k *est impair* ***then***
    $A \leftarrow A\,S \pmod{n}$
  end if
  $k \leftarrow \lfloor k/2 \rfloor$; $S \leftarrow S^2 \pmod{n}$.
end while
return(A).

4.1.3 Exemple

Alice choisit $p = 2357$ et $q = 2551$ (en pratique p et q seront beaucoup plus grands) ; d'où $n = p\,q = 6012707$ et $\Phi = (p-1)(q-1) = 6007800$. Elle choisit ensuite $e = 3674911$ et vérifie qu'il est premier avec Φ grâce à l'algorithme d'EUCLIDE (voir page 10) ; cet algorithme fournit également d tel que $d\,e \equiv 1 \pmod{\Phi}$: $d = 422191$, qui constitue la clef privée.
La clef publique est $(n, e) = (6012707, 3674911)$.
Pour chiffrer le message $m = 5234673$, Bob calcule $c = m^e \pmod{n} = 3650502$.
Pour déchiffrer le message c, Alice calcule $c^d \pmod{n} = 5234673$ et retrouve bien m.
À titre d'exercice le lecteur pourra appliquer l'algorithme 4 au calcul de c sur cet exemple et, s'il est très courageux, au déchiffrement pour retrouver m.

4.1.4 Choix des paramètres

Les nombres premiers p et q doivent être de tailles comparables, de l'ordre de 2^{512} c'est-à-dire environ 150 chiffres décimaux au minimum[3]. Cependant $|\,p-q\,|$ ne doit pas être trop petit, sinon p et q sont voisins de $\sqrt{n}$ et peuvent être découverts par recherche exhaustive.
Certains auteurs préconisent de prendre pour p et q des « nombres premiers forts » ; p est un « nombre premier fort » si
– $p-1$ possède un grand facteur premier r,
– $r-1$ possède un grand facteur premier,

[3]Le 22 août 1999 un nombre de 155 chiffres a été factorisé en produit de deux nombres premiers de 78 chiffres à l'aide de 300 ordinateurs fonctionnant pendant trois mois et demi.

– $p+1$ possède un grand facteur premier.
Ce choix permet de mettre en échec certains algorithmes de factorisation ; en outre un tel choix est peu coûteux en temps de calcul (voir [5] § 4.53).

4.1.5 Failles de RSA

Mal utilisé, RSA peut présenter des failles. Donnons deux exemples.

1. Diffusion d'un même message à deux utilisateurs en employant le même module n. On a $c_1 = m^{e_1} \pmod n$ et $c_2 = m^{e_2} \pmod n$. Si e_1 et e_2 sont premiers entre eux (ce qui est très fréquent) on peut trouver m sans connaître les clefs secrètes d_1 et d_2 et sans factoriser n ; on calcule, par l'algorithme d'EUCLIDE étendu (cf. page 10), u et v tels que $u\,e_1 + v\,e_2 = 1$ et on trouve $c_1^u\,c_2^v \equiv m \pmod n$.
2. Diffusion d'un même message à plusieurs utilisateurs en employant un exposant e petit (certaines applications commerciales utilisent $e = 3\ldots$) et commun à tous les destinataires. On a : $\forall i \in \{1, 2, \ldots, k\}, c_i = m^e \pmod{n_i}$. Si les n_i sont premiers entre eux, ce qui est probable, on peut déterminer $m^e \pmod{n_1\,n_2\ldots n_k}$ par le théorème 8. Si $e \leq k$ on calcule m par extraction de racine dans $\mathbb{Z}$.

4.2 EL GAMAL

Le système cryptographique EL GAMAL, publié en 1985, est basé sur le logarithme discret.

4.2.1 Description

Alice choisit au hasard un grand nombre premier p et un générateur α du groupe multiplicatif $\mathbb{Z}_p^\times$; ces deux nombres peuvent être partagés par un groupe d'utilisateurs.
Elle choisit ensuite au hasard un entier a compris entre 1 et $p-2$: ce sera sa clef privée.
Elle calcule $A = \alpha^a \pmod p$. La clef publique d'Alice est (p, α, A) : elle est transmise à ses correspondants ou à un serveur de clefs.
Lorsque Bob veut envoyer un message à Alice :

1. Il le représente par un entier compris entre 0 et $p-1$.
2. Il se procure la clef publique d'Alice et vérifie qu'elle est authentique.
3. Il choisit au hasard un entier k compris entre 1 et $p-2$ et calcule
$$\gamma = \alpha^k \pmod p \qquad \text{et} \qquad \delta = m\,A^k \pmod p.$$
4. Le message chiffré est le couple (γ, δ) ; Bob l'envoie à Alice.

Pour déchiffrer le message (γ, δ) Alice calcule $\gamma^{p-1-a}\,\delta \pmod p$ et retrouve m.
Remarque : le message chiffré a une taille double du message clair.

4.2.2 Comment ça marche ?

L'ordre du groupe $\mathbb{Z}_p^\times$ est égal à $p-1$, donc $\gamma^{p-1} \equiv 1 \pmod p$ et
$$\gamma^{p-1-a}\,\delta \equiv (\alpha^k)^{-a}\,m\,(\alpha^a)^k \equiv m \pmod p.$$

Remarques :

– Le déchiffrement nécessite la connaissance de la clef privée a qui est le logarithme discret de A dans la base α.
– Actuellement on sait calculer le logarithme discret avec un module de 300 bits environ ; pour une sécurité à long terme il faut choisir un module p de 1024 bits au moins.
– Le chiffrement ELGAMAL utilise les mêmes mécanismes que l'échange de clefs de DIFFIE-HELLMAN (voir § 5.1).

4.2.3 Exemple

Alice choisit $p = 2357$, $\alpha = 2$ et $a = 1751$; elle calcule $A = \alpha^a \pmod p = 1185$.
Elle diffuse sa clef publique (p, α, A) et tient secrète sa clef privée a.
Bob veut envoyer à Alice le message $m = 2035$. Il choisit au hasard $k = 1520$ et calcule

$$\gamma = \alpha^k \pmod p = 1430 \qquad \text{et} \qquad \delta = m\,A^k \pmod p = 697.$$

Il transmet à Alice le couple $(\gamma, \delta) = (1430, 697)$.
Alice reçoit le message chiffré (γ, δ) ; elle calcule :

$$\delta\,\gamma^{p-1-a} \pmod p = 697 \times 1430^{2357-1-1751} \pmod{2357} = 2035$$

et retrouve donc le message m.

4.2.4 Variantes

On peut utiliser n'importe quel groupe cyclique à la place de $\mathbb{Z}_p^\times$. Les plus utilisés sont :
– Le groupe multiplicatif $\mathbb{F}_{2^m}^*$ du corps fini $\mathbb{F}_{2^m}$, de caractéristique 2.
– Le groupe des points d'une courbe elliptique sur un corps fini.

4.3 Échange de clefs

L'échange de clef publique semble *a priori* très simple : puisque la clef est publique il suffit de la transmettre à ses correspondants ou de la placer sur un serveur de clefs.
Si l'on peut rencontrer ses correspondants on leur remet une disquette avec la clef (comme au bon vieux temps de la cryptographie symétrique) et le tour est joué.

4.3.1 Attaque « man-in-the-middle »

Mais il est fréquent que l'on doive échanger des messages avec des correspondants que l'on ne rencontre jamais ; on leur transmet alors la clef par le réseau… et les ennuis commencent. Comment Bob peut-il être sûr que la clef qu'il a reçue est bien celle d'Alice ? Peut-être Ève a-t-elle intercepté le message et remplacé la clef d'Alice par une clef qu'elle a fabriquée (ou bien substitué la clef d'Alice sur le serveur de clefs) : elle peut alors déchiffrer les messages de Bob à Alice, puis les chiffrer avec la vraie clef d'Alice et les lui faire parvenir ; elle peut faire la même chose dans l'autre sens et surveiller les échanges d'Alice et Bob sans que ceux-ci s'en doutent.

4.3.2 Empreinte

À la clef publique est associée une empreinte grâce à une fonction de hachage (cf. § 1.2.2). Cette empreinte est communiquée par une voie différente au destinataire de la clef, qui peut ainsi vérifier que la clef est valide. Encore faut-il que Ève ne surveille pas également ce deuxième canal…

4.3.3 Certificat de clef publique

Une autre approche est basée sur les certificats de clef publique. Un tel certificat contient au moins :
– la clef publique ;
– l'identité de son propriétaire : nom, prénom, adresse électronique... et toute information pertinente ;
– la signature (avec sa clef privée, voir encore § 1.2.2) d'une Autorité de Certification (CA : *Certifying Authority*).
La CA peut être un service d'une entreprise, une entreprise spécialisée ou un individu ; il importe seulement que le destinataire de la clef lui fasse confiance.
Le certificat de clef publique garantit seulement que la clef est bien associée à l'identité qui figure sur le certificat.
En pratique les CA d'une entité sont hiérarchisées et forment une Infrastructure de Gestion de Clefs (IGC... ou PKI : *Public Key Infrastructure*) qui fournit un ensemble de services parmi les suivants :
– enregistrement d'un utilisateur,
– génération de certificat,
– distribution de certificat,
– révocation de certificat,
– renouvellement de certificat,
– suspension de certificat,
– archivage de certificat,
– génération de bi-clés,
– recouvrement de clef,
– horodatage.
L'interopérabilité des certificats issus de différentes CA est garantie par la norme X509.

5 Quelques protocoles

5.1 DIFFIE-HELLMAN : échange public de clefs secrètes

Basé sur le logarithme discret ce protocole permet à plusieurs correspondants de choisir une clef secrète par des échanges sur un canal peu sûr.

5.1.1 Échange entre deux correspondants

Soit p un grand nombre premier et g un générateur du groupe multiplicatif $\mathbb{Z}_p^\times$; ces deux nombres peuvent être choisis une fois pour toutes et partagés par un ensemble de correspondants.
Alice choisit au hasard un entier a compris entre 1 et $p-2$ et calcule $A = g^a \pmod p$ qu'elle communique à Bob.
Bob, symétriquement, choisit au hasard un entier b compris entre 1 et $p-2$ et calcule $B = g^b \pmod p$ qu'il communique à Alice.
Alice et Bob calculent la clef secrète $K = g^{ab} \pmod p = A^b \pmod p = B^a \pmod p$.
Ève, qui écoute la conversation, connaît p, g, A et B, mais ne peut en déduire K ; il faudrait qu'elle puisse calculer a ou b, c'est-à-dire un logarithme discret.
Martin, attaquant actif, peut se faire passer pour Alice auprès de Bob et pour Bob auprès d'Alice. Pour déjouer cette attaque Alice et Bob peuvent signer respectivement A et B avec leurs clefs privées.

Exemple : on fixe $p = 2357$ et $g = 2$.
Alice choisit $a = 1751$ et calcule $A = g^a \pmod p = 1185$ qu'elle communique à Bob.
Bob choisit $b = 1925$ et calcule $B = g^b \pmod p = 794$ qu'il transmet à Alice.
Alice calcule la clef secrète $K = B^a \pmod p = 1042$; Bob calcule $A^b \pmod p = 1042$ et trouve la même valeur : ils partagent une clef secrète.
Ève connaît $p = 2357$, $g = 2$, $A = 1185$ et $B = 794$; comment peut-elle déterminer K ? On ne connaît actuellement aucun moyen de le faire sans calculer a ou b.
Remarques :
– Pour augmenter la sécurité il est conseillé de choisir p assez grand (au moins 1024 bits) et tel que $(p-1)/2$ soit également premier.
– Le chiffrement ELGAMAL (voir § 4.2) consiste à définir une clef de session $A^k = \alpha^{ak}$ et à chiffrer le message en le multipliant par cette clef.

5.1.2 Échange entre trois correspondants et plus

Le module p et le générateur g étant fixés,
– Alice choisit une clef secrète a et calcule $A = g^a \pmod p$ qu'elle transmet à Bob ;
– Bob choisit une clef secrète b et calcule $B = g^b \pmod p$ qu'il transmet à Charles ;
– Charles choisit une clef secrète c et calcule $C = g^c \pmod p$ qu'il transmet à Alice ;
– Alice calcule $C' = C^a \pmod p$ qu'elle transmet à Bob ;
– Bob calcule $A' = A^b \pmod p$ qu'il transmet à Charles ;
– Charles calcule $B' = B^c \pmod p$ qu'il transmet à Alice ;
– Alice calcule la clef $K = B'^a \pmod p = g^{abc} \pmod p$;
– Bob calcule la clef $K = C'^b \pmod p = g^{abc} \pmod p$;
– Charles calcule la clef $K = A'^c \pmod p = g^{abc} \pmod p$.
Ève connaît p, g, A, B, C, A', B' et C', mais elle ne peut pas déterminer K ; frustrant, n'est-ce pas ?
Exercice : généraliser le protocole à quatre personnes et plus.

5.2 Authentification

5.2.1 Défi-réponse

En chiffrement symétrique une application importante est l'identification des avions militaires (IFF : *Identification Friend or Foe*). L'avion et le contrôle au sol partagent une clef secrète K. Le contrôle au sol envoie à l'avion un nombre aléatoire x et l'avion répond par $\mathcal{C}(K, x)$, prouvant ainsi qu'il connaît K.

5.2.2 Identification à divulgation nulle

Le chiffrement asymétrique permet l'identification sans partage de secret et sans communiquer d'information (*zero-knowledge*).

FEIGE-FIAT-SHAMIR La sécurité de ce protocole est basée sur la difficulté de l'extraction de racine carrée dans $\mathbb{Z}_n$ lorsque n est composite.
Patricia, le prouveur, veut prouver son identité à Victor, le vérifieur, *sans lui communiquer d'information*. Elle publie dans un annuaire sa clef publique composée de :
– $n = p\,q$, où p et q sont deux (grands) nombres premiers ;

– $v = s^2 \pmod n$, où s est compris entre 1 et $n-1$ et premier avec n.
Remarque : p, q et s doivent rester secrets.
Patricia prouve à Victor sa connaissance de s sans le lui communiquer en répétant t fois les échanges suivants :
– Patricia choisit au hasard un nombre r compris entre 1 et $n-1$ et envoie à Victor $x = r^2 \pmod n$;
– Victor envoie à Patricia un bit *aléatoire* b ;
– Patricia envoie à Victor
  – $y = r$ si $b = 0$,
  – $y = r\,s \pmod n$ si $b = 1$.
À chaque échange Victor vérifie que
– $y^2 = x \pmod n$ si $b = 0$,
– $y^2 = x\,v \pmod n$ si $b = 1$.
Martin, qui tente de se faire passer pour Patricia sans connaître s, peut envoyer r quel que soit b en ayant choisi
– $x = r^2$ de façon à tromper Victor s'il envoie $b = 0$ ou
– $x = r^2/v$ de façon à tromper Victor s'il envoie $b = 1$,
mais ces choix sont exclusifs : il a donc une chance sur deux d'être démasqué.
Après t échanges sans erreur la probabilité pour que le prouveur ne connaisse pas s est donc égale à $(1/2)^t$.

SCHNORR Ce protocole est basé sur le logarithme discret.
Paramètres publics (ils peuvent être communs à un groupe d'utilisateurs) :
– p un grand nombre premier (1024 bits environ),
– q un facteur premier de $p-1$ (de 160 bits au moins),
– $g \in \{1, \ldots, p-1\}$ tel que $g^q \equiv 1 \pmod p$,
– $t \geq 40$ tel que $2^t < q$.
Bi-clef de Patricia :
– clef privée : $s \in \{0, \ldots, q-1\}$, choisie au hasard ;
– clef publique : $v = g^{-s} \pmod p$.
Échanges :
– Patricia choisit au hasard $r \in \{1, \ldots, q-1\}$ et calcule $x = g^r \pmod p$; ce calcul peut être réalisé à l'avance.
  Patricia envoie x à Victor.
– Victor envoie à Patricia un nombre e choisi au hasard entre 1 et 2^t.
– Patricia calcule $y = r + s\,e \pmod q$ et envoie y à Victor.
– Victor vérifie que $g^y\,v^e \pmod p$ est égal à x.
Martin peut deviner la valeur de e avec une probabilité égale à 2^{-t} ; il choisit alors $y < q$ et envoie $x = g^y\,v^e \pmod p$, puis y.
Ce protocole présente les avantages suivants sur celui de FEIGE-FIAT-SHAMIR :
– le calcul de x est possible en temps différé ;
– le calcul de y modulo q est plus simple que modulo p ;
– le nombre de bits à transmettre est plus faible.
En outre la charge de calcul est plus importante pour le vérifieur que pour le prouveur : ce protocole est mieux adapté à une transaction entre une carte à puce et un guichet électronique.

5.3 SHA : *Secure Hash Algorithm*

SHA, seul algorithme de hachage sûr actuellement, est utilisé dans l'algorithme de signature DSA (cf. § 5.4) ; tous les deux sont des standards américains (SHS et DSS).

Description de l'algorithme :

1. Le message est complété pour que sa longueur soit un multiple de 512 bits ; pour cela on ajoute un bit à 1, puis un certain nombre de bits à 0 et un entier sur 64 bits représentant la longueur du message avant remplissage.
2. On initialise les registres A, B, C, D et E avec des constantes fixées :
   - $A =$`0x67452301`,
   - $B =$`0xEFCDAB89`,
   - $C =$`0x98BADCFE`,
   - $D =$`0x10325476`,
   - $E =$`0xC3D2E1F0`.
3. Pour chaque bloc de 512 bits du message
   - on copie A, B, C, D et E dans AA, BB, CC, DD et EE respectivement ;
   - on applique la fonction de compression (voir ci-dessous) à AA, BB, CC, DD, EE pour le bloc courant ;
   - on ajoute les valeurs trouvées à A, B, C, D et E respectivement.
4. Le résultat est la concaténation de A, B, C, D et E.

Description de la fonction de compression :

1. Les 512 bits du bloc courant sont scindés en 16 mots de 32 bits $\langle W^{(0)}, \ldots, W^{(15)} \rangle$.
2. On réalise une expansion par

$$\forall i \in \{16, \ldots, 79\},\ W^{(i)} = ROL_1\big(W^{(i-3)} \oplus W^{(i-8)} \oplus W^{(i-14)} \oplus W^{(i-16)}\big)$$

   où ROL_k représente un décalage circulaire de k bits à gauche et $\oplus$ le *ou exclusif*.
3. Ces 80 mots de 32 bits sont utilisés pour modifier les 5 variables d'état $A^{(i)}$, $B^{(i)}$, $C^{(i)}$, $D^{(i)}$ et $E^{(i)}$:
   - on initialise : $\langle A^{(0)}, B^{(0)}, C^{(0)}, D^{(0)}, E^{(0)} \rangle = \langle AA, BB, CC, DD, EE \rangle$;
   - pour i variant de 0 à 79

$$\begin{aligned} A^{(i+1)} &= ADD\Big(W^{(i)}, ROL_5(A^{(i)}), f^{(i)}\big(B^{(i)}, C^{(i)}, D^{(i)}\big), E^{(i)}, K^{(i)}\Big), \\ B^{(i+1)} &= A^{(i)}, \\ C^{(i+1)} &= ROL_{30}(B^{(i)}), \\ D^{(i+1)} &= C^{(i)}, \\ E^{(i+1)} &= D^{(i)}, \end{aligned}$$

   où
   - ADD représente l'addition modulo 2^{32},
   - les $K^{(i)}$, constantes fixées, et les fonctions $f^{(i)}$ sont définies par

i	$f^{(i)}$	$K^{(i)}$
0-19	$(X \wedge Y) \vee (X \wedge Z)$	`0x5A827999`
20-39	$X \oplus Y \oplus Z$	`0x6ED9EBA1`
40-59	$(X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$	`0x8F1BBCDC`
60-79	$X \oplus Y \oplus Z$	`0xCA62C1D6`

où $\vee$ représente le *ou* (inclusif) et $\wedge$ le *et*.

L'architecture de SHA est résumée dans la figure ci-dessous.

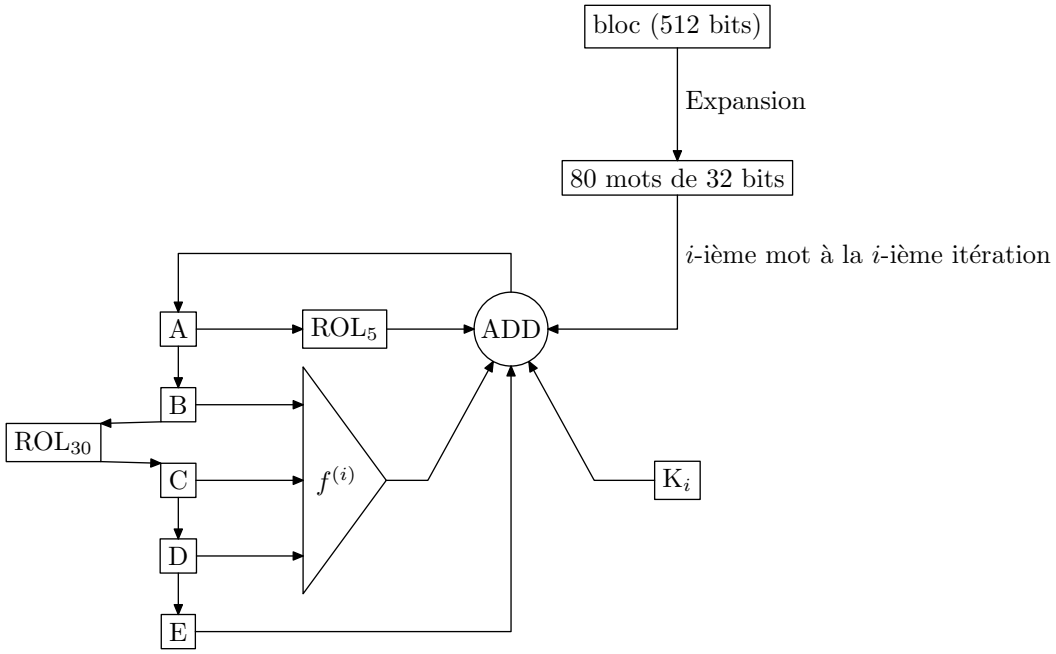


FIG. 2 – Architecture de SHA

5.4 DSA : *Digital Signature Algorithm*

Le DSA a été proposé en 1991 par le NIST comme standard : DSS (*Digital Signature Standard*). Sa sécurité est basée sur la difficulté du calcul du logarithme discret.

5.4.1 Génération des clefs

1. Choisir un nombre premier p dont la longueur en bits, multiple de 64, est comprise entre 512 et 1024.
2. Choisir un nombre premier q long de 160 bits qui divise $p - 1$.
3. Choisir un générateur g du sous-groupe cyclique d'ordre q de $\mathbb{Z}_p^*$: on choisit pour cela $h \in \mathbb{Z}_p^*$ et on calcule $h^{(p-1)/q} \pmod p$ jusqu'à ce qu'on obtienne un résultat différent de 1.
4. Choisir x compris entre 1 et $q - 1$.

5. Calculer $y = g^x \pmod p$.

p, q et g sont publics et peuvent être partagés par un groupe d'utilisateurs.
x est la clef privée d'Alice, y est sa clef publique.

5.4.2 Génération de signature

Pour signer le message m Alice

1. choisit au hasard un nombre k entre 1 et $q-1$,
2. calcule $r = (g^k \pmod p) \pmod q$,
3. calcule $k^{-1} \pmod q$,
4. calcule $s = k^{-1}\big(h(m) + x\,r\big) \pmod q$, où h est la fonction de hachage du SHA.

Si r ou s est nul Alice recommence (avec une autre valeur de k).
La signature d'Alice pour le message m est le couple (r, s).

5.4.3 Vérification de signature

Pour vérifier la signature d'Alice pour le message m Bob

1. se procure les clefs p, q, g et y,
2. calcule $w = s^{-1} \pmod q$,
3. calcule $u_1 = w\,h(m) \pmod q$,
4. calcule $u_2 = r\,w \pmod q$,
5. calcule $v = \big(g^{u_1}\,y^{u_2} \pmod p\big) \pmod q$,
6. accepte la signature si $v = r$.

5.4.4 Pourquoi ça marche ?

D'après la définition de s on a : $h(m) \equiv k\,s - x\,r \pmod q$. Multipliant les deux membres par w, qui est l'inverse de s : $w\,h(m) + x\,r\,w \equiv k \pmod q$, c'est-à-dire $u_1 + x\,u_2 \equiv k \pmod q$. D'où : $g^{u_1+xu_2} = g^{k+\lambda q}$ et, puisque $g^x \equiv y \pmod p$ et $g^q \equiv 1 \pmod q$, $\big(g^{u_1}\,y^{u_2} \pmod p\big) \pmod q = \big(g^k \pmod p\big) \pmod q$.

5.5 SSL : *Secure Socket Layer*

Le protocole SSL a été développé par Netscape Communication Inc. pour améliorer de façon transparente la confidentialité des échanges de tout protocole basé sur TCP/IP : HTTP, SMTP, IMAP...La version 2 a été diffusée en 1994 ; la version 3, proposée en 1996, corrige quelques faiblesses et apporte de nouvelles fonctionnalités.

<table>
<tr><td>SSL Handshake Protocol</td><td colspan="2">SSL Change Cipher Specs
version 3 uniquement</td><td>HTTP</td><td>Autres protocoles</td></tr>
<tr><td colspan="5">SSL Record Protocol</td></tr>
<tr><td colspan="5">TCP</td></tr>
</table>

Ses principales caractéristiques sont :
– confidentialité : chiffrement des échanges ;
– intégrité : chaque fragment est accompagné de son empreinte MAC (*Message Authentication Code*) ;
– authentification du serveur et/ou du client : utilisation de certificats X.509 ;
– rapidité : usage de clefs de session ;
– extensibilité, interopérabilité : négociation des algorithmes de chiffrement ;
– résistance aux attaques usuelles.

5.5.1 SSL *Record protocol*

Le flot de données est fragmenté en unités de 16 K octets au plus. Une empreinte (MAC) est ajoutée, puis des caractères de bourrage pour obtenir une longueur multiple de la taille des blocs de chiffrement. Le résultat est chiffré avec la clef de session. Un en-tête de 2 ou 3 octets indique la longueur des données et du bourrage.
Les algorithmes utilisés pour le chiffrement et le calcul du MAC, ainsi que la clef de chiffrement, sont négociés dans la phase de connexion (*Handshake Protocol*).
Le calcul du MAC inclut, entre autres, le numéro du fragment et une clef secrète : il est impossible de supprimer ou d'ajouter un fragment frauduleusement.

5.5.2 SSL *Handshake protocol*

Pendant la phase de connexion il permet de
– déterminer les spécifications de chiffrement, c'est-à-dire les algorithmes utilisés pour le chiffrement des données et le calcul du MAC ; toute solution est acceptable si elle est connue du client et du serveur ;
– échanger et vérifier les certificats du serveur et/ou du client ;
– générer une clef-maître de session et d'en dériver une clef de chiffrement pour le serveur, une clef de chiffrement pour le client et des clefs pour le calcul des MAC.

Il s'appuie sur le SSL *Record Protocol* ; les échanges sont sécurisés dès que le choix des spécifications de chiffrement est effectué.
Les principales étapes sont résumées dans le tableau suivant :

Séquence	Sens	Informations échangées
`client-hello`	C → S	défi_1, spécifications de chiffrement supportées
`server-hello`	C ← S	Id_connexion, spécifications de chiffrement supportées, certificat du serveur, chaîne de CA
`client-master-key`	C → S	clef-maître de session chiffrée avec la clef publique du serveur
`client-finish`	C → S	Id_connexion chiffrée avec la clef client
`server-verify`	C ← S	défi_1 chiffré avec la clef du serveur
`request-certificate` (facultatif)	C ← S	(types de certificats acceptables, liste de CA autorisés, défi_2) chiffré avec la clef du serveur
`client-certificate` (facultatif)	C → S	(certificat du client, chaîne de CA) chiffré avec la clef du client
`server-finish`	C ← S	id-session chiffrée avec la clef du serveur

Dans la séquence `client-hello` le défi_1 est un nombre aléatoire de 256 bits utilisé dans la séquence `server-verify` pour authentifier le serveur.
Dans la séquence `server-hello` Id_connexion est un nombre aléatoire de 256 bits utilisé par le client pour terminer la phase d'initialisation dans la séquence `client-finish`; la liste des spécifications de chiffrement supportées par le client et le serveur permet au client de sélectionner la spécification de chiffrement qui sera utilisée.
Le client génère une clef-maître de session de 48 octets qu'il transmet au serveur dans la séquence `client-master-key` après l'avoir chiffrée avec la clef publique du serveur. À partir de cette clef-maître le client et le serveur calculent chacun une clef secrète de chiffrement qu'ils utilisent pour tous les échanges qui suivent.
Les séquences `request-certificate` et `client-certificate` sont facultatives : l'authentification du client est optionnelle (celle du serveur est obligatoire).
L'id_session transmise au client dans la séquence `server-finish` est associée à la clef-maître : dans les connexions ultérieures entre le client et le serveur on pourra supprimer les transferts de la clef publique du serveur et de la clef-maître si le serveur a conservé l'id_session et la clef-maître dans son cache.

5.6 Comparaison des systèmes symétriques et asymétriques

		Symétrique	Asymétrique
Confidentialité		Rapide ($>$ 100 Mbits/s)	Lent ($<$ 100 Kbits/s)
Authentification	Défi-réponse	Rapide	
	0-knowledge	Impossible	Possible
Signature		Impossible	Rapide (avec hash)
Échange public de clef secrète		Impossible	Simple
Gestion des clefs		Centralisée, lourde ($O(n^2)$) Manuelle Protection physique globale Compromission locale $\rightarrow$ globale	Décentralisée, annuaire Automatique Protection physique locale Compromission locale $\leftrightarrow$ locale
Réseaux		Cloisonnés	Ouverts

6 Sécurité des cartes à puce

Le micro-processeur des cartes à puce possède trois mémoires :

1. ROM (6 à 20 Ko) : codage du DES et numéro,
2. RAM (128 à 1024 octets) : calculs,
3. EEPROM (1, 4 ou 8 Ko) : personnalisation (clefs,...).

La sécurité d'une carte bancaire est assurée à trois niveaux :

1. PIN code : ce code à 4 ou 5 chiffres (*Personal Identification Number*) permet de s'identifier auprès de la carte ; il est stocké, en clair, dans une zone secrète de l'EEPROM. Quand on entre le PIN code, le micro-processeur le vérifie et retourne *oui* ou *non*.
2. La signature RSA (statique), avec une clef de 320 bits, du numéro de carte ($S = (Id|Id)^d \pmod n$) est stockée sur la piste magnétique et dans une zone publique de l'EEPROM. Le terminal vérifie que $S^3 \pmod n$ donne deux fois Id.
3. La clef secrète pour le DES, k, est calculée à partir du numéro de la carte et d'une clef maître K : $k = DES_K(Id)$. Elle est stockée dans une zone secrète de l'EEPROM.

   L'authentification est réalisée par un défi-réponse : le terminal envoie un challenge x et la carte retourne $DES_k(x)$. Le terminal doit connaître K pour effectuer la vérification ; sinon il la sous-traite *on-line* à un serveur.

   Le *Message Authentication Code* (MAC), condensé par une fonction de hachage, de chaque transaction est enregistré dans l'EEPROM.

Avec l'augmentation de la puissance et de la taille de la RAM on voit apparaître des cartes avec cryptographie asymétrique embarquée (RSA 512, 1024...).

7 Législation

La politique du gouvernement français a longtemps été extrêmement restrictive en matière de cryptographie, considérant les outils cryptographiques comme des armes de guerre. La situation a évolué

très rapidement avec le développement du commerce électronique qui nécessite la sécurisation des échanges. La libéralisation du chiffrement a été décidée par le Comité Interministériel pour la Société de l'Information (CISI) le 19 janvier 1999 : décrets n° 99-199[4] et n° 99-200[5] du 17 mars 1999.
Le nouveau cadre législatif peut être synthétisé par les deux tableaux suivants.

Opérations	**Fonctions offertes**				
		Confidentialité			
	Authentif. Signature Intégrité	$\leq$ 40 bits	entre 40 et 128 bits	$>$ 128 bits « séquestré »	$>$ 128 bits « non séquestré »
Utilisation	Libre	Libre	Libre[a]	Libre	Soumise à autorisation[b]
Fourniture	Soumise à déclaration simplifiée	Soumise à déclaration	Soumise à déclaration	Soumise à autorisation	Soumise à autorisation
Importation[c]	Libre	Libre	Libre	Soumise à autorisation	Soumise à autorisation

[a] À condition, soit que les matériels ou logiciels aient préalablement fait l'objet d'une déclaration par leur producteur, un fournisseur ou un importateur, soit que lesdits matériels ou logiciels soient exclusivement destinés à l'usage privé d'une personne physique.
[b] Autorisée si les matériels ou logiciels font l'objet d'une autorisation de fourniture en vue d'une utilisation générale.
[c] En provenance d'un État n'appartenant pas à la Communauté Européenne ou n'étant pas partie à l'accord instituant l'Espace économique européen.

Opérations	**Fonctions offertes**			
		Confidentialité		
	Authentification Signature Intégrité	$>$ 40 bits et $\leq$ 56 bits	$>$ 56 bits et $\leq$ 64 bits	$>$ 64 bits
Exportation	Libre	Libre	Soumise à autorisation[a]	Soumise à autorisation[b]

[a] Libre si les matériels ou logiciels remplissent toutes les conditions fixées par la Note cryptographique de la décision 94/942/PESC modifiée.
[b] Libre en transfert intracommunautaire si les matériels ou logiciels remplissent toutes les conditions fixées par le point 5 de l'annexe IV, catégorie 5 de la décision 94/942/PESC modifiée.

La loi portant sur « Droit de la preuve et signature électronique » a été adoptée le 13 mars 2000 ; le décret n° 2001-272 a été publié le 30 mars 2001.[6]
D'autre part un projet de loi sur la « Société de l'Information » (LSI) est en cours de finalisation.[7]

[4] http://www.internet.gouv.fr/francais/textesref/cryptodecret99199.htm
[5] http://www.internet.gouv.fr/francais/textesref/cryptodecret99200.htm
[6] Ces documents peuvent être consultés à : http://www.legifrance.gouv.fr/
[7] Voir
– http://www.internet.gouv.fr/francais/textesref/pagsi2/lsi.htm
– http://www.finances.gouv.fr/societe_information/sommaire.htm

8 PGP

8.1 Contexte

« Pretty Good Privacy » a un statut assez particulier parmi les logiciels de cryptographie. Son créateur, Phil ZIMMERMANN, est un militant des droits civiques qui, après des études de physique et d'informatique et quelques années d'expérience professionnelle, a programmé la première version de PGP et mis son programme sur le réseau : il considère que l'utilisation d'un outil de chiffrement efficace fait partie des droits du citoyen. Il a été l'objet de poursuites car il ne respectait pas des brevets déposés sur RSA et que l'administration fédérale n'appréciait pas que n'importe qui puisse chiffrer ses communications. D'autre part l'exportation d'outils de chiffrement hors des USA était interdite.
PGP est régulièrement soupçonné de comporter une « trappe » permettant (à la NSA ?) de déchiffrer sans peine les messages chiffrés. Le code source de PGP est disponible et chacun peut le vérifier (il faut du temps et une certaine compétence, mais un certain nombre de programmeurs l'ont fait) : ces soupçons sont sans doute vains.
Actuellement Network Associates Inc. commercialise PGP ; une version « libre » (freeware) est également proposée pour une utilisation privée.

8.2 Fonctionnement

Jusqu'à la version 5 le chiffrement asymétrique et la signature étaient assurés par RSA ; à partir de la version 5 le chiffrement asymétrique utilise ELGAMAL (cf. § 4.2) et la signature DSA (cf. § 5.4). La longueur de la clef peut être choisie entre 768 à 4096 bits.
Le chiffrement asymétrique sert uniquement à chiffrer une clé de session aléatoire qui est elle-même utilisée pour le chiffrement symétrique des données. Celles-ci sont préalablement compressées dans un double but :
– gagner de la place,
– rendre la cryptanalyse plus difficile.
Le chiffrement symétrique se fait en mode CFB ; on a le choix entre trois algorithmes (cf. § 3.2) :
– CAST avec une clef de 128 bits,
– IDEA avec une clef de 128 bits,
– Triple-DES avec une clef de 168 bits (interdit en France, voir § 7).

8.3 Distribution des clefs

PGP permet de vérifier une clef publique par son empreinte (cf. § 4.3.2) ou par un certificat (cf. § 4.3.3). Lorsque Bob est sûr que la clef d'Alice est valide il la signe, la signature pouvant être exportée ; dans ce cas Charles validera la clef d'Alice…s'il fait confiance à Bob. À chaque clef publique, plus précisément à son propriétaire, PGP associe un niveau de confiance pour la validation d'autres clefs publiques : on peut être sûr de la validité de la clef sans faire confiance à son propriétaire pour autant.
Les (certificats de) clefs publiques peuvent être déposé(e)s par leur propriétaire sur des serveurs de clefs ; elles doivent être signées par un certain nombre d'utilisateurs de PGP en qui les correspondants ont confiance. Contrairement au modèle hiérarchique avec Autorité de Certification on utilise un modèle distribué (*Web of trust*).
Il est possible à une organisation de se doter d'une Autorité de Certification délivrant des « certificats » PGP, c'est-à-dire des clefs publiques revêtues de sa signature.

9 Quelques pointeurs

9.1 Sites Web

`http://www.internet.gouv.fr`
Le site officiel du gouvernement français concernant la société de l'information.

`http://www.ssi.gouv.fr`
La documentation officielle sur la sécurité informatique et la cryptographie en France.

`http://www.cacr.math.uwaterloo.ca/hac`
Site où l'on peut télécharger [5]

`http://www.nai.com`
Le site de Network Associates Inc., distributeur de PGP.

`http://www.pgpi.org`
Un site proposant le téléchargement de la version internationale de PGP.

9.2 Newsgroups

`fr.misc.cryptologie`
En français. Verbeux.

`sci.crypt`
Généraliste.

`sci.crypt.research`
Pour suivre les derniers développements.

`comp.security.pgp.tech`

`comp.security.pgp.announce`

Références

[1] Gilles BRASSARD. *Cryptologie contemporaine*. Masson, 1993.

[2] Jean-Paul DELAHAYE. *Merveilleux nombres premiers – Voyage au cœur de l'arithmétique*. Belin, 2000.

[3] Gilles DUBERTRET. *Initiation à la cryptographie*. Vuibert, 1999.

[4] David KAHN. *La guerre des codes secrets*. InterEditions, 1980.

[5] A. MENEZES, P. Van OORSCHOT, et S. VANSTONE. *Handbook of applied cryptography*. CRC Press, 1996.

[6] Bruce SCHNEIER. *Cryptographie appliquée*. International Thomson Publishing France, Paris, 1995.

[7] Simon SINGH. *Histoire des codes secrets*. JC Lattès, 1999.

[8] Jacques STERN. *La science du secret*. Éditions Odile Jacob, 1998.

[9] Jacques VÉLU. *Méthodes mathématiques pour l'informatique*. Dunod, 1999.

[10] André WARUSFEL. *Structures algébriques finies*. Hachette, 1971.

[11] Gilles ZÉMOR. *Cours de cryptographie*. Cassini, Paris, 2000.