Liste des figures :

GLOSSAIRE :

RSA : Rivest-Shamir-Adleman (initiales des trois inventeurs de l'algorithme)
p : un nombre premier différent de q
q : un nombre premier différent de p
n : module de chiffrement
e : exposant de chiffrement
EdDSA : Edwards-curve Digital Signature Algorithm (Algorithme de Signature Courbe d'Edwards)
DSA : Digital Signature Algorithm (Algorithme de signature numérique standardisé par le NIST aux Etats-Unis du temps où RSA était breveté)
ECDSA : Elliptic Curve Digital Signature Algorithm (algorithme de signature numérique à clé publique, variante de DSA. Il fait appel à la cryptographie sur les courbes elliptiques)
NIST : National Institute of Standards and Technology
SMS : Short Message Service
IND-COA : INDistinguishability under Ciphertext-Only Attack
IND-KPA : INDistinguishability under Known-Plaintext Attack
IND-CPA : INDistinguishability under Chosen-Plaintext Attack
IND-CCA : INDistinguishability under Chosen-Ciphertext Attack
DES : Data Encryption Standard
AES : Advanced Encryption Standard
IDEA : International Data Encryption Algorithm
MD5 : Message Digest 5
Attaque XSL : eXtended Sparse Linearization (méthode heuristique de cryptanalyse contre les chiffrements par bloc)
ECB : Electronic Code Book
RC4 : Rivest Cipher 4
PKCS : Public Key Cryptography Standards
RC6 : Rivest Cipher 6 (dérivé de RC5)
IBM : International Businesses Machines Corporation
x : Message clair
y : Message chiffré
E_k () : Chiffrement
D_k () : Déchiffrement
NSA : National Security Agency
MIT : Massachusetts Institute of Technology
Padding : rembourrage – remplissage = inclut l'ajout de données au début au milieu ou à la fin d'un message avant le chiffrement

Table des Matières

INTRODUCTION GENERALE :

« Montrez aux gens les problèmes, puis montrez-leur les solutions ils seront incités à agir » Bill GATES.

Le développement fulgurant des réseaux informatiques, privés ou publics, provoque une masse de plus en plus importante de données sauvegardées et transmises occasionnant ainsi de nouvelles exigences en matière de sécurité. Et une société, où la dépendance à un système informatique est considérable, la sécurité devient une préoccupation cruciale.

Dans ce contexte, la cryptographie est un apport vital de la sécurité informatique moderne. Cependant, tout utilisateur non-éclairé peut ne pas discerner son intérêt. L'objectif principal de ce mémoire est d'introduire différentes notions d'attaques, d'étudier et de mettre en œuvre une attaque sur le crypto système RSA.

En général, la cryptographie est définie comme une technique de protection des informations basée sur les mathématiques pour assurer des services de sécurité, l'intégrité, l'authentification, la confidentialité et la non-répudiation. Quant à la cryptanalyse, c'est l'ensemble des techniques cherchant à percer le secret des informations sans connaitre les clés utilisées. Ces deux branches sont regroupées sous le terme générique de cryptologie qui veut dire « science du secret »

Chronologiquement, la cryptographie et la cryptanalyse ont toujours été adoptées et analysées pour et par les armées afin de garantir la confidentialité des transmissions notamment radioélectriques et d'amasser des informations sensibles sur ses adversaires. Actuellement, le développement important des infrastructures numériques pour l'échange des informations, le stockage des données font de la cryptologie l'un des moyens les plus sûrs pour assurer la sécurité des informations. De nos jours, les algorithmes cryptographiques se spécifient selon les contextes de leur emploi.

La cryptographie se subdivise en deux grandes familles : la cryptographie à clé secrète ou symétrique qui exige que les deux parties utilisent une clé secrète pour communiquer entre elles et la cryptographie à clé publique ou asymétrique qui requiert une paire de clés. Dans ce dernier cas, chacune des parties bénéficie d'un couple de clés dont l'une des clés n'est connue que de son seul utilisateur (clé privée) et tandis que l'autre est rendue publique (clé publique). Son organisation se rapproche à celle d'un annuaire : si nous considérons la clé publique comme étant le numéro d'un utilisateur, il est possible de lui communiquer des informations qu'il ne pourra retrouver que s'il détient la bonne ligne téléphonique c'est-à-dire la clé privée associée. Ces deux types de science présentent aussi des questionnements aux aspects très différents.

En général, les primitives fournies par la cryptographie à clé secrète sont plus performantes en termes d'exécution d'opérations et de stockage que celles fournies par la cryptographie à clé publique. Cette dernière répond à un besoin majeur de la cryptographie symétrique : le partage sécurisé de la clef entre deux correspondants, afin d'éviter l'interception de cette clef par une personne tierce non autorisée et donc la lecture des données chiffrées sans autorisation. Cependant, la cryptographie symétrique pose des inconvénients parce que la distribution des clés est très ardue.

Parmi les cryptosystèmes, le RSA représente sans conteste le système à clef publique le plus utilisé de nos jours puisqu'il est parvenu à survivre aux critiques des spécialistes ainsi qu'à des attaques pendant plus d'un quart de siècle. La fiabilité de ce cryptosystème mondialement

connu repose notamment sur la difficulté mathématique de décomposer en produit de facteurs premiers, ou de factoriser, de grands nombres. Mais les menaces intentionnelles (attaque passive et active) sont de plus en plus considérables. Ainsi, il faut se mettre à la place de l'attaquant pour mieux anticiper les agressions. Et la technique qui permet d'attaquer et de déduire un texte clair d'un texte chiffré est appelée cryptanalyse qui est aussi une branche de la cryptologie.

Ce mémoire se présente comme suit :

Dans la partie I, nous commençons par définir des objectifs, des hypothèses, et des résultats.

Nous rappelons **dans la partie II** quelques notions de la cryptologie en citant les disciplines qui la composent comme la cryptographie et la cryptanalyse, nous présenterons aussi l'algorithme RSA, et exposerons les techniques d'attaques réalisées sur ce cryptosystème. Notre étude sera axée sur l'attaque par factorisation du module de RSA et nous ferons une implémentation et une simulation de cette attaque.

Et **dans la partie III**, nous traiterons de l'analyse de RSA en étudiant l'amélioration de son efficacité.

CADRE THEORIQUE

Chapitre I : Cadre Théorique

Le cadre théorique de notre travail s'articule autour de la présentation de la problématique, des objectifs de l'étude ainsi que des hypothèses.

I-1- Problématique :

Les systèmes informatiques occupent une place prédominante dans les entreprises, dans les administrations et dans le quotidien des particuliers. Et avec l'essor de l'Internet, par les nombreux avantages qu'il offre et la diversité des services rendus accessibles, nous remarquons un énorme besoin de sécurité de l'information. Besoin soulevé par notre dépendance croissante aux systèmes informatiques dans divers aspects de la vie quotidienne et leur omniprésence. Et dans les efforts des gouvernements pour assurer la sécurité, ceux-ci ont recours à du personnel pour mettre en place des structures de chiffrement et de cryptanalyse.

Pour arriver à décrypter un message chiffré avec un cryptosystème robuste, il faut plusieurs années (AES, Triple DES, RSA, ElGamal…). C'est d'ailleurs sur cet élément que l'on juge la fiabilité du chiffrement. D'où le choix de RSA par plusieurs organisations pour assurer la protection des données. En effet, RSA est jugé indécryptable en termes de temps et de moyens humains et matériels. En effet, si multiplier deux grands nombres premiers p et q entre eux est relativement rapide pour n'importe quel ordinateur, factoriser le produit – c'est-à-dire retrouver p et q à partir de leur produit – est un problème difficile à résoudre, voir complexe, surtout lorsque la taille des nombres augmente. Notez que nous parlons ici d'un nombre composé de plusieurs centaines de chiffres.

Puisqu'il n'existe pas (encore) d'algorithme permettant de simplifier à l'extrême cette tâche, il faut alors chercher les nombres p et q en testant des combinaisons les unes après les autres et en vérifiant qu'il s'agit bien de nombres premiers. Et même avec des ordinateurs ultrapuissants, ce travail demande beaucoup de temps puisque la durée augmente de manière exponentielle avec la taille des nombres.

Cependant, en dépit des efforts conséquents qui ont été investis depuis un certain nombre d'années pour tenter d'endiguer les problèmes de sécurité, force est de constater que le nombre de vulnérabilités dans les systèmes informatiques et, de surcroît, les activités malveillantes qui essaient et qui réussissent à les exploiter, continuent régulièrement à se multiplier. Surtout dans un monde où la cryptographie repose sur des calculs mathématiques et sur le temps nécessaire à les mener à bien, la question n'est pas de savoir si un code sera cassé, mais quand.

Actuellement, avec l'avènement de l'informatique quantique [17] qui est révolutionnaire, d'ambitieux programmes sont en cours. En effet, elle pourrait permettre de résoudre des problématiques d'ordre sanitaires et même environnementales hors de portée des ordinateurs classiques. A cela s'ajoute le fait qu'elle puisse prendre moins de temps à décrypter des algorithmes bien établies (des milliards d'années pour un ordinateur classique) comme RSA. Elle utilise les principes de la mécanique quantique, à savoir la superposition et l'intrication [14], et repose sur les propriétés physiques du milieu porteur de la communication. De manière très générale, la cryptographie quantique permet de générer des clefs, localement ou à distance puis de les utiliser dans des protocoles de chiffrement (symétriques ou asymétriques) classiques, ou à terme, post-quantiques. Concrètement, il s'agit de coder l'information que l'on souhaite échanger sur l'état d'un système physique quantique (un état correspond à une information). La méthode la plus utilisée actuellement utilise la polarisation de la lumière, via les photons,

particules vectrices de lumière. Très utile pour partager des clefs privées, cette méthode, appelée Distribution quantique de clefs (QKD) [12], permet aussi de détecter très efficacement les « intrusions » sur un réseau de communication.

Toutefois, mis à part la cryptographie quantique [3] qui permet d'établir des protocoles de cryptographie pouvant atteindre des niveaux de sécurité prouvés ou jugés non atteignables en utilisant que des phénomènes classiques (non-quantiques), les spécialistes ont conçu la cryptographie post quantique [18] qui permet de garantir la sécurité de l'information face à un attaquant disposant d'un calculateur quantique [7]. Mais son nom prête à confusion car la cryptographie post quantique n'utilise pas des phénomènes quantiques. Et rien ne permet d'affirmer avec certitude que les nouveaux algorithmes y afférents ne pourront pas être contournés par les ordinateurs quantiques. En l'effet, les algorithmes quantiques de Shor, de Grover et de Simon [4] étendent les capacités par rapport à un attaquant ne disposant que d'un ordinateur classique. A l'heure actuelle, s'il n'existe pas de calculateur quantique représentant une menace concrète sur la sécurité des cryptosystèmes déployés, ces algorithmes permettent conceptuellement de résoudre certains problèmes calculatoires sur lesquels sont fondés plusieurs primitives populaires. Même si les scientifiques cherchent à préserver la sécurité de RSA (algorithme de Shor [9]), il faut tenir compte du fait que les attaquants seront prêts à user de tous les moyens pour venir à bout de RSA.

I-2- Objectifs :

Avec l'avènement de la quantique, toutes les théories formulées portent à croire que cette dernière met fin à la cryptographie mathématique. Mais dans ces mêmes théories, nous pouvons encore espérer pour l'avancée de la cryptographie. C'est dans le cadre de ce mémoire que nous avons pu noter comment les chercheurs s'y prennent pour essayer de protéger les cryptosystèmes. En effet, ils essaient de déployer des attaques à l'aide de la cryptanalyse pour contrer d'autres attaques externes à la cryptographie. Cette méthode les aide à analyser les situations générant un risque d'attaque.

Tout d'abord, l'objectif fondamental de la cryptographie **[11]** est de permettre une communication entre Alice et Bob sur un canal public peu sûr de telle façon qu'Eve (attaquant) ne soit pas en mesure de comprendre les données échangées.

Le but spécifique de la recherche est d'obtenir des connaissances et un aperçu sur une attaque par factorisation sur RSA et de vérifier la portée de RSA répond à toutes les attentes des utilisateurs comme indiqué. Mais encore de présenter quelques recommandations et solutions pour son évolution en fonction de l'évolution de la technologie.

I-3- Hypothèses :

La cryptographie RSA dépend de ce que l'on appelle l'«hypothèse de difficulté calculatoire» [13] selon laquelle un problème de théorie des nombres (tel que la factorisation d'un nombre entier ou le problème du logarithme discret) n'a pas de solution efficace. En revanche, ces hypothèses s'appuient sur la puissance de traitement des ordinateurs classiques.

Et face à la menace de l'ordinateur quantique vis-à-vis des algorithmes actuels de sécurisation des données, deux grands axes de transition se mettent en place : la cryptographie quantique et post quantique (voir Problématique). En effet, la mécanique quantique même si elle fait percevoir la fragilité fondamentale de la cryptographie mathématique, elle peut aussi proposer aussi une solution pour échapper aux incertitudes dont elle ne réussit pas à se libérer.

- *Quantique et Cryptographie à clef publique :*

Pour les problèmes de la factorisation d'un entier ***n***, le meilleur algorithme classique connu est en complexité temporelle heuristique [6] **:** $L_n[1/3, \sqrt[3]{64/9}]$.

Ce qui permet d'appuyer la sécurité d'algorithmes tels que la signature de Rabin à 128 bits en utilisant un module ***n*** de 3072 bits. La sécurité d'une signature RSA utilisant un tel module est alors d'au plus 128 bits face à un attaquant classique. Le meilleur algorithme quantique connu possède une complexité quadratique en n ; adapter la sécurité de RSA à ce contexte serait possible. Mais il requiert des tailles de clef phénoménales, de l'ordre du téraoctet. En pratique, au-delà des problèmes de stockage et de transmission de ce type de clefs et de tels messages, c'est que toutes les opérations sont ralenties. De plus, il n'existe pas de preuves que l'algorithme de Shor soit le plus efficace possible : chercher à préserver RSA dans ce contexte est une course perdue d'avance.

De manière générale, les algorithmes à clefs publiques basés sur la difficulté de la factorisation peuvent être considérés comme cassés par un attaquant quantique : RSA, Paillier, Naccache-Stern, Rabin, Benaloh, Blum-Goldwasser, Damgård–Jurik, Goldwasser-Micali, Okamoto-Uchiyama, Schmidt-Samoa. [5]

De même, les algorithmes à clefs publiques qui reposent sur la difficulté du calcul d'un logarithme discret peuvent être cassés par un attaquant quantique : Diffie-Hellman, Cramer-Shoup, DSA et ECDSA (aussi ses précurseurs Schnorr et ElGamal, et ses variantes comme EdDSA), SRP, etc.

D'autres problèmes classiquement difficiles reposant sur des problèmes d'arithmétique modulaire [10] sont eux aussi affectés par des variantes de l'algorithme de Shor.

I-4- Résultats attendus :

Ce travail de recherche permettra de mieux comprendre la cryptographie en général et plus précisément RSA et son importance.

Il s'agira d'étudier des différentes attaques possibles sur RSA et leurs enjeux, de choisir une technique d'attaque et de la mettre en œuvre.

Il s'agira également de comprendre les facteurs qui peuvent contribuer à l'amélioration de RSA. En effet, les constats faits sur les attaques permettront d'orienter et d'améliorer l'appui et le renforcement des capacités de RSA.


CADRE METHODOLOGIQUE

Chapitre II : Cadre Méthodologique

Dans ce chapitre, il sera question de rappeler quelques notions de la cryptologie, de développer les différentes techniques d'attaques sur le cryptosystème RSA mais également d'indiquer les méthodes ou techniques utilisées pour collecter des informations et analyser les résultats.

II-1- Présentation Générale :

II-1-1- Rappels sur la cryptologie :

La cryptologie est la science du secret. Elle se divise en deux disciplines :

- La cryptographie qui est l'étude des mécanismes basés sur des outils mathématiques pour assurer des services de sécurité, comme rappelé ci-dessus.
- La cryptanalyse qui est l'opération qui vise à rétablir une information inintelligible en information claire sans connaître la clé de chiffrement qui a été utilisée.

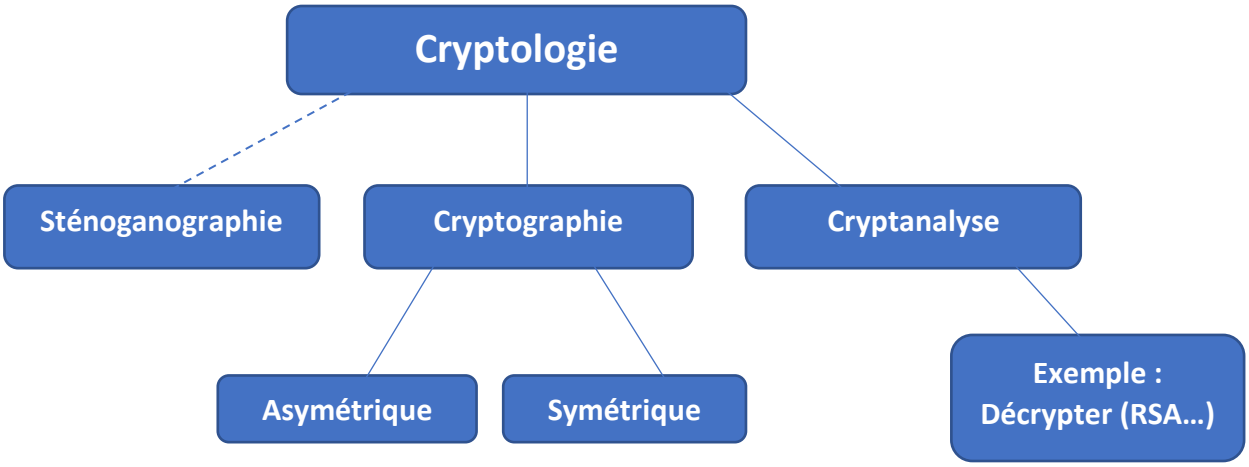


FIGURE 1. ORGANIGRAMME DE CRYPTOLOGIE

II-1-1-1- Stéganographie :

La stéganographie est une technique qui consiste à cacher des messages secrets dans d'autres messages, de sorte que l'existence même du secret est dissimulée. Généralement, l'expéditeur écrit un message inoffensif et dissimule un message secret dans la même feuille de papier.

Voici quelques exemples de procédés sténographiques utilisés :

- *L'encre invisible* : Au 1er siècle av. J-C, apparu une méthode de dissimulation de message consistant à écrire avec du lait ou du jus de citron. Il suffisait de chauffer le support pour faire réapparaître le message.
- *Le crâne rasé :* A l'antiquité, les Grecs utilisaient parfois la tête des esclaves pour tatouer des messages. Ils étaient ensuite envoyés chez le destinataire et après un long voyage, il fallait raser leur tête dont les cheveux avaient repoussé, pour lire le message.

- *Le micro point :* Pendant la seconde guerre mondiale, les Allemands utilisaient la technique du micro-point, consistant à dissimuler des photos ou des textes dans un point de ponctuation d'une lettre. Il suffisait alors d'agrandir le point pour voir apparaître le message.

II-1-1-2- Cryptographie :

Le terme de la cryptographie a été définie dans l'Introduction. Cependant, avant d'aborder notre sujet, il est nécessaire de cerner la terminologie et quelques aspects de cette science. La cryptographie a été inventée, à ses débuts, dans le but de sécuriser des communications, secrètes de l'expéditeur. Étymologiquement, le mot cryptographie signifie « écriture secrète » : elle vient du mot grec « kryptos » qui signifie « cacher », et « graphein » qui signifie « écrire ». Actuellement, c'un terme générique qui signifie l'étude et la conception de mécanismes basés sur les outils mathématiques pour assurer des services de sécurité (Intégrité, Authentification, Confidentialité et Non répudiation).

II-1-1-2-1. Terminologie :

Cryptographie : étymologiquement « écriture secrète », devenue par extension l'étude de cet art (donc aujourd'hui la science visant à créer des cryptogrammes, c'est-à-dire à chiffrer) ;

Cryptanalyse ou analyse cryptographique : L'opération qui consiste à transformer l'information chiffrée en information claire sans connaissance initiale de la clé utilisée lors du chiffrement.

Cryptologie : science regroupant la cryptographie et la cryptanalyse

Chiffrement **:** C'est l'opération qui consiste à transformer une information claire en une information chiffrée.

Déchiffrement : C'est l'opération inverse du chiffrement

Surchiffrement : C'est l'opération de chiffrement appliquée à l'antigramme.

Chiffre **:** C'est l'ensemble des moyens et prestations de cryptologie qui permettent de transformer, à l'aide de codes secrets, des informations claires en informations inintelligibles à toute personne non qualifiée pour les connaître, ou à réaliser l'opération inverse grâce à des moyens matériels ou logiciels conçus à cet effet.

Cryptogramme : C'est le texte chiffré ou d'une manière générale, l'information chiffrée résultant d'une opération de chiffrement, d'apparence inintelligible et présentant un caractère secret.

Chiffrer : C'est transformer une information claire (appelé libellé ou texte claire, message claire, etc.) en une information chiffrée (appelé texte chiffré, message chiffré, cryptogramme y compris les en-têtes et les signatures …. Etc.), inintelligible à toute personne non autorisée).

Antigramme : C'est l'information chiffrée résultant d'une première transformation, et préparant une seconde transformation (immédiate) obligatoire.

Cryptosystème ou Système de chiffrement : Un système de chiffrement ou crypto système est l'application concrète d'un procédé de chiffrement avec toutes les clefs possibles.

Décrypter : retrouver le message clair correspondant à un message chiffré sans posséder la clé de déchiffrement (terme que ne possèdent pas les anglophones, qui eux « cassent » des codes secrets) ;

Cryptolecte : jargon réservé à un groupe restreint de personnes désirant dissimuler leurs communications.

Clé de chiffrement : La transformation de l'information nécessite une clef de chiffrement K délivrée par exemple par un générateur de clefs. Une clef de chiffrement est une fonction de chiffrement, une convention ou un ensemble de conventions qui fixe la manière de chiffrer (ou de déchiffrer). Cette clef peut, par exemple être littérale (ou grammatique), géométrique ou numérique (ex : suite de nombres, de bits, de longueur données-40, 60, 128 256, 1024 bits … etc.).

Algorithme de chiffrement : Ensemble de règles mathématiques (logiques) utilisées pour résoudre les problèmes de chiffrement et de déchiffrement.

Canal sûr : Moyen de communication sûr entre deux entités tel qu'un adversaire (éventuellement un décrypteur) ne puisse pas changer l'ordre, supprimer, insérer ou lire de l'information.

Cryptogramme : C'est le texte chiffré ou d'une manière générale, l'information chiffrée résultant d'une opération de chiffrement, d'apparence inintelligible et présentant un caractère secret.

Procédé de chiffrement : Le procédé de chiffrement fixe le mode opératoire pour chiffrer et déchiffrer ; il s'appuie sur l'algorithme.

Réseaux de chiffrement : Un réseau de chiffrement c'est l'ensemble des correspondants utilisant les mêmes systèmes de chiffrement, et qui échangent entre eux des messages chiffrés.

Fonction à sens unique : Une fonction à sens unique est une fonction facile à calculer dans un sens, mais l'inverse est impossible à produire. En matière de cryptographie, ce type de fonction est très utile, à condition qu'il existe un moyen secret y qui à partir de f(x) permet de déterminer x. Il s'agit alors de fonction à sens unique à brèche secrète.

Fonction de hachage : Une fonction de hachage est une fonction à sens unique qui produit un résumé (une empreinte) d'un message qui ne peut pas être inversé pour reproduire l'original.

Protocole cryptographique : Un protocole cryptographique est un ensemble de règles ou conventions définissant un échange de messages, généralement chiffrés, entre deux (ou plus) correspondants.

La signature numérique ou électronique : C'est un service cryptographique permettant de vérifier l'authenticité de l'expéditeur ainsi que l'intégrité du message reçu, et d'en assurer la non répudiation (Services d'authentification, d'intégrité et de non répudiation).

II-1-1-2-2. Quelques repères historiques :

Tout au long de l'histoire, la cryptographie s'est insérée dans les différents contextes historiques, et a progressé avec les évolutions mathématiques. La plupart du temps elle est utilisée dans un milieu politique ou militaire.

Historiquement le développement de cryptographie est passé par trois âges :

a- Antiquité :

Jusqu'à au XXème siècle (fin de la 1ère guerre mondiale), on entend habituellement les méthodes cryptographiques qui requièrent simplement du papier et un crayon. Il n'est pas surprenant que ces systèmes soient les plus anciens.

Les premières traces de cryptographie remontent au XVIème siècle avant J-C. On a retrouvé à cette époque sur les bords du Tigre en Irak le plus vieux document chiffré connu. Dans l'antiquité on remarque alors l'émergence de nouveaux modes de cryptographie, pour la plupart propres à un peuple. C'est pourquoi nous allons ici organiser l'étude avec ces différentes civilisations.

- *Hébreux (chiffre Atbesh) :*

La plus connue et la plus simple est sans doute le chiffre Atbesh. Datant du Vème siècle avant J-C, ce procédé consiste à faire correspondre l'alphabet classique avec l'alphabet inversé, et d'associer ainsi à chaque lettre, la lettre correspondante en position dans l'alphabet inversé. Voici un exemple pour l'alphabet latin.

- *Grecs (Scytale) :*

A la même époque que les hébreux, les grecs commencèrent à développer quelques méthodes de codage. Parmi celles-ci : la scytale (ou scytale spartiate). La scytale est une des premières techniques de codage par transposition. Elle date du Vème siècle av J-C. Le principe est le suivant : celui qui est l'auteur du message va préalablement enrouler une bande (de parchemin ou de cuir) en hélice autour d'un bâton régulier de diamètre fixé sur laquelle il va pouvoir écrire. Après écriture, l'auteur déroule la bande qui présente alors une succession de caractères n'ayant aucun sens logique. Celui qui veut lire le message doit alors connaître le diamètre du bâton initial puis y enrouler de nouveau la bande.

FIGURE 2. SCYTALE SPATIATE

- *Romains (le chiffre de César) :*

Jules César utilisait un mécanisme de confidentialité rudimentaire, où chaque lettre d'un message était remplacée par celle située trois positions plus loin dans l'alphabet. La méthode se généralise et prend le nom de substitution.

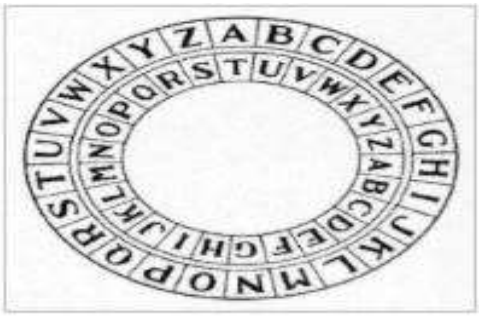


FIGURE 3. ROUE DE CESAR

b- Age technique (Systèmes mécaniques et électromécaniques) :

Le XXème siècle (1919 - 1975), utilisant des appareils mécaniques ont donc été mises au point. L'âge technique garde les substitutions et les permutations mais les met en œuvre à l'aide de machines mécaniques ou électromécaniques. Les plus célèbres sont l'Enigma et la Hebern.

- *Machine Enigma :*

La machine Enigma reste l'une des plus remarquables machines à crypter toutes périodes confondues ; elle a longtemps servi de modèle et a marqué son époque. La naissance de la machine Enigma est initiée en 1919 par un ingénieur hollandais, Hugo Alexander, qui breveté un dispositif de codage électromécanique ; ses idées sont réutilisées par le Dr Arthur Scherbius qui donne le nom Enigma à sa machine initialement développée pour les civils. Cependant, sa société fera faillite peu après, en revanche, les militaires allemands vont lui accorder un grand intérêt pendant la seconde guerre mondiale.

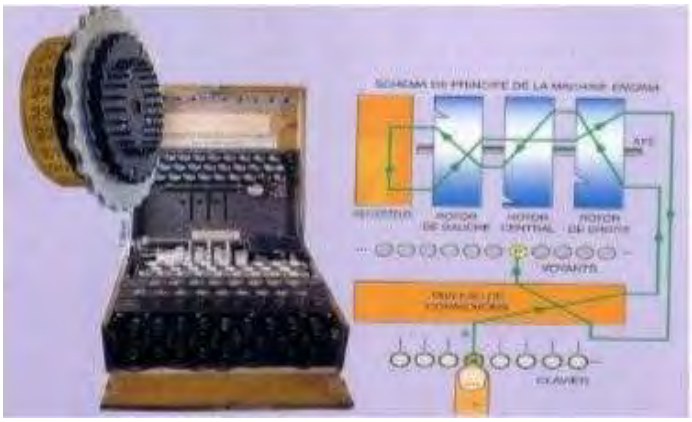


FIGURE 4. LA MACHINE ENIGMA

- *Machine Hebern :*

En 1910, Edward H. Hebern commença à mettre au point des machines de chiffrement aux États-Unis. Au début des années 1920, il inventa la machine à code électrique qui utilisait seule roue codeuse. Il conçut aussi, en 1924, des appareils à trois et cinq roues codeuses comme prototypes pour la marine américaine. Très différentes de l'Enigma, les machines Hebern utilisaient des roues codeuses dont le filage interne pouvait facilement être changé. De plus, ces roues codeuses pouvaient fonctionner dans le sens de rotation conventionnel ou dans le sens contraire. Par contre, ces machines possédaient toutes des faiblesses du point de vue cryptanalytique. Ces faiblesses sont décrites par William Friedman.

c- Age scientifique :

Il a commencé en 1976 avec la prolifération des ordinateurs et l'essor de leur connectivité (méthodes modernes). Plusieurs méthodes cryptographiques utilisent maintenant des logiciels plutôt que des machines électromécaniques.

Il voit l'introduction de mécanismes donnant des réponses positives à des questions a priori hors d'atteinte :

- *Comment assurer un service de confidentialité sans avoir au préalable établi une convention secrète commune ?*
- *Comment assurer un service d'authenticité basé sur la possession d'un secret sans révéler la moindre information sur le secret ?*

- *Algorithme de chiffrement symétrique DES :*

Les principes des algorithmes symétriques commerciaux modernes, principalement le DES (Data Encryption Standard), ont été mis au point dans les années 1970 par IBM avec l'aide de la NSA (National Security Agency). Ils sont basés sur des réseaux de substitution et de de transposition. DES n'est plus à présent sûrs compte tenu de la longueur faible des blocs clairs (64 bits) et des clés (56 bits). Cependant, il y'a l'avènement d'algorithmes plus sûr comme AES dont les clés sont de longueur de 128 bits à 256 bits. Leur cryptanalyse a fait des progrès (cryptanalyse linéaire et différentielle), ce qui permet de mieux cerner leur sûreté cryptologique.

- *Algorithme de chiffrement asymétrique RSA :*

L'algorithme à clés publiques le plus répandu est l'algorithme RSA (Rivest-Shamir & Adleman), publié en 1978. Il est fondé sur les propriétés des nombres premiers. Pour casser l'algorithme, il faut factoriser un grand nombre entier (fourni par la clé publique).

L'algorithme RSA est malheureusement beaucoup plus coûteux à mettre en œuvre que le DES, et son usage est généralement utilisé pour le codage de données de taille limitée. C'est en particulier une excellente solution pour la diffusion des clés du DES.

II-1-1-2-3. Objectifs Fondamentaux et Qualités d'un crypto-système :

Dès que plusieurs entités sont impliquées dans un échange de messages sécurisés, il est nécessaire de faire appel aux protocoles cryptographiques afin de sécuriser la communication.

Lorsque l'on parle de "sécuriser un échange", on souhaite prêter attention aux 3 services de sécurité suivants : la confidentialité, l'intégrité et l'authentification :

a- Intégrité des données :

C'est la prévention d'une modification non autorisée de l'information. L'intégrité du système et de l'information garantit que ceux-ci ne sont modifiés que par une action volontaire et légitime. Les attaques contre l'intégrité sont appelées « substitutions ».

b- Confidentialité :

La confidentialité est la propriété qu'une information n'est ni disponible ni divulguée aux personnes, entités ou processus non autorisés (selon la norme ISO 7498-2). La confidentialité se définit également comme le caractère réservé d'une information dont l'accès est limité aux personnes autorisées à la connaitre pour accomplir leurs missions.

c- Authentification :

L'authentification consiste à vérifier l'identité des différentes parties impliquées dans le dialogue. Concrètement, lorsque de l'information est échangée, l'authentification du message garantit son origine et sa destination. Les attaques contre l'authenticité sont appelées "mascarades".

d- Non-Répudiation :

La non-répudiation consiste à prouver qu'un message a bien été émis par son expéditeur ou reçu par son destinataire. Plus généralement, la non-répudiation consiste à garantir que l'auteur d'un message ou d'un document ne peut pas nier l'avoir écrit ou transmis. Elle se décompose comme suit :

- ***Non-répudiation d'originel :*** l'émetteur ne peut nier avoir écrit le message.

- ***Non-répudiation de réception :*** Le receveur ne peut nier avoir reçu le message.

- ***Non-répudiation de transmission :*** l'émetteur du message ne peut nier avoir reçu le message.

II-1-1-2-4. Les principaux concepts cryptographiques :

a- Introduction :

On se placera dans la problématique d'un *émetteur* et d'un *récepteur* désirant échanger un message sur un canal de transmission public (donc ouvert à la possibilité qu'une tierce personne intercepte le message). Le but est de décrire et d'analyser des procédés (cela inclus leurs implémentations concrètes) permettant de transformer le message original, que l'on désignera par message clair (ou texte clair), en un message chiffré. Ce procédé sera appelé *chiffrement)*.

Une fois que le message est chiffré, il transite via un canal non sécurisé vers son destinataire. Afin de déchiffrer le message, le destinataire doit lui faire subir l'opération inverse du chiffrement qui est le déchiffrement. Cela permet d'assurer la confidentialité du message. On parlera alors de message chiffré. En pratique, nous ne considérerons que des messages de type texte. Les méthodes que nous décrirons devront éventuellement être adaptées si l'on souhaite chiffrer du son ou/et de l'image, même si du point de vue de l'ordinateur ce sera une suite de nombres binaires.

- Notations :

Nous adopterons les notations suivantes :

- $x = \{x_1, x_2, x_3, \ldots, x_i, \ldots, x_n\}$ = Ensemble des unités de chiffrement représentant l'information claire

- $y = \{y_1, y_2, y_3, \ldots, y_i, \ldots, y_n\}$ = Ensemble des unités cryptographiques représentant l'information chiffrée (ou la cryptogramme)

- $K = k_1, k_2, k_3, \ldots\ldots\ldots k_i, \ldots, k_n\}$ = Ensemble fini de clefs de chiffrement

- $E = \{E_{k1}, E_{k2}, E_{k3}, \ldots., E_{ki}, \ldots., E_{kn}\}$ = Ensemble des transformations, des fonctions de chiffrement possibles.

- $D = \{D_{k1}, D_{k2}, D_{k3}, \ldots\ldots D_{ki}, \ldots, D_{kn}\}$ = Ensemble des fonctions de déchiffrement possibles.

- $\forall k \in K, \forall E_k \in E / E_k : x \text{ -----> } y$

$$x_i \text{ -----> } y_i = E_k (x_i)$$

- $\forall k \in K, \forall D_k \in D / D_k : y \text{ -------> } x$

$$y_i \text{ ------> } x_i = D_k (y_i)$$

- Ek et Dk sont des transformations telles que :

$y_i = E_k(x_i)$; $x_i = D_k (y_i) = D_k [E_k(x_i)] = D_{k0} E_k(x_i)$

$D_{k0} E_k = I$ ------> $D_k = E_k^{-1}, \forall x_i \in x$

b- Principe de la substitution :

C'est le remplacement des unités claires par d'autres unités secrètes.

Exemple : Soit x=y= {les 26 lettres de l'alphabet normal} et la permutation suivante Π :

(a b c d ……. w x y z)

Π = (

(X M N Q …….. V L I Y)

Le chiffrement du clair pourra se faire avec la fonction de chiffrement suivante, qui permettra de substituer à chaque unité claire, l'unité secrète correspondante et toujours la même :

E_Π : x ----------> $E_\Pi (x) = y$

x_i ----------> $E_\Pi (x_i) = y_i = \Pi (x_i)$

⇨ E_Π (a) = X ; E_Π (b) = M ; E_Π (c) = N … etc.

Le déchiffrement se fera avec la fonction de déchiffrement associée à la permutation inverse Π^{-1}.

E_Π^{-1} (X) = a ; E_Π^{-1} (M) = b ; E_Π^{-1} (N) = c

Nous distinguons les méthodes de chiffrement par la substitution suivantes :

- les substitutions monoalphabétiques (substitutions simples à représentations uniques, à représentations multiples, bigrammatiques , codiques)
- les substitutions polyalphabétiques ou à double clef (classique ou non classique)

c- ***Principe de la transposition :***

C'est le mélange des unités de chiffrement afin d'obtenir des unités secrètes.

Exemple :

Soit x=y, et un entier m > 0 tel que K soit l'ensemble des permutations de {1,2,3,……,m}.

∀ Π une permutation, on définit :

$E_\Pi (x_1, x_2, x_3, …., x_m) = (y_1, y_2, y_3, … , y_m) = (x_{\Pi(1)}, x_{\Pi(2)}, …, x_{\Pi(m)})$,

et

$D_\Pi (y_1, y_2, y_3, …., y_m) = (x_1, x_2, x_3, ..., y_m) = (y_{\Pi(1)}^{-1}, y_{\Pi(2)}^{-1}, …, y_{\Pi(m)}^{-1})$

Application :

Soit m=6 et une permutation, telles que :

$$\Pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 5 & 1 & 6 & 4 & 2 \end{pmatrix} \qquad \Pi^{-1} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 6 & 1 & 5 & 2 & 4 \end{pmatrix}$$

En appliquant E_Π au texte clair suivant, il vient :

X = clair = ‘’ Elle vend des coquillages sur le sol ‘’

1234 5612 345 61234561234 561 23 456

3516 4235 146 23516423516 428 51 642

Y = crypto = LVEE ELDS NCE DULOLIQESAU SGE OR LSL

Ces lettres du cryptogramme peuvent être relevées et présentées d’une autre manière différente, suivant une convention fixée.

Nous distinguons les méthodes de chiffrement par la transposition :

- les transpositions simples (ordinaires à tableaux, par grilles, améliorées à tableaux)
- les doubles transpositions (ordinaires, améliorées)

Il existe des méthodes de chiffrement combinées qui combine les méthodes de transposition et de substitution.

d- Principes de Kerckhoff :

Un algorithme de chiffrement (voir la terminologie ci-dessus) désigne les fonctions mathématiques utilisées pour le chiffrement et le déchiffrement. Ces fonctions peuvent aussi être regroupées au sein d'une seule fonction (ou algorithme). Si initialement, la sécurité d'un cryptosystème reposait entièrement sur le fait que l'algorithme de cryptographie était tenu secret, ce n'est que vers la fin du 19ème siècle qu'a été reconnu le fait que la sécurité ne devait pas reposer sur la méthode cryptographique. Les articles d'Auguste Kerckhoff (« La cryptographie militaire », Journal des sciences militaires, vol. IX, pp. 5 U-38, Janvier 1883, pp. 161 -U 191, Février 1883) sont les précurseurs des fondements de la cryptographie moderne.

Voici les six principes de Kerckhoff pour la conception de « cryptosystème militaire » :

1- Le système doit être matériellement, sinon mathématiquement, indécryptable ;
2- Il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi ;
3- La clef doit pouvoir en être communiquée et retenue sans le secours de notes écrites, et être changée ou modifiée au gré des correspondants ;
4- Il faut qu'il soit applicable à la correspondance télégraphique ;
5- Il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes ;
6- Enfin, il est nécessaire, vu les circonstances qui en commandent l'application, que le système soit d'un usage facile, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer.

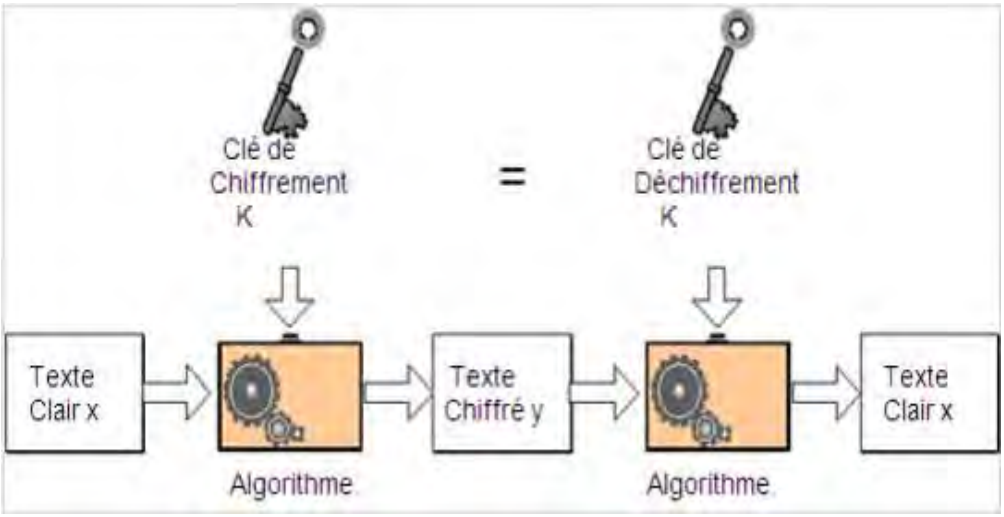
En pratique, le respect des règles de Kerckhoffs impose que le système intègre des clefs pour le chiffrement et le déchiffrement. Du point de vue mathématique, le cryptosystème est composé d'un algorithme, de tous les messages clars, de tous les messages chiffrés,et des clefs possibles.

e- La cryptographie symétrique :

Les clés de chiffrement et de déchiffrement sont identiques ou peuvent être facilement calculées l'une à partir de l'autre.

Principe : Les algorithmes mis en œuvre sont à clefs secrètes, basés sur les procédés de substitution et de transpositions des unités claires.

- La même clef est utilisée pour chiffrer et déchiffrer, et elle doit être gardée secrète par les utilisateurs.
  $K_E = K_D = K$



Génération des clés : La clef est choisie aléatoirement dans l'espace des clefs (ou Ensemble des clefs possibles)

Taille des clés : Elle est souvent de l'ordre de 64 bits (standard) ; certains algorithmes en utilisent 56, 128, 256…

Avantage : l'avantage principal du chiffrement symétrique est sa rapidité (quelques dixièmes de secondes pour un texte clair d'un mégaoctet) et il est bien adapté pour la protection du secret.

Inconvénient : Le principal inconvénient réside dans la distribution des clefs. Par mesure de sécurité, l'échange des clefs se fait de manière manuelle, mais ce mode de distribution, et partant de gestion des clefs devient vite compliquée et inextricable quand le nombre de correspondants augmente ; le nombre de clefs pour un réseau de n correspondants étant, en effet de : **[n(n-1)] /2**

Il existe deux catégories de chiffrement symétrique (noter que maintenant l'expression a un sens, puisque chiffrement et déchiffrement sont des algorithmes similaires) :

- ***Chiffrement par flots (stream ciphers) :*** ce sont des systèmes de chiffrement qui opèrent sur le message clair par bit (ou quelquefois par petit groupement de bits). Il est plus complexe que le chiffrement par blocs. Il utilise XOR pour le cryptage qui peut facilement être inversé au texte brut. Modèles d'algorithmes utilisés : CFB (Cipher FeedBack) et OFB (Output FeedBack) ; Nombre de bits utilisés : 8bits.
- ***Chiffrement par blocs (block ciphers) :*** ce sont des systèmes de chiffrement qui opèrent sur le message clair par « grand » groupes de bits (bloc). Il utilise le schéma de Feistel. Il est très difficile d'inverser le texte crypté. Modèles d'algorithmes utilisés : ECB (Electronic Code Book) et CBC (Cipher Block Chaining) ; Nombre de bits utilisés : 64bits ou plus.

Moralement, les systèmes par flots opèrent sur les caractères, alors que ceux par blocs opèrent sur les mots.

Exemple : Les plus connus : DES, triple DES, AES, RC4, masque jetable.

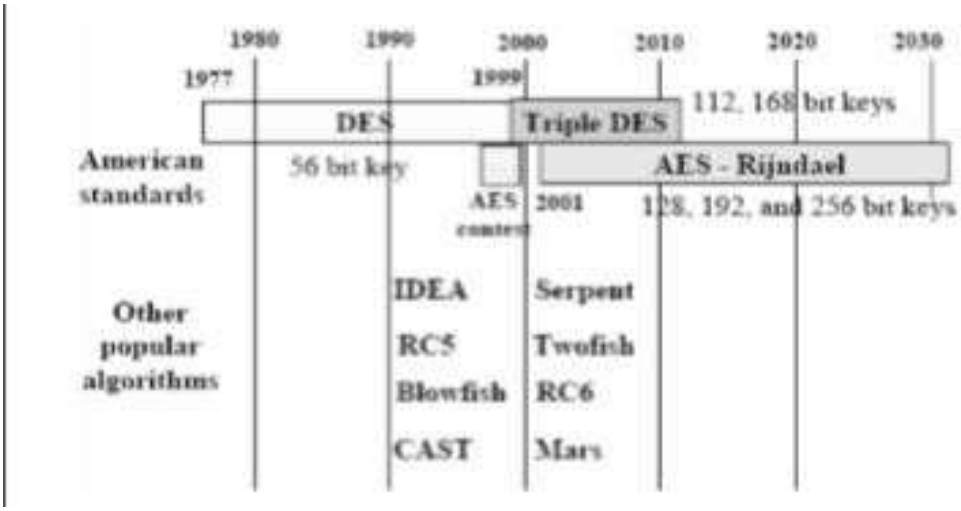


FIGURE 5. CHIFFREMENTS SYMETRIQUES LES PLUS FREQUENTS

e-1- DES :

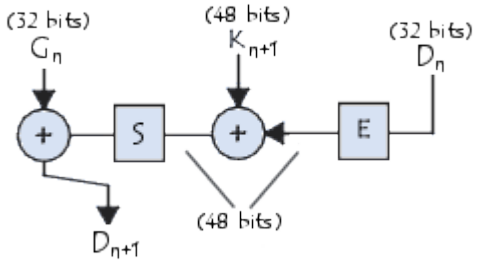
Le D.E.S. (Data Encryption Standard, c'est-à-dire Standard de Chiffrement de Données) est un standard mondial depuis plus de 35 ans. Bien qu'il soit un peu vieillissant, il résiste toujours très bien à la cryptanalyse et reste un algorithme très sûr.

Au début des années 70, le développement des communications entre ordinateurs a nécessité la mise en place d'un standard de chiffrement de données pour limiter la prolifération d'algorithmes différents ne pouvant pas communiquer entre eux. Pour résoudre ce problème, l'Agence Nationale de Sécurité américaine (N.S.A.) a lancé des appels d'offres. La société I.B.M. a développé alors un algorithme nommé Lucifer, relativement complexe et sophistiqué. Après quelques années de discussions et de modifications, cet algorithme, devenu alors D.E.S., fut adopté au niveau fédéral le 23 novembre 1976.

- ***CHIFFREMENT :***

Algorithme :

Equation de chiffrement :



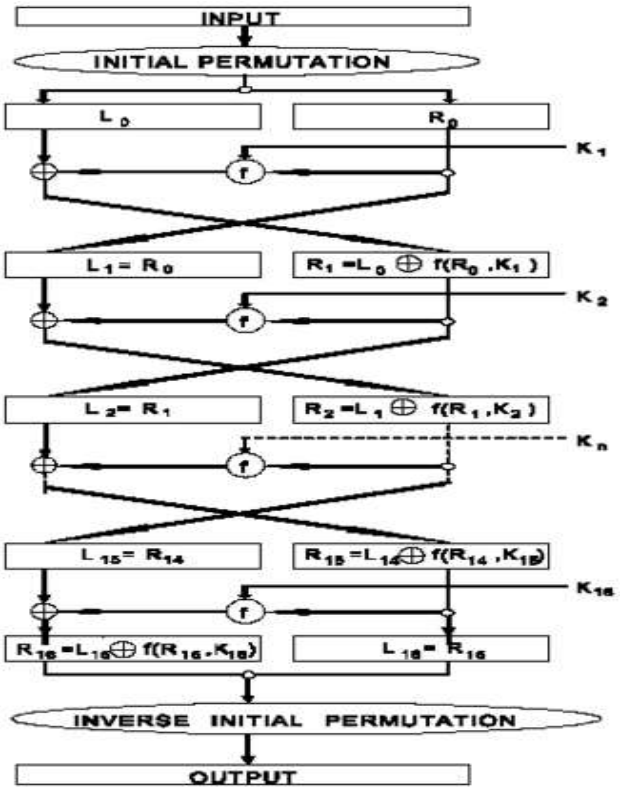


FIGURE 6. ALGORITHME DU DES

- ***Etapes de l'algorithme :***

1. **Fractionnement du texte en blocs de 64bits (8octets)**
2. **Permutation Initiale des blocs**
3. **Découpage des blocs en 2 parties : gauche et droite, nommées L et R ;**
4. **Etapes de permutation et de substitution répétées 16 fois (appélées rondes) : Calcul Médian**
5. **Recollement des parties gauche et droite puis permutation initiale inverse (permutation finale).**

Les Clés :

Etant donné que l'algorithme du DES présenté ci-dessus est public, toute la sécurité repose sur la complexité des clés de chiffrement. Les étapes ci-dessous montre comment obtenir 8 clés diversifiées de 48 bits chacune servant dans l'algorithme DES à partir d'une clé de 64bits.

- Clé initiale de 64 bits
  1. *Réduction à 56 bits*
  2. *Division de la clé en 2 sous-clés de 28 bits*
  3. *Rotation de la clé :* A chaque tour, chaque sous-clé subit une rotation d'un ou 2 bits vers la gauche
  4. *Regroupement des 2 sous-clés*
  5. *Réduction :* La sous-clé résultante (56 bits) est réduite à une sous-clé de 48 bits

- ***DECHIFFREMENT :***

Appliquer le même algorithme mais à l'inverse en tenant bien compte que chaque itération du déchiffrement traite les mêmes paires de blocs utilisés dans le chiffrement

$$R_{n-1} = L_n$$
$$L_{n-1} = R_n \oplus f(L_n, K_n)$$

Voir Chiffrement pour R et L : bloc droite (R), bloc gauche (L)

 => XOR « ou exclusive »

- **Quelques Faiblesses et Solutions :**

  - ***Faiblesses :***
    - Les S-Boxes contiendraient des failles
    - La faible taille de la clé
    - Sensible à l'attaque par la force brute (recherche exhaustive), à la cryptanalyse différentielle qui nécessite moins de temps de travail, à la cryptanalyse linéaire qui correspond à l'attaque à clairs connus (la plus efficace connue à ce jour contre DES)
  - ***Solution :***
- Il est déconseillé d'utiliser DES pour protéger des données sensibles, notamment gouvernementales.
- Il a été proposé d'utiliser le DES deux fois (Double DES) ou trois fois (Triple DES) pour en renforcer la sécurité. Un autre algorithme DESX a été également proposé comme amélioration du DES (par Ron Rivest de RSA data security).

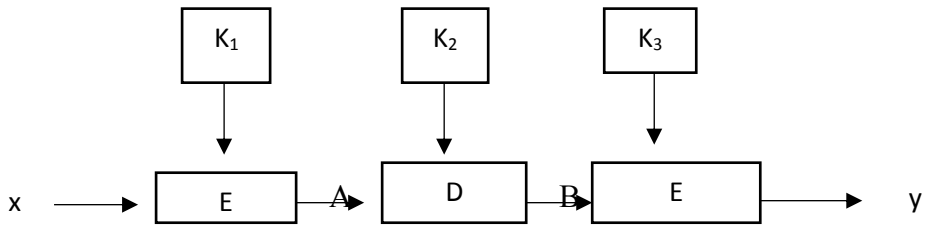
e-2- Triple DES :

Cet algorithme utilise le DES trois fois en série avec des clefs différentes :

- On chiffre avec la 1ère clef, on déchiffre avec la 2ème et on rechiffre avec la 3ème.

$Y = E_{k3}(D_{k2}(E_{k1}(x)))$ avec $D_{k2} = E_{k2}^{-1}$ et $k_1 \neq k_2 \neq k_3$ (d'une manière générale).

- **CHIFFREMENT :**



1ere possibilité : Elle utilise deux clés ($K_1 = K_3$)

(K_1, K_2, K_1)

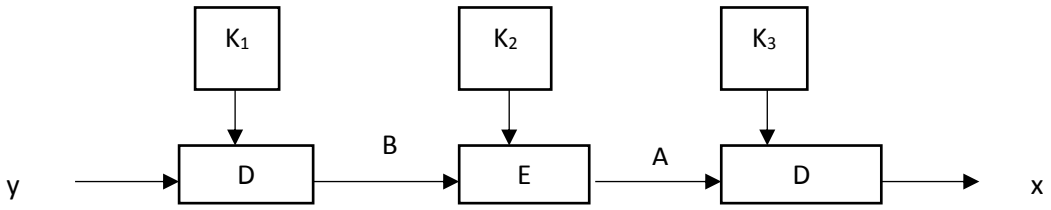
$E_{K1}[E_{K2}^{-1}[E_{K1}(x)]] = y$ ==> 2^{112} clefs possibles

2e possibilité : Elle utilise 3 clefs ($K_1 \neq K_2 \neq K_3$)

(K_1, K_2, K_3)

$\mathbf{E_{K3}\,[E_{K2}{}^{-1}\,[E_{K1}\,(x)]] = y}$

- **DECHIFFREMENT :**



1ere possibilité : Elle utilise deux clés ($K_1 = K_3$)

(K_1, K_2, K_1)

$\mathbf{E_{K1}{}^{-1}\,[E_{K2}\,E_{K1}{}^{-1}\,(y)] = x}$

2e possibilité : Elle utilise 3 clefs ($K_1 \neq K_2 \neq K_3$)

(K_1, K_2, K_3)

$\mathbf{E_{K1}{}^{-1}[E_{K2}\,[E_{K3}{}^{-1}\,(y)]] = x}$ ==> 2^{168} clefs possibles

Conclusion : Elle est très robuste contre toutes les attaques faisables connues. Mais, il est beaucoup plus lent que le DES car il triple les opérations.

Remarques :

- Le Triple DES permet d'éviter les problèmes liés à la taille de la clé trop courte du DES ; la programmation est simple et la sécurité est améliorée ;
- Mais son principal défaut est : il n'est pas rapide.

Solution : Il faudrait migrer vers d'autres algorithmes plus robustes (IDEA, AES, …)

e-3- AES :

L'AES (Advanced Encryption Standard) est, comme son nom l'indique, un standard de chiffrement symétrique destiné à remplacer le DES (Data Encryption Standard) qui est devenu trop faible au regard des attaques actuelles.

Historiquement, le développement de l'AES a été instigué par le NIST (National Institue of Standards and Technology) le 2 janvier 1997. L'algorithme a été choisi il y a peu de temps : il s'agit de l'algorithme Rijndael (prononcer "Raindal"). Cet algorithme suit les spécifications suivantes :

- L'AES est un standard, donc libre d'utilisation, sans restriction d'usage ni brevet.
- C'est un algorithme de type symétrique (comme le DES)
- C'est un algorithme de chiffrement par blocs (comme le DES)

- Il supporte différentes combinaisons [longueur de clé] - [longueur de bloc] : 128-128, 192-128 et 256-128 bits (en fait, Rijndael supporte également des tailles de blocs variables, mais cela n'est pas retenu dans le standard)

En termes décimaux, ces différentes tailles possibles signifient concrètement que :

3.4 x 1038 clés de 128-bit possibles

6.2 x 1057 clés de 192-bit possibles

1.1 x 1077 clés de 256-bit possibles

Pour avoir un ordre d'idée, les clés DES ont une longueur de 56 bits (64 bits au total dont 8 pour les contrôles de parité), ce qui signifie qu'il y a approximativement 7.2 x 1016 clés différentes possibles. Cela nous donne un ordre de 1021 fois plus de clés 128 bits pour l'AES que de clés 56 bits pour le DES.

- ***CHIFFREMENT :***

Algorithme :

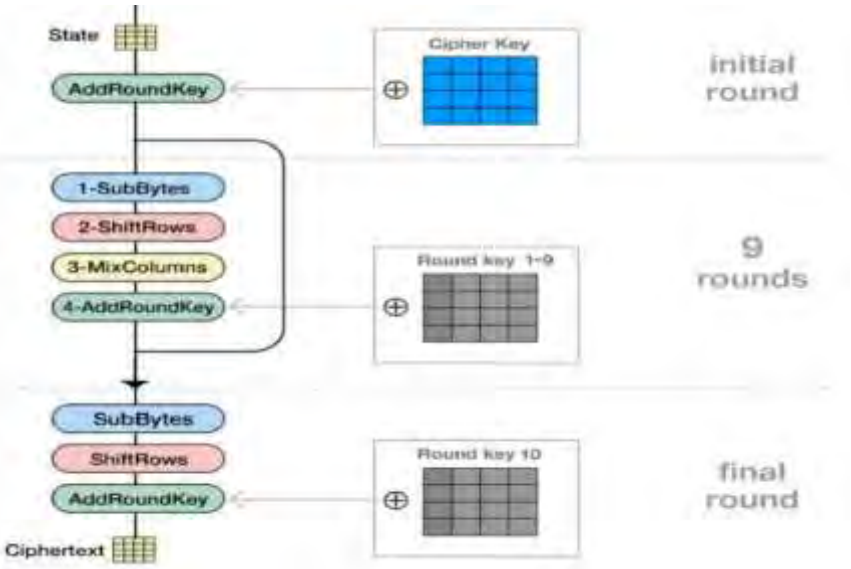


FIGURE 7. PRINCIPE CHIFFREMENT AES

Description du schéma :

Le schéma suivant décrit succinctement le déroulement du chiffrement :

1) On calcule la clef étendue

2) On effectue un AddRoundKey initial (``tour 0'')

3) On effectue (Nr-1) tours :

- ByteSub (Etat) ;

- ShiftRow (Etat) ;

- MixColumn (Etat) ;

- AddRoundKey (Etat, Ki) ;

4) On effectue un tour final :

- ByteSub (Etat);

- ShiftRow (Etat);

- AddRoundKey (Etat, Ki);

Un état peut également être considéré comme un tableau de colonnes, chaque colonne comprenant 4 octets / 32 bits, i.e. de mots de 32 bits.

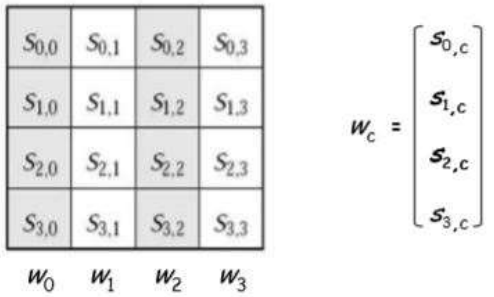


FIGURE 8. ETAT

Fonctionnement :

L'algorithme prend en entrée un bloc de 128 bits (16 octets), la clé fait 128, 192 ou 256 bits. Les 16 octets en entrée sont permutés selon une table définie au préalable. Ces octets sont ensuite placés dans une matrice de 4x4 éléments et ses lignes subissent une rotation vers la droite. L'incrément pour la rotation varie selon le numéro de la ligne. Une transformation linéaire est ensuite appliquée sur la matrice, elle consiste en la multiplication binaire de chaque élément de la matrice avec des polynômes issus d'une matrice auxiliaire, cette multiplication est soumise à des règles spéciales selon GF (2^8) (groupe de Galois ou corps fini). La transformation linéaire garantit une meilleure diffusion (propagation des bits dans la structure) sur plusieurs tours.

Finalement, un OU exclusif XOR entre la matrice et une autre matrice permet d'obtenir une matrice intermédiaire. Ces différentes opérations sont répétées plusieurs fois et définissent un « tour ». Pour une clé de 128, 192 ou 256, AES nécessite respectivement 10, 12 ou 14 tours.

Une même clef secrète est utilisée pour les opérations de chiffrement et de déchiffrement (c'est un secret partagé entre l'expéditeur et le destinataire du message).

AES est un algorithme de chiffrement par blocs, les données sont traitées par blocs de 128 bits pour le texte clair et le chiffré. La clef secrète a une longueur de 128 bits, d'où le nom de version : AES 128 (il existe deux autres variantes dont la clef fait respectivement 192 et 256 bits).

Etapes :

A chaque round quatre transformations sont appliquées (sauf à la dernière ou 3 sont appliquées)

- Substitution d'octets dans le tableau d'état : ByteSub
- Décalage d'octets dans le tableau d'état : ShiftRow
- Déplacement de colonnes dans le tableau dans le tableau d'état : MixColumn
- Addition d'une clef de round qui varie à chaque round : AddRoundKey

- ***DECHIFFREMENT :***

Le déchiffrement consiste à appliquer les opérations inverses, dans l'ordre inverse et avec des sous-clés également dans l'ordre inverse.

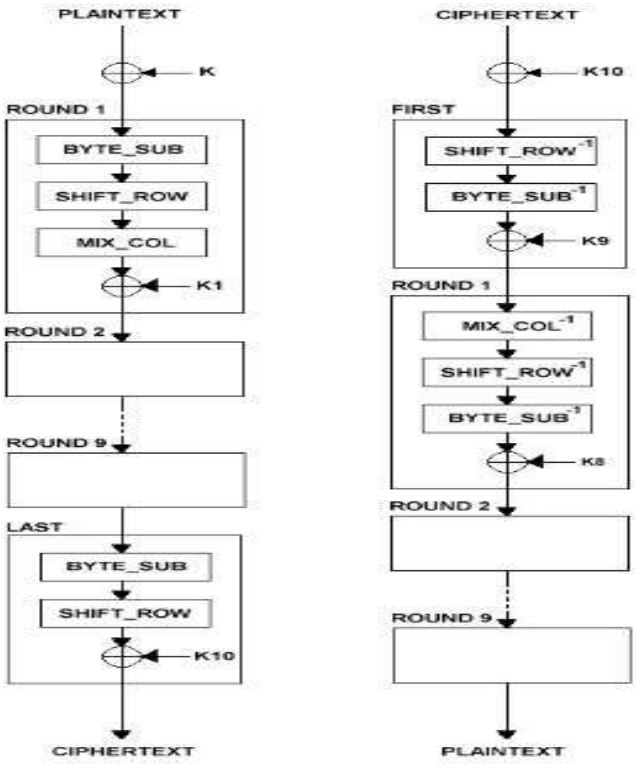


FIGURE 9. DECHIFFREMENT AES

Caractéristiques et Points forts de l'AES :

Le choix de cet algorithme répond à de nombreux critères plus généraux dont nous pouvons citer les suivants :

- Sécurité ou l'effort requis pour une éventuelle cryptanalyse.
- Facilité de calcul : cela entraine une grande rapidité de traitement
- Besoins en ressources et mémoire très faibles
- Flexibilité d'implémentation : cela inclut une grande variété de plateformes et d'applications ainsi que des tailles de clés et de blocs supplémentaires
- Hardware et software : il est possible d'implémenter l'AES aussi bien sous forme logicielle que matérielle (câblé)

Si l'on se réfère à ces critères, on voit que l'AES est également un candidat particulièrement approprié pour les implémentations embarquées qui suivent des règles beaucoup plus strictes en matière de ressources, puissance de calcul, taille mémoire, etc... C'est sans doute cela qui a poussé le monde de la 3G (3ème génération de mobiles) à adopter l'algorithme pour son schéma d'authentification "Millenage".

f- La cryptographie asymétrique (ou à clefs publiques) :

Représentation : Le principe se représente par l'image des boîtes et des cadenas.

Le principe de la cryptographie à clef publique a été introduit par Whitfield Diffie et Martin Hellman en 1976 (après les études de Ralph Merkle sur la méthode du sac à dos). Il utilise une fonction à sens unique (clé publique) avec trappe (clé privée).

- Cela permet de résoudre les problèmes de la cryptographie symétrique.
  - **Inconvénient** : 100 à 1000 fois plus lent que le chiffrement symétrique. Dans la pratique, on mélange les deux (SSL, SSH, PGP par exemple).

La cryptographie à clé publique repose exactement sur ce principe. On dispose d'une fonction **P** qui permet de chiffrer les messages. Ce procédé est inversible, c'est-à-dire que l'on dispose d'une fonction de déchiffrement **S**. On peut fabriquer simultanément un couple (**P, S**), mais connaissant uniquement P, il est impossible (ou au moins très difficile) de retrouver **S**.

- **P** est la clé publique (le cadenas), que vous pouvez révéler à n'importe qui. Si Louis veut vous envoyer un message, il vous transmet P(message).
- **S** est la clé secrète (la clé du cadenas), elle reste en votre seule possession. Vous décodez le message en calculant **S(P(message)) =message**.
- La connaissance de **P** par un tiers ne compromet pas la sécurité de l'envoi des messages codés, puisqu'elle ne permet pas de retrouver **S**. Il est possible de donner librement **P**, qui mérite bien son nom de clé publique.

Bien sûr, il reste une difficulté : comment trouver de telles fonctions **P** et **S** ? Diffie et Hellman n'ont pas eux-mêmes proposé de fonctions satisfaisantes, mais dès 1977, D. Rivest, A. Shamir et L. Adleman trouvent une solution possible, la meilleure et la plus utilisée à ce jour, la cryptographie RSA. Le RSA repose sur la dichotomie suivante :

- Il est facile de fabriquer de grands nombres premiers **p** et **q**
- Étant donné un nombre entier **n = pq** produit de 2 grands nombres premiers, il est très difficile de retrouver les facteurs p et q.

La donnée de **n** est la clé publique : elle suffit pour chiffrer. Pour déchiffrer, il faut connaître **p** et **q**, qui constituent la clé privée. Le problème de factorisation de grands entiers étant très difficile, la connaissance de la clé publique **n** ne permet pas de retrouver les entiers **p** et **q**, qui constituent la clé secrète.

Cela dit, les algorithmes à clé publique sont rarement utilisés pour chiffrer complètement un long message. Ils sont en effet très lents, beaucoup plus lents que leurs homologues symétriques. Pour des applications où il faut échanger de nombreuses données, ils sont inutilisables en pratique. On a alors recours à des cryptosystèmes hybrides. On choisit un algorithme symétrique (AES…) pour l'échange du message. La cryptographie à clé publique est alors utilisée pour l'échange de la clé de ce chiffrement symétrique.

- ***Fonctionnement :***

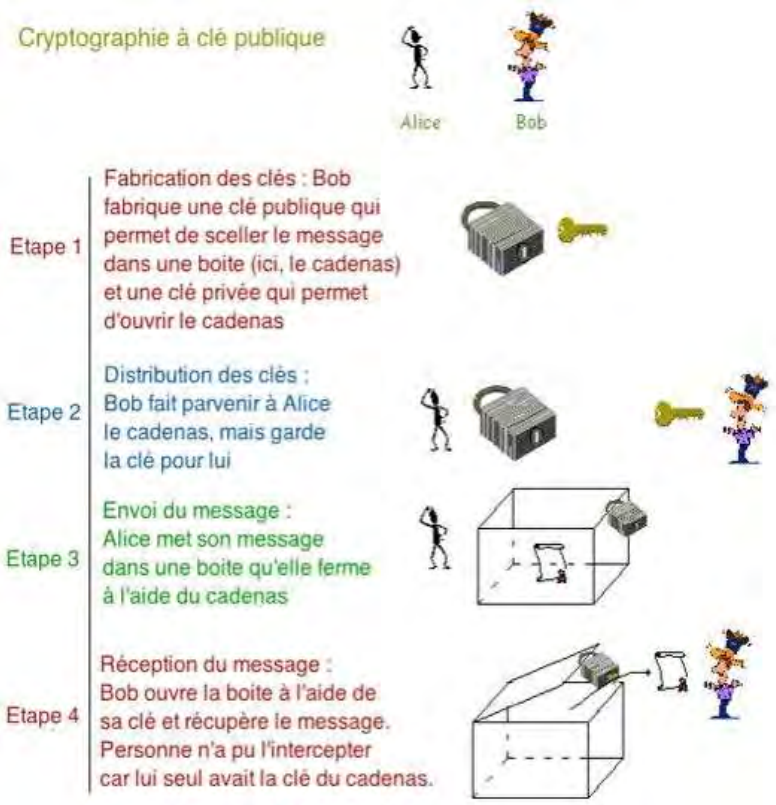


FIGURE 10. CRYPTOGRAPHIE ASYMETRIQUE : FONCTIONNEMENT PRATIQUE

- ***Certificats :***

La cryptographie asymétrique est également utilisée pour générer des- certificats numériques, ceux-ci contenant la clef publique de l'entité associée aux certificats. La clef privée est quant à elle stockée au niveau de cette dernière entité. Une application des certificats est par exemple la mise en œuvre d'une infrastructure à clefs publiques (PKI) pour gérer l'authentification et la signature numérique d'une entité, par exemple un serveur web (Apache avec le module SSL par exemple), ou simplement un client souhaitant signer et chiffrer des informations à l'aide de son certificat de la façon décrite dans les sections précédentes.

- ***Sécurité :***

Un chiffrement symétrique au moyen d'une clef de 128 bits propose 2^{128} (~ 3,4 10^{38}) façons de chiffrer un message. Un pirate qui essaierait de déchiffrer le message par la force brute devrait les essayer une par une.

Pour les systèmes à clef publique, il en va autrement. Tout d'abord les clefs sont plus longues (par exemple 1 024 bits minimum pour RSA) ; en effet, elles possèdent une structure mathématique très particulière (on ne peut pas choisir une suite de bits aléatoire comme clef secrète, par exemple dans le cas du RSA, seuls les nombres premiers sont utilisés). Certains algorithmes exploitant cette structure sont plus efficaces qu'une recherche exhaustive sur, par exemple, 1 024 bits. Ainsi, dans le cas de RSA, le crible général des corps de nombres est une méthode plus efficace que la recherche exhaustive pour la factorisation.

Il faut noter le développement actuel de la cryptographie utilisant les courbes elliptiques, qui permettent (au prix d'une théorie et d'implémentations plus complexes) l'utilisation de clefs nettement plus petites que celles des algorithmes classiques (une taille de 160 bits étant considérée comme très sûre actuellement), pour un niveau de sécurité équivalent.

f-1- Diffie-Hellman : (Système de distribution de clefs secrètes-Secret Sharing)

En cryptographie, l'échange de clés Diffie-Hellman, du nom de ses auteurs Whitfield Diffie et Martin Hellman, est une méthode, publiée en 1976, par laquelle deux correspondants Alice et Bob, peuvent se mettre d'accord sur un nombre (qu'ils peuvent utiliser comme clé pour chiffrer leur communication) sans qu'une tièrce personne appelée Oscar puisse découvrir le nombre, même en ayant écouté tous leurs échanges. Ces deux cryptologues ont transformé le concept des « Puzzles de Merkel » en méthode mathématique. Ils ont d'ailleurs collaboré avec Ralph Merkel pour arriver à construire ce qu'on appelle « l'échange de clés Diffie-Hellman -Diffie Hellman Key Exchange : DHKE ».

Ce système repose sur la difficulté de calculer le logarithme discret sur le sous-groupe multiplicatif d'un corps fini à nombres premiers d`éléments (Corps de Galois GF (p) contenant un nombre (p) premiers d'éléments.

- ***Description mathématique :***

-Rappels sur les primitives :

On appelle racine primitive d'un nombre premier (p), un entier (a) (1<= a <=p-1) tel que la séquence des résidus a^x (mod p) avec (0<= a < p) soit une permutation de

$$(\mathbb{Z}/p\mathbb{Z})^* = (\mathbb{Z}/p\mathbb{Z} - \{0\}) = \{\dot{1}, \dot{2}, \dot{3}, \ldots\ldots, \overline{(p-1)}\}$$

-Définition :

Dans leur article précité « New Directions in Cryptograpy », fondement de la cryptographie à clefs publiques, Diffie et Hellman ont suggéré d'utiliser la difficulté de calculer le logarithme

discret (DLP) dans certains groupes pour créer des primitives cryptographiques, notamment le groupe multiplicatif d'un corps fini (Z/pZ)* où (p) est un nombre premier.

Si p est premier, le sous-groupe ***[(Z/pZ)*,.]*** *est isomorphe au sous-groupe* ***(G,.)*** = ***{a^x / 1≤ a ≤ p-1}*** *avec* ***$a^x \equiv 1 \bmod p$****,* ***a*** *est racine primitive de* ***p*** *et* ***(G,.)*** *est le sous-groupe (cyclique) engendré par* ***x****.*

[(Z/pZ)*,.] ≈ (G,.) = {a^x / 1≤ a ≤ p-1}

$Y = a^x \bmod p \Rightarrow x = \log_a y \bmod p$,

Il est difficile à calculer, pour de grandes valeurs de ***p*** *(1024 bits et plus)*

On appelle fonction de Diffie-Hellman, une fonction de type :

$Y = f(x) = a^x \pmod p$

Où (p) est un nombre premier et (a) une racine primitive de (p).

$X = \log_a y \pmod p$

Remarques : (f) est une fonction à sens unique (One Way Fonction) c'est-à-dire que f est aisée à calculer et f^{-1} est quasi incalculable (pour p suffisamment grand) comme indiqué plus haut (le DLP).

a) f est aisée à calculer :

Il a été démontré que l'opération requiert approximativement ($2\log_2 p$) multiplications.

Ex : pour calculer (a^{37})

De plus, les calculs peuvent se faire sans jamais dépasser la valeur (p^2), ce qui est d'une importance primordiale quand (p) devient grand.

b) f^{-1} est quasi incalculable

Trouver (x) connaissant (a), (p) et (a^x) est très difficile. C'est le problème du logarithme discret qui consiste à trouver x lorsqu'on se donne y.

En effet, on a montré que si chaque opération prend 1µs, le temps nécessaire pour résoudre le problème est donné par :

$e^{[\log p \log(\log p)].0,5} / 10^6$ secondes

Récapitulation :

- **f est à sens unique**

F est « facile » à calculer dans un sens et « difficile » à inverser.

Sauf si on connaît la brèche secrète.

- ***Protocole :***

Parallèlement à leur principe de cryptographie à clé publique, Diffie et Hellman ont proposé un protocole d'échanges de clés totalement sécurisé, basé sur des fonctions difficiles à inverser.

(1) Alice et Bob choisissent **(p)** très grand ; ils choisissent aussi un générateur **(a)** avec **a < p**.

Ils publient **(a)** et **(p)**.

(2) Alice génère une valeur secrète "S_{KA}" avec $0<S_{KA}<p-1$ et Bob une clé secrète "S_{KB}" avec $0<S_{KB}<p-1$.

(3) Ils calculent alors respectivement leurs clefs publiques en utilisant les paramètres (a) et (p) et leurs valeurs privées (S_{KA}) et (S_{KB}).

a. Clef publique d'Alice : $P_{KA}=a^{SKA}(p)$

b. Clef publique de Bob : $P_{KB}=a^{SKB}(p)$

(4) Ils échangent leurs clefs publiques **(P_{KA}, P_{KB})**

(5) Finalement, Alice calcule sa clef secrète :

$$K_{AB}=P_{KB}^{SKA}=(a^{SKB})^{SKA}=a^{SKASKB}(\text{mod } p)$$

Et Bob calcule aussi sa clef secrète :

$$K_{BA}=P_{KA}^{SKB}=(a^{SKA})^{SKB}=a^{SKASKB}\text{mod}(p)$$

Alors, il vient :

$$P_{KA}^{SKB}=P_{KB}^{SKA} \Rightarrow K_{AB}=K_{BA}$$

(6) Alice et Bob ont en définitive la même clef secrète (K) pour protéger leurs informations avec : $K=K_A=K_B$

Remarques :

- $K=K_{AB}=K_{BA}$ peut servir pour le chiffrement par blocs (DES, Triple DES, IDEA, AES…)
- Ce système de Diffie-Hellman est en fait un protocole d'échange de clefs secrètes, le 1er qui est basé sur le logarithme discret.
- Cette technique fondamentale de l'accord de clé est mise en œuvre dans de nombreux protocoles cryptographiques ouverts et commerciaux comme Secure Shell (SSH), Secured Socked Layer / Transport Layer Security (SSL/TLS) et Internet Protocol Security (IPSec).

- ***Vulnérabilité :***

La nature de l'échange de clés Diffie-Hellman le rend plus susceptible aux attaques de l'homme du milieu (HDM), car il n'authentifie aucune des parties prenantes lors de l'échange. La manœuvre HDM peut également créer une paire de clés et usurper les messages entre les deux parties, qui pensent à tort communiquer entre elles. C'est pourquoi l'algorithme Diffie-Hellman est employé conjointement à une autre méthode d'authentification, en règle générale des signatures numériques.

- ***Solution :***

La parade classique à cette attaque consiste à signer les échanges en se basant sur des certificats, les clés publiques étant validées par une Autorité de certification.

Alice peut ainsi être assurée que la clé publique qu'elle reçoit provient effectivement de Bob, et réciproquement pour Bob.

Alice peut ainsi être assurée que la clé qu'elle reçoit provient effectivement de Bob, et réciproquement pour Bob.

- ***Les limites du système :***

Le schéma de Diffie-Hellman, bien qu'astucieux, reste un schéma de principe et souffre d'un inconvénient majeur : il n'assure pas les services de sécurité classiques que sont l'authentification mutuelle des deux intervenants, le contrôle de l'intégrité de la clé et l'anti-rejeu (vérifier qu'une information déjà transmise ne l'est pas à nouveau). L'ennemi peut donc facilement usurper l'identité d'Alice, en remplaçant son élément public par le sien.

f-2- Le système de chiffrement RSA (Rivest-Shamir-Adleman) :

f-2-1- RSA : Principe de fonctionnement et choix des clés :

Le chiffrement RSA (nommé par les initiales de ses trois inventeurs) est un algorithme de cryptographie asymétrique, très utilisé dans le commerce électronique, et plus généralement pour échanger des données confidentielles sur Internet. Cet algorithme a été décrit en 1977 par Ronald Rivest, Adi Shamir et Leonard Adleman. RSA a été breveté par le Massachusetts Institute of Technology (MIT) en 1983 aux États-Unis. Le brevet a expiré le 21 septembre 2000.

- ***Fonctionnement :***

Le chiffrement RSA est souvent utilisé pour communiquer une clé de chiffrement symétrique, qui permet alors de poursuivre l'échange de façon confidentielle : Bob envoie à Alice une clé de chiffrement symétrique qui peut ensuite être utilisée par Alice et Bob pour échanger des données.

Ronald Rivest, Adi Shamir et Leonard Adleman ont publié leur chiffrement en 1978 dans *A Method for Obtaining Digital Signatures and Public-key Cryptosystems*. Ils utilisent les congruences sur les entiers et le petit théorème de Fermat, pour obtenir des fonctions à sens unique, avec brèche secrète (ou porte dérobée).

Tous les calculs se font modulo un nombre entier n qui est le produit de deux nombres premiers. Le petit théorème de Fermat joue un rôle important dans la conception du chiffrement.

Les messages clairs et chiffrés sont des entiers inférieurs à l'entier n^2. Les opérations de chiffrement et de déchiffrement consistent à élever le message à une certaine puissance modulo n (c'est l'opération d'exponentiation modulaire).

La seule description des principes mathématiques sur lesquels repose l'algorithme RSA n'est pas suffisante. Sa mise en œuvre concrète demande de tenir compte d'autres questions qui sont essentielles pour la sécurité. Par exemple le couple (clé privée, clé publique) doit être engendré par un procédé vraiment aléatoire qui, même s'il est connu, ne permet pas de reconstituer la clé privée. Les données chiffrées ne doivent pas être trop courtes, pour que

le déchiffrement demande vraiment un calcul modulaire, et complétées de façon convenable (par exemple par l'Optimal Asymmetric Encryption Padding).

Nombres	Descriptions
p, q	2 grands nombres premiers
n	p*q
φn	(p-1)*(q-1)
e	p,q < e < φn, pgcd(n , e) = 1
d	p,q < d < φn, e*d mod n = 1

FIGURE 11. DESCRIPTIONS DES DONNEES

- ***Création et choix de la clé :***

- *Principe :*

Chaque correspondant doit posséder une paire de clés, l'une doit être privée et l'autre publique, calculée comme suit :

(1) Génération de 2 nombres premiers p et ***q*** *avec* ***φ(n) = (p – 1) (q – 1)***

Rappel : **φ(n) est la fonction d'Euler (nombre entiers positifs < n et premiers à n) :**

- *Si p est un nombre premier,* **φ(p)=p-1**
- *Si n=p*q (avec pet q premiers) :* **φ(n)= φ(p) φ(q)= (p – 1) (q – 1)**

*(2) Calcul de leur produit n= p*q*

Selon le niveau de sécurité souhaité, la taille de n peut varier : 512, 768, 1024, ou 2048 bits

(3) Détermination d'un nombre (e) tel que : ***3 < e < (p – 1) (q – 1) et e Λ (p – 1) (q – 1)***

(4) Calcul d'un nombre (d) tel que ***: ed ≡ 1[(p – 1) (q – 1)] => ed ≡ 1[φ(n)]***

(5) On pose :

Clé publique : P_k = (e, n)

Clé privée : S_k = (d, n)

-n et e sont publics

-p, q, et d restent secrets

(6) Soient X=Y=(Z/nZ) et K={n,p,q,e,d} où X, Y et K représentent respectivement l'espace des clairs, l'espace des cryptos et l'espace des clefs.

On définit les équations de chiffrement et de déchiffrement comme suit :

-Equation de chiffrement :

$E_k : X \rightarrow Y$

$x \rightarrow y$ *(avec x et y le clair et le crypto)*

$\boldsymbol{y = E_k(x) = x^e \pmod n}$

-Equation de déchiffrement :

$D_k : Y \rightarrow X$

$y \rightarrow x = D_k(y)$

$\boldsymbol{D_k(y) = y^d \pmod n = x^{ed} \pmod n = x}$

(6) Exemple :

Un exemple avec de petits nombres premiers (en pratique il faut de très grands nombres premiers) :

1. On choisit deux nombres premiers p = 3, q = 11 ;
2. Leur produit n = 3 × 11 = 33 est le module de chiffrement ;
3. Φ(n) = (3 – 1) × (11 – 1) = 2 × 10 = 20 ;
4. On choisit e= 3 (premier avec 20) comme exposant de chiffrement ;
5. L'exposant de déchiffrement est d = 7, l'inverse de 3 modulo 20 (en effet ed = 3 × 7 ≡ 1 mod 20).

La clé publique d'Alice est (n, e) = (33, 3), et sa clé privée est (n, d) = (33, 7). Bob transmet un message à Alice.

- Chiffrement de x = 4 par Bob avec la clé publique d'Alice : $4^3 \equiv 31 \bmod 33$, le chiffré est y = 31 que Bob transmet à Alice ;
- Déchiffrement de y = 31 par Alice avec sa clé privée : $31^7 \equiv 4 \bmod 33$, Alice retrouve le message initial x = 4.

Le mécanisme de signature par Alice, à l'aide de sa clé privée, est analogue, en échangeant les clés.

- *Justification :*

La démonstration repose sur le petit théorème de Fermat, à savoir que comme **p** et **q** sont deux nombres premiers, si **x** n'est pas un multiple de p on a la première égalité, et la seconde s'il n'est pas un multiple de **q** :

$\boldsymbol{x^{p-1} \equiv 1 \pmod p,\ x^{q-1} \equiv 1 \pmod q.}$

En effet,

$\boldsymbol{y^d \equiv (x^e)^d \equiv x^{ed} \pmod n.}$

Or

ed ≡ 1 (mod (p-1) (q-1))

Ce qui signifie qu'il existe un entier **k** tel que

ed = 1+k(p-1)(q-1)

Donc, si x n'est pas multiple de p d'après le petit théorème de Fermat

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

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

Les deux égalités sont en fait réalisées pour n'importe quel entier x, car si x est un multiple de p, x et toutes ses puissances non nulles sont congrus à 0 modulo p. De même pour q.

L'entier $\mathbf{x^{ed} - x}$ est donc un multiple de p et de q, qui sont premiers distincts, donc de leur produit pq = n (on peut le voir comme une conséquence de l'unicité de la décomposition en facteurs premiers, ou plus directement du lemme de Gauss, sachant que p et q sont premiers entre eux, étant premiers et distincts).

- ***Relation entre la clef privée et la clé publique : (Théorème de l'inversibilité)***

-Rappels :

-Théorème de Fermat

Théorème : Si p est premier et si (a) est un entier qui n'est pas visible par p alors

$$\mathbf{a^{p-1} \equiv 1\ (p)}$$

-Théorème d'Euler

Théorème : Si (a) est un entier premier avec (n) alors

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

où φ(n) = indicatrice d'Euler = nombre d'entiers x tels que x Λ n = 1 ou nombre de nombres premiers avec (n). φ(n) est le nombre d'éléments ayant un inverse modulo n.

Remarque sur la recherche de l'inverse :

-Soit à chercher $a^{-1} \equiv ?\ (n)$.

-On sait que cet inverse existe si a Λ n = 1. On sait alors qu'il existe deux nombres x et y tels que :

ax + xy = 1 (voir th. De Bezout)

⇨ **ax ≡ 1 (n) et x est l'inverse recherché.**

-*Première extension du théorème d'Euler :*

a ≡ k(n) **=> $\mathbf{a^n \equiv k^n\ (m)}$ si a Λ n = 1**

Alors :

$\mathbf{a^{k\varphi(n)} \equiv 1(n)}$ ➔ $\mathbf{a.a^{k\varphi(n)} \equiv a\ (n)}$

⇨ **$\mathbf{a^{1+k\varphi(n)} \equiv a\ (n)}$**

-Deuxième extension du théorème d'Euler :

$a \equiv k(n) \Rightarrow \forall c,\ ac \equiv kc(n)$

Alors :

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

⇨ **$a^{1+k\varphi(n)} \equiv a\ (n)$**

<u>-Relation entre les clefs</u>

Soit x un message clair à chiffrer avec $x \wedge n = 1$

Choisissons la clé publique $P_k = (e, n)$ et la clé privée $S_k = (d, e)$ telles que :

$$x^{k\varphi(n)+1} \equiv x(n)$$

Alors les équations de chiffrement et de déchiffrement sont :

Chiffrement : ***$y = x^e\ (n)$***

Déchiffrement : ***$x = y^d\ (n)$***

Il vient donc => **$x = x^{ed}\ (n)$**

⇨ **$ed = k\varphi(n) + 1 \rightarrow ed \equiv 1\ [\varphi(n)]$**

En outre, on a :

$$D_k(E_k(x)) = ((x)^e \bmod n)^d \bmod n = (x^e)^d \bmod n = x^{ed} \bmod n$$

Il vient alors :

$$x^{ed} = x^{k\varphi(n)+1} = x^{k\varphi(n)}.x \bmod n = 1.x \bmod n = x \bmod n$$

<u>Théorème de l'inversibilité :</u>

Soit un message x tel que $x < n$ et x relativement premier à n. Si le crypto y est le résultat du chiffrement du clair x à l'aide de l'algorithme RSA, alors :

$$[(x^e \bmod n)^d \bmod n] = x$$

Récapitulation :

Soit n = pq où p et q sont premiers. Soit X=Y=(Z/nZ)

On définit K = {(n, p, q, e, d) : n= pq et $ed \equiv 1\ (\varphi(n)$}

$$E_k(x)=x^e \pmod n$$

$$D_k(y)=y^d \pmod n$$

où (x, y) € Z/nZ avec x=clair et y=crypto ; les valeurs P_k=(e, n) sont publiques et celles S_k=(d,e) sont secrètes ou privées.

- ***Implémentation :***

Engendrer les clés :

Il est requis de choisir deux nombres premiers de grande taille, de façon qu'il soit « calculatoirement » impossible de factoriser leur produit.

Pour déterminer un nombre premier de grande taille, on utilise un procédé qui fournit à la demande un entier impair aléatoire d'une taille suffisante, un test de primalité permet de déterminer s'il est ou non premier, et on s'arrête dès qu'un nombre premier est obtenu. Le théorème des nombres premiers assure que l'on trouve un nombre premier au bout d'un nombre raisonnable d'essais.

La méthode demande cependant un test de primalité très rapide. En pratique on utilise un test probabiliste, le test de primalité de Miller-Rabin **[18]** ou une variante. Un tel test ne garantit pas exactement que le nombre soit premier, mais seulement une (très) forte probabilité qu'il le soit.

Chiffrer et Déchiffrer :

Le calcul de $x=y^d \bmod n$ ne peut se faire en calculant d'abord y^d, puis le reste modulo n, car cela demanderait de manipuler des entiers beaucoup trop grands. Il existe des méthodes efficaces pour le calcul de l'exponentiation modulaire.

On peut conserver une forme différente de la clé privée pour permettre un déchiffrement plus rapide à l'aide du théorème des restes chinois.

- ***Applications :***

Lorsque deux personnes souhaitent s'échanger des informations numériques de façon confidentielle, sur Internet par exemple avec le commerce électronique, celles-ci doivent recourir à un mécanisme de chiffrement de ces données numériques. RSA étant un algorithme de chiffrement asymétrique, celui-ci hérite du domaine d'application de ces mécanismes de chiffrement. On citera :

- L'authentification des parties entrant en jeu dans l'échange d'informations chiffrées avec la notion de signature numérique ;
- Le chiffrement des clés symétriques (nettement moins coûteuse en temps de calcul) utilisées lors du reste du processus d'échange d'informations numériques chiffrées.

Ce dernier est en fait intégré dans un mécanisme RSA. En effet, le problème des algorithmes symétriques est qu'il faut être sûr que la clé de chiffrement ne soit divulguée qu'aux personnes qui veulent partager un secret. RSA permet de communiquer cette clé symétrique de manière sûre. Pour ce faire, Alice va tout d'abord choisir une clé symétrique. Voulant échanger un secret avec Bob elle va lui transmettre cette clé symétrique en utilisant RSA. Elle va, pour cela, chiffrer la clé symétrique avec la clé publique (RSA) de Bob, ainsi elle sera sûre que seul Bob pourra déchiffrer cette clé symétrique. Une fois que Bob reçoit le message, il le déchiffre et peut alors utiliser la clé symétrique définie par Alice pour lui envoyer des messages chiffrés que seuls lui et Alice pourront alors déchiffrer.

- ***Signature électronique :***

Après la confidentialité de la transmission d'un message subsiste un problème : son authenticité. Alice voudrait bien envoyer un message ***x*** à Bob de telle façon que celui-ci soit sûr qu'elle est réellement l'émettrice du message, et qu'un intrus ne tente pas de violer la teneur de la communication.

Le système RSA fournit une solution à ce problème :

Rappelons les données :

- Alice seule détient la clé secrète d et diffuse la clé publique (n,e)
- Alice va se servir de la clé publique pour chiffrer le message x

(1) Alice accompagne son message chiffré de sa signature, qui correspond à :
x^d

(2) Bob va donc voir si l'égalité $(x^d)^e \bmod n = x$ est vérifiée. Si c'est le cas, Alice est bien l'émettrice du message.

- ***Sûreté cryptologique du RSA :***
  - *Approches d'attaques :*

Il existe trois approches pour attaquer le RSA :

- La recherche par force brute des clefs, qui est impossible étant donné la taille des données
- L'approche mathématique basée sur la difficulté de calculer **φ(n),** la factorisation du module n
- Les attaques de synchronisation basées sur le déchiffrement

  - *Approche mathématique :*

La factorisation des grands nombres premiers est un problème qui a toujours préoccupé les cryptologues et les mathématiciens : c'est un problème « intraitable » en termes de « Théorie de la complexité ».

Pour une bonne sûreté cryptologique du RSA, il est absolument essentiel de choisir (n) suffisamment grand pour que la factorisation soit impossible (par le décrypteur). Il faut noter que si **φ(n)** ou d étaient connus, (n) pourrait être factorisé aisément. Ceci signifie qu'il est aussi difficile de déterminer **φ(n)** ou d que de déterminer le facteur n.

a) Cas où **φ(n)** est connu :

(i) **N=p*q**
φ(n)=(p-1)*(q-1)=p*q-(p+q)+1
φ(n)=n-(p+q)+1
p+q=1+n-φ(n)

(ii) **$(p-q)^2=(p+q)^2-4pq=(p+q)^2-4n$**
D'où
$(p-q)=\sqrt{([1+n-\varphi(n)]^2-4n)}$
$[1+n-\varphi(n)]^2=1+n^2+\varphi(n)^2+2n-\varphi(n)-n\,\varphi(n)$

$$(p-q)=\sqrt{(1+n^2+\varphi(n)^2-2n-\varphi(n)-n\,\varphi(n)}$$

(iii) $q=1/2([(p+q)-(p-q)])$

$$q=1/2([1+n-\varphi(n)-\sqrt{(1+n^2+\varphi(n)^2-2n-\varphi(n)-n\,\varphi(n))}])$$

et la valeur de p sera:

$$p=1/2([1+n-\varphi(n)+\sqrt{(1+n^2+\varphi(n)^2-2n-\varphi(n)-n\,\varphi(n))}])$$

Dans le cas où φ(n) est connu, on peut donc connaître p et q, et par voie de conséquence retrouver la clef (d).

b) D'autre part : **$ed \equiv 1[\varphi(n)]$**

Si on a **$ed \equiv 1[\varphi(n)] \Rightarrow ed - 1 = k\ ed \equiv 1[\varphi(n)]$,** une méthode de factorisation de (n) est connue si on connaît un multiple de **φ(n).**

Remarques :

A l'heure actuelle, la factorisation connait de lentes améliorations, avec notamment l'optimisation des algorithmes. Excepté un changement dramatique, le RSA-1024 bits restera encore sûr pendant les prochaines années. D'après les projections, une clef de 2048 bits est sensée tenir jusqu'en 2079 si on tient compte de la loi Moore sur l'évolution de la puissance des ordinateurs et de la complexité du matériel informatique. Ces valeurs ne sont correctes que si l'on respecte les propriétés de **e,d,p,** et **q**.

- *Conseils d'utilisations du RSA :*

Pour garantir une bonne sûreté cryptologique, il faut respecter notamment les règles suivantes :

-Ne jamais utiliser de valeur de n trop petite,

-Ne pas utiliser de clefs secrètes trop courtes

-N'utiliser que des clefs fortes (p-1 et q-1 ont un grand facteur premier)

-Ne pas chiffrer de blocs trop courts

-Ne pas utiliser de n commun à plusieurs clefs

-Si (d,n) est compromise ne plus utiliser n

-Ne jamais chiffrer ou authentifier un message provenant d'un tiers sans le modifier

f-2-2- Quelques rappels :

- ***PKCS 1 :***

En cryptographie, PKCS #1 est le premier d'une famille de normes *appelées Normes de cryptographie à clé publique (PKCS)*,publiée par RSA Laboratories. Il fournit les définitions de base et les recommandations pour la mise en œuvre de l'algorithme RSA pour la cryptographie à clé publique. Il définit les propriétés mathématiques des clés publiques et privées, les opérations primitives pour le chiffrement et les signatures, les schémas cryptographiques sécurisés etc.

La version actuelle est 2.2 (2012-10-27). Par rapport à 2.1 (2002-06-14), qui a été réédité sous le nom de RFC 3447, la version 2.2 met à jour la liste des algorithmes de hachage autorisés pour les aligner sur fips 180-4 **[8]**, ajoutant ainsi SHA-224, SHA-512/224 et SHA-512/256.

- Clés

La norme #1 PKCS définit les définitions mathématiques et les propriétés que les clés publiques et privées du RSA doivent avoir. La paire de clés traditionnelle est basée sur un module n, qui est le produit de deux grands nombres distincts de premier choix, p et q, de telle sorte que n=pq. À partir de la version 2.1, cette définition a été généralisée pour permettre des touches multi-prime, où le nombre de nombres premiers distincts peut être de deux ou plus. Lorsqu'il s'agit de touches multi-prime, les principaux facteurs sont généralement étiquetés comme r_i pour certains i, de telle sorte que :

$$\mathbf{n = r_1 . r_2 . \ldots . r_i, \text{ pour } i \geq 2}$$

Comme commodité notationnelle, $p = r_1$ et $q = r_2$.

La clé publique RSA est représentée comme le t-uple (n, e), où l'entier e est l'exposant public.

La clé privée RSA peut avoir deux représentations. La première forme compacte est le tuple (n, d) où d est l'exposant privé. Le deuxième formulaire a au moins cinq termes (p, q, dp, dq, qinv), ou plus pour les touches multi-prime. Bien que mathématiquement redondants à la forme compacte, les termes supplémentaires permettent certaines optimisations computationnelles lors de l'utilisation de la clé. En particulier, le deuxième format permet de tirer la clé publique.

- Primitives

La norme définit plusieurs primitives de base. Les opérations primitives fournissent les instructions fondamentales pour transformer les formules mathématiques brutes en algorithmes computables.

- **I2OSP** - Integer en Octet String Primitive - Convertit un entier non négatif (potentiellement très grand) en une séquence d'octets (chaîne d'octet).
- **OS2IP** - Octet String to Integer Primitive - Interprète une séquence d'octets comme un entier non négatif
- **RSAEP** - RSA Encryption Primitive - Crypte un message à l'aide d'une clé publique
- **RSADP** - RSA Décryptage Primitif - Décrypte le texte chiffré à l'aide d'une clé privée
- **RSASP1** - RSA Signature Primitive 1 - Crée une signature sur un message à l'aide d'une clé privée
- **RSAVP1** - RSA Verification Primitive 1 - Vérifie qu'une signature est pour un message utilisant une clé publique

- Schémas

En elles-mêmes, les opérations primitives n'assurent pas nécessairement aucune sécurité. Le concept d'un schéma cryptographique est de définir des algorithmes de niveau supérieur ou des utilisations des primitifs afin qu'ils atteignent certains objectifs de sécurité.

Il existe deux schémas de cryptage et de décryptage :

- **RSAES-OAEP** basé sur le système optimal de rembourrage de cryptage asymétrique proposé par Mihir Bellare et Phillip Rogaway.

- **RSAES-PKCS1-v1_5** : ancien système de cryptage/décryptage tel que normalisé pour la première fois dans la version 1.5 de PKCS #1.

Remarque : Un petit changement a été apporté au RSAES-OAEP dans la version 2.1 de PKCS #1, ce qui a rendu RSAES-OAEP dans la version 2.0 de PKCS #1 totalement incompatible avec RSA-OAEP dans la version 2.1 de PKCS #1 et la version 2.2.

Il existe également deux schémas de traitement des signatures :

- **RSASSA-PSS :** basé sur le schéma de signature probabiliste inventé à l'origine par Bellare et Rogaway.
- **RSASSA-PKCS1-v1_5**: d'abord normalisé dans la version 1.5 de PKCS #1.

Les deux schémas de signature utilisent des méthodes d'encodage définies séparément :

- EMSA-PSS : schéma probabiliste de signature.
- EMSA-PKCS1-v1_5 : comme d'abord normalisé dans la version 1.5 de PKCS #1.

Les schémas de signature sont en fait des signatures avec annexe, ce qui signifie que plutôt que de signer certaines données d'entrée directement, une fonction de hachage est utilisée d'abord pour produire une représentation intermédiaire des données, puis le résultat du hachage est signé. Cette technique est presque toujours utilisée avec RSA parce que la quantité de données qui peuvent être directement signées est proportionnelle à la taille des touches ; qui est presque toujours beaucoup plus petite que la quantité de données qu'une application peut souhaiter signer.

- ***OAEP : Optimal Assymetric Encryption Padding***

En cryptologie, l'OAEP (Optimal Asymmetric Encryption Padding) est un schéma de remplissage, utilisé généralement avec le chiffrement RSA. L'OAEP est une forme de réseau de Feistel qui nécessite une source d'aléa ainsi que deux fonctions de hachage.

On note :

- ***m le message d'origine (complété ici par des zéros)***
- ***r des données aléatoires***
- ***G,H les deux fonctions de déterministes de générations de masque (elles transforment, via une fonction de hachage, une entrée de longueur quelconque en une sortie de la longueur voulue mais attention il ne s'agit pas d'un générateur de hasard car la sortie générée est fonction de l'entrée fournie).***

Pour les étapes de chiffrement on a :

$X = m \oplus G(r)$
$Y = H(X) \oplus r$
$OAEP(m) = X \,||\, Y$

Où « || » désigne la concaténation et ⊕ l'opérateur « ou exclusif ».

Pour les étapes de déchiffrement/vérification, il suffit de calculer :

$r = H(X) \oplus Y$,

on a alors :

$m = X \oplus G(r)$.

La réversibilité de l'algorithme, permet de vérifier que le des données déchiffrées correspondent bien à celles attendues. C'est généralement une étape du chiffrement RSA, on parle d'ailleurs dans ce cas de RSA-OAEP, la fonction de hachage communément utilisée est le SHA-1.

L'un des intérêts de RSA-OAEP est de pouvoir être prouvé sûr dans un modèle théorique idéalisé, celui de l'oracle aléatoire. Cet algorithme est recommandé par la directive PKCS (Public Key Cryptography Standard) 1, version 2.1 (juin 2002).

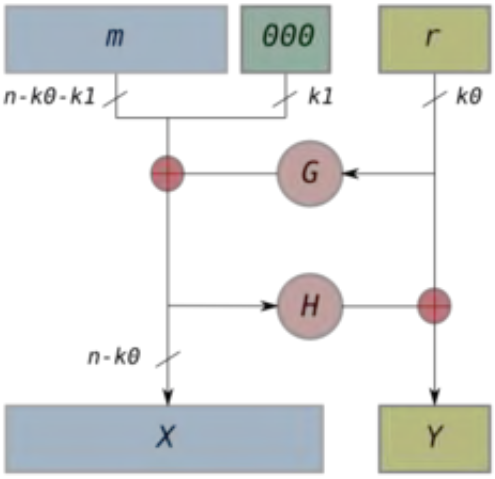


FIGURE 12. DESCRIPTION OAEP

g- *La sécurité des algorithmes :*

En 1994, Lars Knudsen a proposé une classiffication du « cassage » d'un cryptosystème (cela peut s'appliquer au procédé général ou bien une implémentation particulière du procédé cryptographique).

Cette classification est la suivante, et est utile en pratique pour l'évaluation de cryptosystèmes :

1- **« Total Break »** : on est capable de trouver la clef **K** tel que **Y=Dk(X)**
2- **« Déduction globale »** : l'attaquant est capable de trouver un algorithme alternatif pour le déchiffrement sans connaissance de la clef (i.e., l'algorithme produit $D_K(Y)$ sans connaissance de K).
3- **« Déduction locale » (cassage ou craquage d'une instance) :** l'attaquant est capable de trouver le message clair à partir du message chiffré.
4- **« Déduction partielle » :** l'attaquant est capable d'obtenir une information partielle sur la clef ou le message clair. Par exemple, cela peut être la connaissance de quelques bits de la clef.

Un algorithme est dit inconditionnellement sûr ou parfaitement secret, si quel que soit le nombre de messages chiffrés à la disposition du cryptanalyste, il n'y a pas assez d'information pour déduire le message clair. En pratique, les cryptologue s'intéressent aux systèmes qui sont difficiles à casser, même s'ils disposent d'une grande puissance de calcul, il s'agit des algorithmes « calculatoirement » sûrs. La complexité du craquage de l'algorithme s'exprime alors par le nombre d'opérations nécessaire pour le craquer. Actuellement un algorithme nécessitant au moins 2^{80} opérations (élémentaires) est considéré sûr (avec une raisonnable

marge pour le futur). Néanmoins, ce chiffre a tendance à être ré−évaluer à la hausse (disons 2^{85}...).

II-1-1-3- Cryptanalyse :

C'est la technique qui consiste à déduire un texte en clair d'un texte chiffré sans posséder la clé de chiffrement. Le processus par lequel nous tentons de comprendre un message en particulier est appelé *attaque*

Cette technique peut se subdiviser en deux grandes familles d'attaques qui viseront deux cibles différentes :

- *Les Algorithmes : ici l'attaquant essaie de trouver des algorithmes efficaces pour résoudre les problèmes mathématiques sur lesquels reposent les cryptosystèmes.*
- *Les Implémentations matérielles : ici l'attaquant utilisera les fuites d'information lors de l'exécution d'un cryptosystème pour retrouver les secrets.*

Les attaques utilisées se résument comme suit :

- ***Attaque à texte chiffré seul*** (*ciphertext-only* en anglais) : le cryptanalyste possède le texte chiffré. Il peut alors faire des hypothèses sur les messages originaux qu'il ne possède pas. La cryptanalyse est plus ardue pour l'attaquant à cause du manque d'informations.
- ***Attaque à texte clair connu*** (*known-plaintext attack* en anglais) : le cryptanalyste possède des messages ou des parties de messages en clair ainsi que les versions chiffrées. La cryptanalyse linéaire fait partie de cette catégorie.
- ***Attaque à textes clairs choisis*** (CPA ou *chosen-plaintext attack* en anglais) : L'attaquant peut obtenir les chiffrés des textes clairs de son choix. Ceci est toujours réalisé dans un système de chiffrement à clef publique. La cryptanalyse différentielle est un exemple d'attaque à texte clair choisi.
- ***Attaque non – adaptative à textes chiffrés choisis*** (CCA1 ou *chosen-ciphertext attack* en anglais) : Cette attaque a été introduite par M. Naor et M. Yung dans [NY90]. Dans ce cas, l'attaquant dispose, en plus de la clé publique, d'un oracle de déchiffrement qu'il peut utiliser avant qu'on ne lui présente le challenge. Il n'y a plus accès après la connaissance de celui-ci. Cette attaque est appelée parfois *lunchtime attack,* car c'est exactement ce à quoi un adversaire a accès lorsqu'il trouve un ordinateur en session, laissé momentanément sans surveillance.
- ***Attaque adaptative à textes chiffrés choisis (CCA2 :*** *Adaptative−Choosen−ciphertext attack])* : Dans cette attaque, décrite par C. Rackoff et D. Simon, en plus de la clé publique, l'adversaire a accès à l'oracle de déchiffrement avant et après la donnée de challenge. Bien sûr, il ne peut demander le déchiffrement du challenge lui-même.
- ***Attaques parallèles (PA0, PA1)*** : Ces notions, plus techniques, introduites par M. Bellare et A. Sahai, permettent de relier la notion de non-malléabilité à celle d'indistinguabilité. Dans l'attaque parallèle à textes chiffrés choisis, l'attaquant peut demander, après connaissance du challenge, le déchiffrement d'une suite finie de textes chiffrés (ne contenant pas le challenge bien sûr). Les attaques parallèles peuvent être divisées en trois catégories : PA0, PA1 et PA2, selon la possibilité

d'accès à l'oracle de déchiffrement et sans tenir compte du vecteur de chiffrés supplémentaire (jamais, avant, avant et après la réception du challenge).

II-1-1-3-1. Niveau de résistance :

On définit la résistance d'un cryptosystème par rapport à la puissance d'un adversaire. Un cryptosystème est considéré comme sécurisé si un adversaire connaissant tout du système sauf la clef de chiffrement (principe de Kerckhoffs) n'est pas capable de distinguer deux chiffrés (IND).

On parle alors d'indistinguabilité : étant donné deux messages clairs et le chiffré d'un de ces deux messages, l'adversaire ne peut différencier lequel des deux messages a été chiffré.

Suivant la puissance de l'adversaire (par ordre croissant), on dit que le cryptosystème est :

- **IND-COA**

Un cryptosystème est IND-COA si l'indistinguabilité des textes chiffrés résiste à un adversaire qui dispose du texte chiffré de plusieurs messages. Le but de l'adversaire est de déchiffrer les messages voire de retrouver la clé utilisée.

En pratique, on a souvent une connaissance partielle des messages, comme par exemple la langue utilisée.

- **IND-KPA**

Un cryptosystème est IND-KPA si l'indistinguabilité de ses textes chiffres résiste à un adversaire qui dispose de plusieurs paires de messages clairs et chiffrés correspondants.

Le but de l'adversaire est de retrouver la clé ou de déchiffrer d'autres messages chiffrés avec la même clé.

- **IND-CPA**

Un cryptosystème est IND-CPA s'il a la propriété d'indistinguabilité lors d'une attaque à clairs choisis (Chosen Plaintext Attack)

- **IND-CCA**

Un cryptosystème est IND-CCA s'il a la propriété d'indistinguabilité contre une attaque à chiffrés choisis adaptative ou non-adaptative (CCA1 et CCA2).

On distingue deux sous-catégories d'attaque :

- *CCA « pause-déjeuner » : Le principe des attaques CCA1 est que l'adversaire ne dispose que d'une quantité limitée de paires de messages et chiffrés correspondants, car il ne peut plus en demander de nouveau à partir du moment où on lui fournit son challenge.*
- *CCA adaptative : Le principe des attaques CCA2 est que l'adversaire peut adapter ses demandes de déchiffrement au challenge, tant qu'il ne demande pas à déchiffrer le challenge lui-même. Ce genre d'attaque a peu de sens en pratique, mais il est utile pour l'étude formelle et la preuve de sécurité.*

II-1-1-3-2. Familles d'attaques cryptanalytiques :

a- Attaques cryptanalytiques génériques :

Il existe plusieurs familles d'attaques cryptanalytiques, les plus connues étant l'analyse fréquentielle, la cryptanalyse différentielle et la cryptanalyse linéaire.

- *L'analyse fréquentielle :*

L'analyse fréquentielle examine les répétitions des lettres du message chiffré afin de trouver la clé. Elle est inefficace contre les chiffrements modernes tels que DES, RSA. Elle est principalement utilisée contre les chiffrements mono-alphabétiques qui substituent chaque lettre par une autre et qui présentent un biais statistique.

- *L'indice de coïncidence :*

L'indice de coïncidence permet de calculer la probabilité de répétitions des lettres du message chiffré. Il est souvent couplé avec l'analyse fréquentielle. Cela permet de savoir le type de chiffrement d'un message (chiffrement mono-alphabétique ou poly-alphabétique) ainsi que la longueur probable de la clé.

- *L'attaque par mot probable :*

L'attaque par mot probable consiste à supposer l'existence d'un mot probable dans le message chiffré. Il est donc possible d'en déduire la clé du message si le mot choisi est correct. Ce type d'attaque a été mené contre la machine Enigma durant la Seconde Guerre Mondiale.

- *L'attaque par dictionnaire :*

L'attaque par dictionnaire consiste à tester tous les mots d'une liste comme mots clés. Elle est souvent couplée à l'attaque par force brute.

- *L'attaque par force brute :*

L'attaque par force brute consiste à tester toutes les clés possibles. C'est le seul moyen de récupérer la clé, mais elle est très difficile compte tenu de la taille de l'espace des clés, notamment en ce qui concerne les algorithmes modernes comme AES.

- *L'attaque par paradoxe des anniversaires :*

Le paradoxe des anniversaires est un résultat probabiliste qui est utilisé dans les attaques contre les fonctions de hachage. Ce paradoxe permet de donner une borne supérieure de résistance aux collisions de telles fonctions. Cette limite est de l'ordre de la racine de la taille de la sortie, ce qui signifie que, pour un algorithme comme MD5 (empreinte sur 128 bits), trouver une collision quelconque avec 50% de chance nécessite 2^{64} hachages d'entrées distinctes.

b- Attaques modernes :

Dès les années 70 apparaissent les algorithmes de chiffrement modernes par blocs comme DES. Il sera passablement étudié et attaqué ce qui mènera à des attaques majeures dans le monde de la cryptographie. Les méthodes présentées, ci-dessous, ne sont pas vraiment génériques et des modifications sont nécessaires pour attaquer un type de chiffrement donné.
Souvent, on ne s'attaque pas à une version complète de l'algorithme de chiffrement mais une variante avec moins de tours (dans le cas des schémas de type Feistel **[15]** ou les fonctions de hachage). Cette analyse préliminaire, si elle permet de déceler des vulnérabilités, laisse entrevoir une attaque sur l'algorithme complet.

- *Cryptanalyse linéaire :*

La cryptanalyse linéaire, due à Mitsuru Matsui, consiste à faire une approximation linéaire de la structure interne de l'algorithme de chiffrement. Elle remonte à 1993 et s'avère être l'attaque la plus efficace sur DES. Les algorithmes plus récents sont insensibles à cette attaque.

- *Cryptanalyse différentielle :*

La cryptanalyse différentielle est une analyse statistique des changements dans la structure de l'algorithme de chiffrement après avoir légèrement modifié les entrées. Avec un très grand nombre de perturbations, il est possible d'extraire la clé. Cette attaque date de 1990 (présentée à la conférence Crypto 90). Elle est due à Eli Biham et Adi Shamir. Toutefois, on sait maintenant que les concepteurs de DES connaissaient une variante de cette attaque nommée attaque-T. Les algorithmes récents (AES, IDEA, etc.) sont conçus pour résister à ce type d'attaque. Les attaques différentielles sont aussi possibles sur les fonctions de hachage, moyennant des modifications dans la conduite de l'attaque. Une telle attaque a été menée contre MD5.

- *Cryptanalyse différentielle-linéaire :*

Introduite par Martin Hellman et Langford en 1994, la cryptanalyse différentielle-linéaire combine les deux principes. L'attaque différentielle produit une approximation linéaire de l'algorithme. Avec cette attaque, Hellman et Langford ont pu attaquer un DES de 8 rondes avec seulement 512 textes en clair et quelques secondes sur un PC de l'époque. Cette méthode a également été employée pour trouver des clés faibles dans IDEA (International Data Encryption Algorithm). Ce type de cryptanalyse a été améliorée par Eli Biham en 2002.

- *Cryptanalyse X^2 :*

La cryptanalyse X^2, concept dû à Serge Vaudenay, permet d'obtenir des résultats similaires à des attaques linéaires ou différentielles. L'analyse statistique associée permet de s'affranchir des défauts de ces dernières en évitant d'avoir à connaître le fonctionnement exact du chiffrement.

- *Cryptanalyse quadratique :*

La cryptanalyse quadratique **[2]** est une invention récente de Nicolas Courtois et Josef Pieprzyk. Cette attaque (nommée attaque XSL) vise en particulier AES et les autres chiffrements basés sur Rinjdael. L'attaque XSL est le sujet de beaucoup de controverses quant à sa véritable efficacité de par sa nature heuristique. Elle consiste à résoudre un système d'équations de très grande taille.

- *Cryptanalyse modulo n :*

Suggérée par Bruce Schneier, David Wagner et John Kelsey en 1999, cette technique consiste à exploiter les différences de fonctionnement (selon une congruence variable) des algorithmes qui utilisent des rotations binaires.

- *Attaques par canal auxiliaire :*

Les attaques par canaux auxiliaires font partie d'une vaste famille de techniques cryptanalytiques qui exploitent des propriétés inattendues d'un algorithme de cryptographie lors de son implémentation logicielle ou matérielle. En effet, une sécurité « mathématique » ne garantit pas forcément une sécurité lors de l'utilisation en « pratique ». Les attaques portent sur différents paramètres : le temps, le bruit, la consommation électrique, etc.

- *Compromis temps/mémoire :*

Ce concept a été introduit par Martin Hellman en 1980. Il a été amélioré en 1993 par Philippe Oechslin avec le concept de table arc-en-ciel, qui lui a permis par exemple d'attaquer les mots de passe de sessions Windows, lorsqu'ils sont stockés au format LanManager, comme c'est encore le cas le plus souvent. Il s'agit d'un compromis entre une attaque par force brute et l'utilisation de dictionnaires. Une recherche exhaustive nécessite en effet beaucoup de temps alors qu'un dictionnaire de tous les mots de passe possibles nécessiterait énormément de place. Grâce à des procédés algorithmiques, on cherche à trouver un juste milieu entre ces deux principes, en construisant des tables de taille gérable.

- *Attaques sur les modes opératoires :*

Les chiffrements par bloc comme DES ou AES ne peuvent chiffrer que des blocs clairs de taille donnée (128 bits dans le cas d'AES). Pour chiffrer des blocs de données de longueurs plus longues, on utilise des modes opératoires.

Un mode opératoire permet de chiffrer des messages clairs de longueurs quelconques pour obtenir des messages chiffrés de longueurs quelconques. Ils combinent l'algorithme, des opérations simples et des réinjections.

Par exemple, on peut découper les données en blocs de 128 bits et les chiffrer séparément. C'est le cas du mode ECB (Electronic Code Book) qui est vulnérable puisque la présence de deux blocs chiffrés identiques indique que les deux blocs clairs correspondants dans le message clairs original sont également identiques. D'autres modes évitent ce problème mais ne sont pas totalement exempts de vulnérabilités. On utilise alors des vecteurs d'initialisation qui permettent d'éviter la répétition de séquences identiques entre plusieurs messages chiffrés (CBC, CFB, OFB, CTR et GCM)

- *Attaque par rencontre au milieu :*

Chiffrer deux fois avec le même algorithme mais via deux clés différentes est une ouverture à une attaque de type rencontre au milieu. Le chiffrement obtenu n'est pas équivalent à un chiffrement avec une clé deux fois plus longue (on ne passe pas de 2^{56} à 2^{112} dans le cas de DES). Il suffit en effet d'essayer toutes les clés pour déchiffrer la

première étape. On obtient un résultat, toujours chiffré, qui se trouve entre les deux blocs de chiffrement. Ce résultat est soumis à son tour à une recherche exhaustive avec toutes les clés possibles. Au final, la complexité est seulement multipliée par deux. Dans le cas de DES, on obtient une résistance de l'ordre de 2^{57} ; c'est pourquoi on utilise 3DES qui a une complexité finale de 2^{112} opérations (malgré une clé plus longue de 3*56 bits). Grâce à trois chiffrements, chaque sortie du deuxième bloc de chiffrement doit être essayée avec toutes les clés, ce qui augmente considérablement le nombre de possibilités.

- *Attaques sur les systèmes asymétriques :*

Casser un crypto système à clés publique nécessite d'autres approches. Dans le cas de RSA, c'est la difficulté de la factorisation qui assure la résistance du chiffrement. Pour ElGamal, il faut faire face au problème du logarithme discret. Toutefois, certaines failles peuvent apparaître selon l'utilisation que l'on fait de ces algorithmes. RSA est vulnérable si des exposants de faible magnitude sont utilisés (attaques de Don Coppersmith et Wiener). Sous des conditions particulières, un surchiffrement avec RSA peut être attaqué. Le standard PKCS assure une utilisation plus robuste de RSA, même si les premières ébauches du standard étaient sensibles à des attaques par des canaux auxiliaires (Bleichenbacher).

II-1-1-3-3. Autres propriétés analysées :

- *Clefs faibles :*

Certains algorithmes sont susceptibles d'avoir des clés dites faibles. Si une telle clé est utilisée pour chiffrer un message une première fois et que l'on rechiffre le résultat, toujours avec la même clé, alors on obtient le message en clair. Plus formellement, **Ek(Ek(m))=m**. DES possède 4 clés de ce genre. Il y a aussi des clés dites semi-faibles. Dans ce cas, $\mathbf{Ek_1(Ek_2(m)) = m}$.

- *Biais statistiques :*

On peut chercher si la structure de chiffrement produit des biais statistiques. En général, un algorithme de chiffrement est censé produire un résultat proche d'un générateur de nombres aléatoires uniformément distribués, afin de fournir le moins d'informations possible et de maximiser l'entropie. Si un biais est observé (par exemple, on observe plus de bits 1 que de bits 0) alors des analyses supplémentaires peuvent parfois permettre de concevoir une attaque. Citons entre autres des attaques sur RC6 (algorithme de chiffrement par blocs) dont les permutations s'écartent sensiblement des caractéristiques normalement observées dans les générateurs de nombres pseudo-aléatoires

II-1-2- Les différentes techniques d'attaques sur le crypto système à clés publiques RSA :

Depuis 25 ans, la communauté cryptographique mondiale étudie la solidité du RSA face à toute une panoplie d'attaques. Ces attaques se divisent en trois catégories : les attaques qui cherchent à trouver une faille dans les fondements mathématiques mêmes de l'algorithme, les attaques sur

les protocoles construits autour du RSA, enfin, les attaques sur les implémentations du protocole.

Nous pouvons citer :

- l'attaque par la factorisation de grands entiers (module n),

- les attaques élémentaires ;

- les attaques contre un faible exposant privé comme l'attaque de Wiener qui s'appuie sur les fractions continues ;

- les attaques contre un représentant public faible qui se base sur le théorème de Coppersmith comme l'attaque de Hastad ou Broadcast, l'attaque de Franklin-Reiter, etc.

-les attaques sur les implémentations comme le Timing Attacks (attaque par chronométrage), attaque par fautes (Random Faults : Bellcore…), et l'attaque de Bleichenbacher.

Mais avant tout, les attaques sont d'abord classifiées.

II-1-2-1- Classification des attaques :

Tout acte sur un système dont l'intention est de nuire au moins à l'une des propriétés de sécurité est qualifié de malveillant et constitue, de ce fait, une attaque sur ce système. Nous trouvons dans la littérature des manières différentes de classer les attaques.

Certaines taxinomies les organisent en fonction d'un unique critère. Parmi ces critères les plus récurrents sont :

- La *cause* de l'attaque (utilisateur interne ou externe, intrus, etc.)
- Le *mode* ou le *type* de l'attaque (virus, ver, écoute passive, déguisement, etc.)
- Le *résultat* de l'attaque (divulgation, perturbation, etc.)
- La *vulnérabilité* exploitée par l'attaque ; plusieurs aspects peuvent alors être considérés : la phase de création ou d'occurrence de la vulnérabilité (lors de la conception, du développement, de l'exploitation du système, etc.), la nature de la vulnérabilité (concurrence, défaut de validation, etc.)

Il est opportun de remarquer, dès à présent, qu'une même technique d'attaque peut être utilisée pour nuire à différents niveaux d'abstraction du système (voir figure ci-dessous).

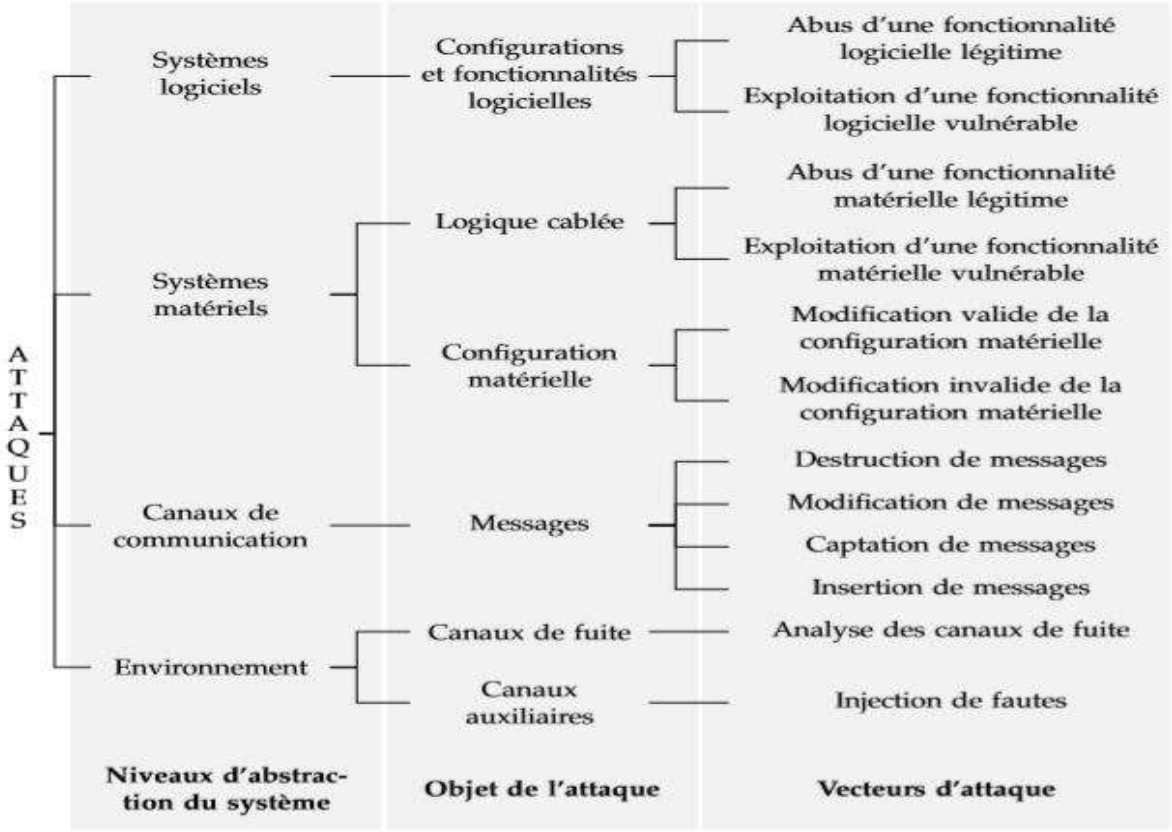


FIGURE 13. CLASSIFICATION DES ATTAQUES SUR LES SYSTEMES INFORMATIQUES

Dans notre étude, les attaques se feront sur le niveau *Canaux de communication.*

II-1-2-2- Stratégies d'attaques du système RSA :

II-1-2-2-1- Attaques physiques :

Il s'agit, dans ce cas, d'attaques se basant sur les propriétés physiques de l'appareil en charge de l'implémentation du protocole :

- ***Attaque sur le temps de calcul :*** L'algorithme de calcul de la fonction RSA est toujours le même. En particulier, il fait intervenir une boucle pour le calcul de $\boldsymbol{y^d mod\ n}$ (ou $\boldsymbol{d}$ est la clé secrète). $\boldsymbol{d} = \boldsymbol{d_1 d_2 d_3} \ldots \boldsymbol{d_n}$, chaque bit $\boldsymbol{di}$ est égal à 0 ou 1. Or dans la boucle de calcul, on trouve une instruction du type '*si* $\boldsymbol{di} = \boldsymbol{1}$ *alors faire tel calcul sinon ne pas le faire*'.
  En mesurant les temps de calcul pour de nombreuses valeurs initiales de y (message crypté), on peut déduire si le calcul qui est fait lorsque $\boldsymbol{di} = \boldsymbol{1}$ a été réellement effectué et de retrouver la clé secrète d bit par bit. On peut contourner ce type d'attaque en masquant les différences de temps de calcul.
- ***Attaque sur la consommation électrique :*** Le même type d'attaque peut être effectué avec la consommation électrique pour chacun des calculs, on mesure la courbe de consommation électrique du composant qui fait le calcul. En analysant la statistique de la consommation électrique à chaque étape du calcul, on déduit de proche en proche la valeur de la clé secrète $\boldsymbol{d}$.
  Des méthodes existent pour fausser la consommation électrique et rendre cette information inutilisable.

- ***Attaque par injections de fautes :*** Cette attaque a été conçue en 1996 et présentée comme un nouveau modèle d'attaque physique sur les cartes à puce : « Cryptanalyse en présence d'erreurs de calcul dans les processeurs », Elle se base sur l'utilisation d'une signature erronée puis de la signature correcte.

II-1-2-2-2 Attaques de protocoles :

Même si RSA est solide, la façon dont nous l'utilisons n'est pas neutre. Par exemple, si on envoie le même message à 3 personnes différentes, chiffrés avec 3 clés RSA de ces personnes, nous pouvons facilement retrouver le message en clair à partir des 3 messages chiffrés en utilisant la propriété de « multiplicativité » de la fonction RSA : $\boldsymbol{f(X*y) = f(x) * f(y)}$.

Il est également risqué de chiffrer plusieurs messages liés au moyen de la même clé publique RSA.

II-1-2-2-3 Problème Mathématique :

La génération des clés RSA s'appuie sur l'utilisation d'un grand nombre $\boldsymbol{n}$ produit de deux grands nombres premiers $\boldsymbol{p}$ et $\boldsymbol{q}$: '$\boldsymbol{n=p*q}$ et $\boldsymbol{e}$ premier avec $(\boldsymbol{p-1}) *(\boldsymbol{q-1})$, $(\boldsymbol{n}, \boldsymbol{e})$ forment la clé publique, et $\boldsymbol{d}$ l'inverse de $\boldsymbol{e}$modulo$(\boldsymbol{p-1}) *(\boldsymbol{q-1})$, la clé privée.

La première façon d'attaquer l'algorithme RSA est de factoriser $\boldsymbol{n}$ et de retrouver $\boldsymbol{p}$ et $\boldsymbol{q}$. En 1999, un nombre de 512 bits a été factorisé avec une puissance de calcul de 104 Mips/an (Soit 10^{10} instructions/s pendant un an).

En prenant la loi de Moore comme référence (La puissance double tous les 18 mois) nous pouvons estimer que les clés de 1024 bits pourront être cassées vers l'an 2010 et que les clés de 2048 bits pourront l'être vers 2030.

D'autres attaques sont possibles à condition de faire des hypothèses sur l'exposant secret d'inverse de $\boldsymbol{emod(p-1) *(q-1)}$. Il est facilement possible de retrouver $\boldsymbol{d}$ à partir de $\boldsymbol{n}$ et $\boldsymbol{e}$ par l'attaque de Wiener lorsque $\boldsymbol{d}$ est inférieur à $n^{1/4}$.

D'autres attaques sont possibles si nous supposons que certains bits de la clé $\boldsymbol{d}$ sont connus si $\boldsymbol{d}$ a une taille de k bits, la connaissance des k/4 bits de poids faibles (les moins significatifs) est suffisant de reconstituer complètement la clé.

II-1-2-3- Attaques de RSA :

II-1-2-3-1 Factorisation de grands entiers :

Un système RSA sécurisé repose sur le fait que la clé privée d soit tenue secrète, mais aussi que les facteurs p et q de n soit inconnus. En effet, si l'on connaît ces deux facteurs, on peut facilement recalculer l'indicatrice d'Euler de n en calculant $\boldsymbol{(p-1)\times(q-1)}$, ce qui nous permet ensuite de retrouver la clé privée d puisque son inverse e est publique. Une des attaques possibles pour casser un système RSA est donc de factoriser l'entier n.

a- Méthode P-1 de Pollard :

Né en 1941, John Pollard est un mathématicien anglais qui a inventé des algorithmes permettant la factorisation de grands entiers ainsi que le calcul des logarithmes discrets.

La méthode P-1 fonctionne au mieux pour un entier composé n ayant un facteur premier p pour lequel p -1 n'a que des petits diviseurs premiers. Il est alors possible de déterminer un multiple m de p – 1 sans connaitre p – 1.

Principe :

Proposition : Soit $n \in N^*$ non premier et p premier un facteur de n. Alors $\forall a \in (Z/nZ)^*$ et $\forall m$ multiple de $p - 1$:

$$\mathbf{a^{p-1} = 1 \pmod p,\ a^m = 1 \pmod p}$$

Principe : D'après les équations ci-dessus, si on trouve un m tel que pour un $a \in (Z/nZ)^*$ fixé : $a^m = 1 \pmod p$ alors on a $p \mid pgcd(a^m - 1, n)$ où p est donc un facteur de n.

Définition : Soit a, $B \in N^*$. On dit que a est B-friable si tous les facteurs premiers de a sont inférieurs ou égaux à B.

Proposition : Soit $n \in N^*$ non premier et p premier un facteur de n. Supposons $p - 1$ B-friable. Posons $\boldsymbol{M = \Pi_{q \le B}\, q^{f_q}}$ (q premier) où $f_q = |\log_q n|$, alors $p \mid pgcd(a^m - 1, n)$.

- ***Implémentation :***

Concrètement, on va calculer de proche en proche les $a^{M(B)}$ pour B croissant jusqu'à pouvoir obtenir un des facteurs de n, le second pouvant se déduire assez facilement par une simple division.

Algorithme : Méthode P-1 Pollard
Entrée : n un entier **Sortie :** g un facteur de n $a \leftarrow$ Un élément de $(Z/nZ)^*$ $B \leftarrow 2$, $M \leftarrow 1$, $g \leftarrow 1$ **Tant que** g = 1 **Faire** M ← M x $B^{fq(B)}$ (mod n) g = *pgcd* (a^M – 1, n) B ← **B + 1** **Fin Tant que** **Renvoyer** g

Remarque : Si lors d'une itération, on trouve un pgcd égal à n, alors il pourrait suffire de seulement changer d'élément a, sans changer la valeur de M pour pouvoir trouver un facteur de n.

b- Méthode Rho de Pollard :

- ***Paradoxe des anniversaires :***

Proposition : Soit $x_1, x_2, \ldots, x_k$ tirés selon une loi uniforme dans un ensemble de cardinal p. Alors pour $k > (\sqrt{[8p \log 2 + 1] + 1}) / 2 \ (\approx 1.18\sqrt{p})$, la probabilité de $x_1, \ldots, x_k$ est inférieure à 0, 5.

Preuve : La probabilité P que x_1, ..., x_k soient distincts est égale à $[p(p-1)...(p-k+1)] / p^k$, c'est à dire $\Pi_{i=0}^{k-1} (1 - i/p)$. On a alors :

$$\mathbf{P \leq \Pi_{i=0}^{k-1} e^{-i/p} = e^{[-k(k-1)]/2p}}$$

On cherche quand est-ce que cette probabilité atteint au moins 0, 5 ce qui reviens à résoudre :

$$\mathbf{e\ [-\ k(k-1)]/2p \leq 1/2}$$

Ce qui nous donne (avec une constante variant en fonction de la probabilité choisie, ici 0, 5) :

$$\boldsymbol{k \approx c \sqrt{p}}$$

- ***Idée de Polard :***

Principe : Soit $n \in N^*$ et p le plus petit facteur premier de n. On tire des x_i aléatoirement dans (Z/nZ), alors avec un nombre d'essai en $O(\sqrt{p})$ on a plus d'une chance sur deux de trouver $x_i = xj \pmod p$ tel que $x_i \neq x_j \pmod n$. Ce qui nous donne donc $p = pgcd(x_i - x_j, n) \neq n$. Cependant calculer un pgcd pour chaque couple (i,j) peut être très couteux.

Suite pseudo-aléatoire : Pour éviter de faire des tirages aléatoires, on va choisir les x_i via une suite récurrente :

$$\mathbf{f : x \dashrightarrow x^2 + 1,\ u_0 \in (Z/nZ)\ u_{n+1} = f(u_n)}$$

Cette suite se comporte un peu comme une suite al´eatoire, au moins en ce qui concerne le "paradoxe des anniversaires".

- ***Détection de cycle de Floyd :***

Proposition : Soit $f \in Z[X]$, $u_0 \in (Z/nZ)$ et $u_k = f^k(u_0)$. Alors s'il existe $i < j$ tel que $x_i = x_j \pmod p$ alors il existe $k < 2j$ tel que $x_k = x_{2k} \pmod p$.

Preuve : Soit $j = i + k$ avec $k > i$, tel que $u_i = u_j$. Donc $\forall k > i$ on a $u_k = u_{k+(j-i)}$. En prenant $k = j - i$ on a donc $u_k = u_{2k}$.

- ***Implémentation :***

On peut donc factoriser N d'un système RSA en trouvant ces facteurs premiers avec l'algorithme suivant :

Algorithme : Rho Pollard
Entrée: N un entier **Sortie:** P un facteur de N X ← 1 Y ← X D ← 1 **Tant que** D = 1 **Faire** X ← $X^2 + 1$ Y ← $(Y^2 + 1)^2 + 1$ D ← pgcd(X − Y, N)

Fin Tant que **Si** D = N **Alors** Changer l'initialisation de X et recommencer **Fin Si** **Renvoyer** D

c- Méthode de Fermat :

Principe : Pour factoriser *n*, la méthode de Fermat consiste à trouver a et b tel que $n = a^2 - b^2$. On aurait donc bien une décomposition en facteurs :

$$\mathbf{n = a^2 - b^2 = (a - b)(a + b)}$$

Puisque *n*, dans un système RSA, est le produit de deux facteurs premiers, ceux-ci sont donc les (a−b) et (a + b). On va donc essayer différentes valeurs de $a > [\sqrt{n}]$ en espérant que la valeur de $a^2 - n$ soit un carré, noté b^2, pour ensuite trouver la factorisation de n.

L'algorithme suivant donne une version de cette méthode.

Algorithme : Méthode de Fermat
Entrée : n un entier **Sortie** : p et q les facteurs de n a ← [√n] b ← 1 **Tant que** (a^2 − n) n'est pas un carré Faire a ← a + 1 **Fin Tant que** b ← √ a^2 − n **Renvoyer** p = (a − b), q = (a + b)

II-1-2-3-2- Attaques élémentaires de RSA :

La cryptanalyse élémentaire sur le système RSA consiste à factoriser le module pour retrouver les valeurs de p et q.

a- Cryptanalyse de RSA connaissant φ(n) :

- ***Proposition :*** *Soit N un module de RSA, Si on connait φ(N), alors nous pouvons factoriser N.*

- ***Démonstration :*** *Supposons que φ(N) est connu. Ainsi, nous disposons d'un système de deux équations en p et q :*

$$\begin{cases} p * q = N \\ p + q = N + 1 - \varphi(N) \end{cases}$$

Qui donnent l'équation en p : $\boldsymbol{p^2 - (N+1-\varphi(N))\, p + N = 0}$

Nous obtenons ainsi : $p = [N+1-\varphi(N)+\sqrt{((N+1-\varphi(N))^2 - 4N)}] / 2$

$$q = [N+1-\varphi(N)+\sqrt{((N+1-\varphi(N))^2 - 4N)}] / 2$$

b- Utilisation du même module et deux exposants différents :

- ***Proposition :*** *Soit N un module RSA. Soient e_1 et e_2 deux exposants premiers entre eux. Si un message clair X est chiffré avec e_1 et e_2, alors nous pouvons calculer X.*

- ***Démonstration :***

Soient Y_1 et Y_2 deux messages chiffrés par :

$$Y_1 \equiv X^{e_1} (mod\ N),$$

$$Y_2 \equiv X^{e_2} (mod\ N),$$

Et supposons que Y_1 et Y_2 sont rendus publiques. Si $pgcd(e_1, e_2) = 1$, alors il existe deux entiers x_1 et x_2 tels que $x_i < ½\ e_i$ vérifiant $e_1x_1 - e_2x_2 = \pm 1$. Ces deux entiers peuvent être déterminer par l'algorithme d'Euclide. D'autre part, ils vérifient :

$$\left|\frac{e_1}{x_1} - \frac{e_2}{x_2}\right| = \frac{|e_1x_1 - e_2x_2|}{x_1e_1} = \frac{1}{x_1e_1} < \frac{1}{2x_1^2}$$

Ainsi x_1 et x_2 peuvent être déterminées comme dénominateur et numérateur de l'une des convergentes de e_1/e_2. Si $e_1x_1 - e_2x_2 = 1$, nous obtenons alors :

$$Y_1^{x_1} Y_2^{-x_2} \equiv X^{e_1x_1} X^{-e_2x_2} \equiv X^{e_1x_1 - e_2x_2} \equiv X\ (mod\ N)$$

Ce qui donne le message X. Si $e_1x_1 - e_2x_2 = -1$, nous calculons $Y_1^{-x_1} Y_2^{x_2}$ et nous obtenons le même résultat.

c- Utilisation de modules différents pour le même message :

Dans cette attaque, on considère qu'un message X est envoyé un certain nombre de fois en utilisant plusieurs clés publiques (N_i, ei). Ce scénario peut se produire si le même message doit être diffusé à plusieurs destinataires. Cette attaque, due à Hastad, est basée sur le théorème des restes chinois.

- ***Théorème chinois des restes chinois :***

Si les entiers N_1, N_2, …, Nk sont deux à deux premiers entre eux, alors le système :

$$\begin{cases} x = a_1 (mod\ N_1) \\ x = a_2 (mod\ N_1) \\ \quad \dots \\ x = a_k (mod\ N_k) \end{cases}$$

Admet une solution unique modulo $\mathbf{N} = \prod^{k}_{i=1} \mathbf{N_i}$. Cette solution est :

$x \equiv \sum^{k}_{i=1} \mathbf{a_i p_i} X_i\ (\mathbf{mod\ N})$,

Avec $\mathbf{p_i} = \mathbf{N}/\mathbf{N_i}$ et $X_i \equiv \mathbf{p_i}^{-1} (\mathbf{mod\ N_i})$,

- ***Démonstration :***

Soit $\boldsymbol{N} = \prod^{k}_{i=1} \boldsymbol{N_i}$, $\boldsymbol{p_i} = \boldsymbol{N}\ /\boldsymbol{N_i}$, et $X_i \equiv \boldsymbol{p_i}^{-1} (\boldsymbol{mod\ N_i})$. Alors, pour chaque $\boldsymbol{i}, \mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{k}$, nous avons $\boldsymbol{p_i M_i} \equiv \mathbf{1}\ (\boldsymbol{mod\ N_i})$ et $\boldsymbol{N_i} | \boldsymbol{p_j}$ si $\boldsymbol{i} \neq \boldsymbol{j}$. Ainsi, avec $x \equiv \sum^{k}_{i=1} \boldsymbol{a_i p_i} X_i\ (\boldsymbol{mod\ N})$, nous avons pour $\mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{k}$, en considérant x $\boldsymbol{modulo\ N_i}$:

$$\boldsymbol{x} \equiv \boldsymbol{a_i p_i} X_i + \sum^{k}_{j \neq i} \boldsymbol{a_j p_j} X_j \equiv \boldsymbol{a_i} + \sum^{k}_{j \neq i} \boldsymbol{a_j p_j} X_i \,.\, \boldsymbol{p_j}\ /\boldsymbol{N_i} \equiv \boldsymbol{a_i}\ (\boldsymbol{mod\ N_i}).$$

Ceci montre que x est la solution du système. Si $\boldsymbol{x' - x \equiv 0\ (mod\ N_i)}$. Puisque les entiers $\boldsymbol{N_i, 1 \leq i \leq k}$, sont premiers entre deux à deux, alors $\boldsymbol{x' - x \equiv 0\ (mod\ N)}$ Ainsi, puisque $|x' - x| < N$, on a $\boldsymbol{x' = x}$, ce qui montre l'unicité de la solution.

II-1-2-3-3- Cryptanalyse de RSA contre un faible exposant privé (fractions continues) :

a- Attaque de Wiener :

L'attaque de Wiener, du nom du cryptologue Michael J. Wiener, est une attaque cryptographique contre le chiffrement RSA, utilisable lorsque d'une part l'exposant privé **d** est faible, et d'autre part les deux nombres premiers secrets **p** et **q** utilisés pour fournir le module de chiffrement public (qui en est le produit normalement difficilement décomposable) sont trop proches.

- *Idée de Wiener :*

Soit (n, e) et (n, d) les clés d'un système RSA. On suppose $\mathbf{q < \sqrt{n} < p}$ et que p et q sont de même taille, ce qui implique $\mathbf{1 < p/q < 2}$.

Par construction on a :

$$\mathbf{ed = 1 \pmod{\phi(n)}}$$

Il existe donc un entier k, (0 < k < d) tel que :

$$\mathbf{ed = 1 + k\phi(n)}$$

De plus on sait par construction que :

$$\mathbf{\phi(n) = (p-1)(q-1) = pq - p - q + 1 = n - (p+q) + 1}$$

Ce qui nous donne en remontant à l'égalité précédente :

$$\mathbf{ed = 1 + k(n - (p+q) + 1)}$$

En divisant l'égalité par nd :

$$\mathbf{e/n = k/d + [(1 + k - k(p+q))] / nd}$$

On a par la suite les inégalités suivantes :

$$\mathbf{| e/n - k/d | < [k(p+q) - k - 1] / nd}$$

$$\mathbf{|e/n - k/d| < [k(p+q)] /nd}$$

$$\mathbf{|e / n - k / d| < [kq((p /q) + 1)] /nd}$$

$$\mathbf{|e/n - k/d | < 3kq /nd}$$

$$\mathbf{|e/n - k/d| < 3k/ d\sqrt{n}}$$

$$\mathbf{|e/n - k/d| < 3/\sqrt{n}}$$

Donc si :

$$d < n^{0,25}/\sqrt{6} < n^{0,25}$$

$$1/2d^2 < 6/2\sqrt{n} = 3\sqrt{n}$$

Ce qui nous donne au final :

$$|e/n - k/d| < 1/2d^2$$

D'après les résultats des fractions continues on peut dire que la fraction k/d est une réduite de e/n.

- *Principe :*

Dans un système RSA, la clé (n, e) est une clé publique, donc accessible par n'importe qui. Il est donc simple de calculer le développement en fraction continue de la fraction e/n.

On peut donc facilement calculer les valeurs k, d des réduites de e/n.

A chaque itération il faut calculer la valeur de l'entier **t = (ed−1) /k** et voir s'il nous permet de trouver les facteurs premiers p et q de n. En effet, on peut trouver une approximation de **ϕ(n) = (ed−1)/t**, ce qui nous permet de retrouver une décomposition de n.

L'attaque de Wiener est donc un algorithme qui, si toutes les conditions sont réunies, de trouver l'exposant privé d en un temps polynomial.

- *Implémentation :*

L'attaque de Wiener peut donc être résumée par l'algorithme suivant.

Algorithme. Attaque de Wiener
Entrée : N un module et e une clé publique d'un système RSA **Sortie :** d la clé privée et p, q les facteurs de N L ← La liste des réduites de e /N Pour chaque réduite x/y de L **Faire** T ← (ey – 1) /x D ← $(N - T + 1)^2 - 4N$ R ← (N -T + 1 - √D) / 2 **Si** pgcd(R, N) > 1 **Alors** **Renvoyer** d=y, p = R, q = N/ R **Fin Si** **Fin Pour**

- *Exemple :*

Soit la clé publique (27962863, 25411171). On suppose que l'indice d est tel que l'on puisse le trouver grâce à l'attaque de Wiener. On commence par calculer les quotients partiels de la fraction e/n ce qui nous donne :

25411171 /27962863 = [0, 1, 9, 1, 23, 7, ...]

Ce qui nous donne les réduites suivantes :

0 1 9/10 10/11 239/263 ...

On peut dans un premier temps éliminer les deux premières solutions. En effet, les deux signifient que d = 1 ce qui voudrait dire e = 1 ce qui n'est pas vérifié.

Avec la fraction 10/11 on trouve pour la première fois un t = (ed−1) /k = 27952288 entier. On a alors :

$$\mathbf{X^2 - (n - t + 1)\,X + n = X^2 - 10576X + 27962863}$$

Les racines de ces polynômes sont 5279 et 5297 et on a finalement la décomposition en deux facteurs premiers p et q de n.

$$\mathbf{n = 5279.5297}$$

II-1-2-3-4- Attaques de RSA contre un faible exposant public :

Pour réduire le temps de chiffrement ou de vérification de la signature, il est courant d'utiliser un petit exposant public e. La plus petite valeur possible pour e est 3, mais pour vaincre certaines attaques, la valeur $e = 2^{16} + 1 = 65537$ est recommandé. Lorsque la valeur $2^{16} + 1$ est utilisée, la vérification de signature nécessite 17 multiplications, par opposition à environ 1000 lorsqu'un $e \leq \varphi(N)$ aléatoire est utilisé. Contrairement à l'attaque de la section précédente, les attaques qui s'appliquent lorsqu'un petit e est utilisé sont loin d'être une rupture totale.

a- Méthode de CopperSmith :

Les attaques les plus puissantes sur RSA à faible exposant public sont basées sur un théorème dû à Coppersmith. Le théorème de Coppersmith a de nombreuses applications, dont seules quelques-unes seront couvertes ici. La preuve utilise l'algorithme de réduction de la base du réseau LLL comme expliqué ci-dessous.
En 1996, D. Coppersmith a décrit une méthode pour déterminer les petites racines modulaires d'un polynôme à une variable ainsi que les petites racines entières d'un polynôme à deux variables.

- **Théorème 1 : (Coppersmith)**

Soit N un entier positif (sans factorisation connue) et f un polynôme de degré d à coefficient dans Z.

$$f(x) = \sum_{i=1}^{n} a_i x^i.$$

C*haque terme x^i est appelé un monôme. La norme Euclidienne de f est définie par :*

$$||f(x)|| = \sqrt{\sum_{i=1}^{n} a_i^2}.$$

Le problème de la petite racine modulaire consiste à trouver un nombre entier x_0 qui vérifie

$$f(x_0) \equiv 0 \bmod (N)$$

$$|x_0| < X,$$

Où X est une borne fixée.

L'idée de Coppersmith s'applique plus généralement dans le cas suivant :

Paramètres connus :

- Un entier N.
- L'existence d'un facteur inconnu b de N avec $b > N^{\beta}$.
- L'expression d'un polynôme $f_b(x)$, de degrés δ.
- Une borne X pour laquelle $f_b(x_0) \equiv 0 \pmod b$ avec $|x_0| < X$.

Paramètres inconnus :
- Le facteur b de N.
- La racine x_0.

La méthode de Coppersmith **[1]** permet de transformer l'équation modulaire $f_b(x_0) \equiv 0 \pmod b$ en une équation entière $f(x) = 0$, c'est à dire sur l'ensemble des entiers. Pour déterminer l'équation entière, on fixe un entier m et on construit une série de polynômes de Z[x] :

$$\mathbf{g_{jk}(x) = x^j (f_b(x))^k N^{m-k}},$$

où les valeurs de j et k dépendent de m et du degré de f. Puisque $f_b(x_0) \equiv 0 \pmod b$, alors $\mathbf{f_b(x_0) = ab}$ pour un entier a et donc

$$\begin{aligned} \mathbf{g_{jk}(x_0)} &= \mathbf{x_0^j\ (f_b(x))^k\ b^{m-k}\ (N/b)^{m-k}} \\ &= \mathbf{a^k x_0^j\ b^k b^{m-k}\ (N/b)^{m-k}} \\ &= \mathbf{a^k x_0^j\ b^m\ (N/b)^{m-k}} \\ &\equiv \mathbf{0 \pmod{b^m}.} \end{aligned}$$

Ceci implique que toute combinaison linéaire h(x) des polynômes $g_{jk}(x)$ vérifiera $h(x_0) \equiv 0 \pmod{b^m}$. Si en plus h vérifie $|h(x_0)| < b^m$, alors on a tout simplement $h(x_0) = 0$, ce qui est facile à résoudre. Le passage de l'équation modulaire $h(x_0) \equiv 0 \pmod{b^m}$ à l'équation entière $h(x_0) = 0$ peut être fixé par le lemme suivant.

- **Lemme :** (Howgrave-Graham).

Soit $h(x) \in Z[x]$ un polynôme de degré d ayant au plus ω monômes et X un nombre positif. Si x_0 est un entier et M un nombre positif tels que :
(1) $|x_0| < X$,
(2) $h(x_0) \equiv 0 \pmod M$,
(3) $||h(xX)|| < M/\sqrt{\omega}$,
alors $h(x_0) = 0$ en tant qu'équation sur Z.

Démonstration. Soit $h(x) = \sum_i^d a_i x^i$ ayant ω monômes. Supposons que $|x_0| < X$.
Alors
(1)

$$|h(x_0)| = \left|\sum_i a_i x^i\right| \le \sum_i |a_i x^i| < \sum_i |a_i X^i|.$$

D'autre part, l'inégalité de Cauchy-Schwarz donne :
(2)

$$\left(\sum_i \alpha_i \beta_i\right)^2 \le \left(\sum_i \alpha_i^2\right)\left(\sum_i \beta_i^2\right).$$

En utilisant cette inégalité, on obtient :

(3)

$$\left(\sum_i |a_i X^i|\right)^2 \le \left(\sum_i 1^2\right)\left(\sum_i (a_i X^i)^2\right) = \omega \sum_i (a_i X^i)^2.$$

Ainsi, si $\|h(xX)\| < M/\sqrt{\omega}$, alors

$$\sum_i |a_i X^i| < \sqrt{\omega}\sqrt{\sum_i (a_i X^i)^2} - \sqrt{\omega}\|h(xX)\| < M.$$

Donc $\|h(x_0)\| < M$. Si on suppose que $h(x_0) \equiv 0 \pmod{M}$, alors $h(x_0) = 0$.

Le problème de la résolution de la congruence $f_b(x_0) \equiv 0 \pmod{b}$ peut donc être résolu par une équation $h(x_0) = 0$ où h vérifie le lemme (Howgrave-Graham). On observe d'abord que la condition (3) de ce théorème signifie que les coefficients de h sont assez petits et que la condition (2) avec $M = b^m$ est satisfaite par toute combinaison linéaire des polynômes $g_{jk}(x)$. Ceci peut être obtenu en appliquant la réduction d'un réseau de dimension n dont une base est formée à l'aide des coefficients des polynômes $g_{jk}(xX)$. Avec l'algorithme LLL, on peut obtenir un vecteur h(xX) qui vérifie la condition (3). L'algorithme LLL donnera alors des vecteurs $v_1(xX), \cdots, v_n(xX)$ où n est la dimension du réseau. On sait d'après le théorème, que le polynôme $v_1(xX)$ va vérifier

(4)
$$\|v_1(xX)\| \leq 2^{\frac{n-1}{4}} \det(L)^{\frac{1}{n}}.$$

Ainsi, pour satisfaire la condition (3), il suffit d'avoir

(5)
$$2^{\frac{n-1}{4}} \det(L)^{\frac{1}{n}} < \frac{b^m}{\sqrt{n}}.$$

Le vecteur recherché h(xX) sera donc le premier vecteur $v_1(xX)$ de la base réduite.

En effet, on considère le réseau formé par les vecteurs colonnes de la matrice définie par les polynômes

$$\mathbf{g_{i,j}(x) = x^j N^i (f_b(x))^{m-i}, j = 0, \cdots, \delta - 1, i = m, \cdots, 1,}$$

$$\mathbf{h_i(x) = x^i (f_b(x))^m, i = 0, \cdots, t - 1,}$$

où $\delta = \deg(f_b)$ et m et t sont des entiers fixés. En explicitant les polynômes $g_{i,j}(x)$, on obtient les valeurs

(6)

	$j=0$ $\downarrow$	$j=1$ $\downarrow$	$j=2$ $\downarrow$	$\cdots$ $\downarrow$	$j=\delta-1$ $\downarrow$
$i=m \rightarrow$	N^m,	$N^m x$,	$N^m x^2$,	$\cdots$	$N^m x^{\delta-1}$,
$i=m-1 \rightarrow$	$N^{m-1} f$	$N^{m-1} x f$	$N^{m-1} x^2 f$	$\cdots$	$N^{m-1} x^{\delta-1} f$
$i=m-2 \rightarrow$	$N^{m-2} f^2$	$N^{m-2} x f^2$	$N^{m-2} x^2 f^2$	$\cdots$	$N^{m-2} x^{\delta-1} f^2$
$\vdots \rightarrow$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$i=2 \rightarrow$	$N^2 f^{m-2}$	$N^2 x f^{m-2}$	$N^2 x^2 f^{m-2}$	$\cdots$	$N^2 x^{\delta-1} f^{m-2}$
$i=1 \rightarrow$	$N f^{m-1}$	$N x f^{m-1}$	$N x^2 f^{m-1}$	$\cdots$	$N x^{\delta-1} f^{m-1}$

On observe de cette façon que les degrés des polynômes sont croissants de la position (i, j) = (m, 0) à la position (i, j) = (1, δ − 1). On peut continuer ce tableau avec les polynômes $h_i(x)$.

$$\mathbf{i = 0, \cdots, t - 1 \Rightarrow f^m, xf^m, x^2 f^m, \cdots, x^{t-1} f^m}$$

Une fois encore les degrés des polynômes sont croissants. Globalement, les degrés remplissent totalement l'intervalle $\{0, 1, \cdots, m\delta + t - 1\}$. On peut alors considérer les matrices définies par les coordonnées des différents polynômes $g_{i,j}(Xx)$ et $h_i(Xx)$, lignes par lignes dans la base $(1, x, x^2, \cdots, x^{m\delta+t-1})$.

(7)

$$
\begin{aligned}
M_m &= \begin{bmatrix} N^m & & & \\ & N^m X & & \\ & & \ddots & \\ & & & N^m X^{\delta-1} \end{bmatrix}, \\
M_{m-1} &= \begin{bmatrix} - & - & - & - & N^{m-1}X^{\delta} & & & \\ - & - & - & - & - & N^{m-1}X^{\delta+1} & & \\ - & - & - & - & - & - & \ddots & \\ - & - & - & - & - & - & - & N^{m-1}X^{2\delta-1} \end{bmatrix}, \\
\vdots &= \vdots \\
M_1 &= \begin{bmatrix} - & - & \cdots & - & NX^{(m-1)\delta} & & & \\ - & - & \cdots & - & - & NX^{(m-1)\delta+1} & & \\ - & - & \cdots & - & - & - & \ddots & \\ - & - & \cdots & - & - & - & - & NX^{(m-1)\delta+\delta-1} \end{bmatrix}, \\
M_0 &= \begin{bmatrix} - & - & \cdots & - & X^{m\delta} & & & \\ - & - & \cdots & - & - & X^{m\delta+1} & & \\ - & - & \cdots & - & - & - & \ddots & \\ - & - & \cdots & - & - & - & - & X^{m\delta+t-1} \end{bmatrix}.
\end{aligned}
$$

Avec ses matrices, on forme la matrice carrée (n×n) M, avec $n = m\delta+t$, qui définit le réseau L par les colonnes de la matrice :

(8)

$$
M = \begin{bmatrix} M_m \\ M_{m-1} \\ \vdots \\ M_1 \\ M_0 \end{bmatrix}
$$

Cette matrice est diagonale et son déterminant est donc le produit des différentes diagonales :

$$\mathbf{det(L) = N^{m\delta} \cdot N^{(m-1)\delta} \cdot \cdots \cdot N^{\delta} X^{1+2+\cdots+n-1} = N^{1/2m(m+1)\delta} X^{1/2\, n(n-1)}}$$

Alors l'inégalité (5) devient :

$$2^{\frac{n-1}{4}} N^{\frac{m(m+1)\delta}{2n}} X^{\frac{1}{2}(n-1)} < \frac{b^m}{\sqrt{n}}.$$

Puisque $b > N^{\beta}$, il suffit alors d'avoir

$$2^{\frac{n-1}{4}} N^{\frac{m(m+1)\delta}{2n}} X^{\frac{1}{2}(n-1)} < \frac{N^{m\beta}}{\sqrt{n}}.$$

En isolant X, on obtient alors :

$$X < 2^{-\frac{1}{2}} n^{\frac{-1}{n-1}} N^{\frac{2mn\beta - m(m+1)\delta}{n(n-1)}}.$$

En considérant l'exposant de N dans l'inégalité précédente, on observe qu'il atteint son maximum pour :

$$m_0 = \frac{2n\beta - \delta}{2\delta},$$

ce qui donne la borne

$$X < 2^{-\frac{1}{2}} n^{\frac{n-1}{n-1}} N^{\frac{\beta^2}{\delta} + \frac{\beta^2}{(n-1)\delta} + \frac{\delta}{4n(n-1)} - \frac{\beta}{n-1}},$$

ce qui peut être écrit sous la forme

$$X < 2^{-\frac{1}{2}} N^{\frac{\beta^2}{\delta} - \varepsilon},$$

avec

$$\varepsilon = \frac{\log n}{(n-1)\log N} + \frac{\beta}{n-1} - \frac{\beta^2}{(n-1)\delta} - \frac{\delta}{4n(n-1)},$$

qui dépend de n mais qui vérifie $\lim_{n \to +\infty} \varepsilon = 0.$

On peut maintenant résumer cette méthode dans le théorème suivant :

Théorème 2 : *Pour tout ε > 0, il existe un entier N_0 qui vérifie la propriété : Soit $N > N_0$ un entier de factorisation inconnue qui a un diviseur $b > N^{\beta}$. Soit $f_b(x)$ un polynôme de degrés δ. Toutes les solutions x_0 de la congruence $f_b(x) \equiv 0 \pmod b$ vérifiant*

$$|x_0| < 2^{-\frac{1}{2}} N^{\frac{\beta^2}{\delta} - \varepsilon}$$

peuvent être déterminées en temps polynômial en log N.

En appliquant ce théorème avec b = N et donc β = 1, on obtient le théorème suivant :

Théorème 3 : *Pour tout ε > 0, il existe un entier N_0 qui vérifie la propriété : Soit $N > N_0$ un entier de factorisation inconnue. Soit f(x) un polynôme de degrés δ. Toutes les solutions x_0 de la congruence $f(x) \equiv 0 \pmod N$ vérifiant*

$$|x_0| < 2^{-\frac{1}{2}} N^{\frac{1}{\delta} - \varepsilon}$$

peuvent être déterminées en temps polynômial en log N.

- ***L'algorithme LLL :***

L'algorithme LLL est lié à la méthode d'orthogonalisation de Gram-Schmidt. Il concerne des bases spéciales, appelées *bases réduites.*

L'algorithme LLL (nommé d'après ses inventeurs L. Lovasz, A. Lenstra et H. Lenstra Jr.) a de nombreuses applications à la fois en théorie des nombres et en cryptographie. Sa découverte en 1982 a fourni un algorithme efficace pour factoriser les polynômes sur les entiers et, plus généralement, sur les anneaux de nombres. LLL est fréquemment utilisé pour attaquer divers cryptosystèmes. Par exemple, de nombreux systèmes cryptographiques basés sur le "problème du sac à dos" ont été brisés en utilisant LLL.

- **Théorème 4 :**

Soit ($b_1, \cdots, b_n$) une base LLL-réduite d'un réseau. Alors pour tout j, $1 \leq j \leq n$, on a

$$\|b_j\| \leq 2^{\frac{n(n-1)}{4(n-j+1)}} (\det L)^{\frac{1}{n-j+1}}.$$

Le temps d'exécution pour LLL est quartique dans la longueur de l'entrée.
Démonstration : Soit ($b_1^*, \cdots, b_n^*$) la base orthogonale associée par la méthode de Gram-Schmidt à ($b_1, \cdots, b_n$). On a, pour $1 \leq j \leq i \leq n$:

$$\|b_j\| \leq 2^{\frac{i-1}{2}} \|b_i^*\|.$$

En appliquant cette inégalité pour chaque i = j, j + 1, ⋯, n et en multipliant, on obtient

$$\|b_j\|^{n-j+1} \leq \prod_{i=j}^{n} 2^{\frac{i-1}{2}} \|b_i^*\|.$$

On obtient alors en commençant à i = 1 :

$$\|b_j\|^{n-j+1} \leq \prod_{i=1}^{n} 2^{\frac{i-1}{2}} \|b_i^*\| = \prod_{i=1}^{n} 2^{\frac{i-1}{2}} \prod_{i=1}^{n} \|b_i^*\| = 2^{\frac{n(n-1)}{4}} \det(L),$$

Ce qui donne le résultat recherché.

On donne maintenant la complexité de l'algorithme LLL.
Théorème : *Soit ($b_1, \cdots, b_n$) une base d'un réseau L et $B = \max_i ||b_i||$. L'algorithme LLL produit une base réduite en un temps polynômial :*

$$\mathcal{O}\left(n^4 \log^3 B\right).$$

- **Factorisation de N :**

Une autre application de la méthode de Coppersmith est la factorisation de $N = pq$ si on connait une fraction importante de p ou de q. Plus précisément, on a le résultat suivant :

Théorème : *Soit $N = pq$ un module RSA dont les facteurs premiers p et q sont inconnus et tels que $q < p$. Si on connait une valeur ~p telle que :*

$$|\tilde{p} - p| < N^{\frac{1}{4}},$$

alors on peut factoriser N en temps polynômial en log N.

Démonstration. Supposons que $|\tilde{p} - p| < N^{1/4}$. On considère la fonction f_p définie par $f_p(x) = x + \tilde{p}$. Alors on a $f_p(p - \tilde{p}) = p \equiv 0 \bmod p$. Ainsi, $x_0 = p - \tilde{p}$ est un entier qui vérifie

$$f_p(x_0) \equiv 0 \mod p, \quad |x_0| < N^{\frac{1}{4}}.$$

On peut donc appliquer le théorème 2 avec $b = p > N^{1/2}$, en prenant $\beta = 1/2$.

b- *Attaque de Hastad (Broadcast) :*

L'attaque de Håstad, aussi connu sous le nom de « Broadcast Attack », l'une des premières attaques découvertes (en 1985), repose sur la possibilité que l'exposant public e soit suffisamment petit. En interceptant le même message envoyé à plusieurs destinataires différents, il est possible de retrouver le message originel à l'aide du théorème des restes chinois.

- *Principe*

Imaginons qu'Alice envoie à 3 destinataires différents ayant respectivement comme clé publique (n_1, e), (n_2, e) et (n_3, e). Supposons que les n_i soient premiers entre eux. On se retrouve alors avec le système d'équations suivant :

$$\begin{cases} x^e = y_1 \pmod{n_1} \\ x^e = y_2 \pmod{n_2} \\ x^e = y_3 \pmod{n_3} \end{cases}$$

Si un attaquant arrive à récupérer les trois chiffrés y_1, y_2 et y_3 alors vu que n_1, n_2, n_3 sont publiques, il peut alors appliquer le théorème des restes chinois et trouve des solutions a vérifiant :

$$\boldsymbol{a = y_1.y_2.y_3 = x^3 \pmod{n_1.n_2.n_3}}$$

*Si l'exposant **e** est suffisamment petit alors on a $\boldsymbol{x < n_1.n_2.n_3}$ et on peut donc retrouver le message clair d'origine avec un calcul de racine cubique de **x**.*

Pour résumer, l'Attaque de Hastad permet de casser un chiffrement RSA quelle que soit la taille des clés, sous réserve que l'on ait intercepté le même message chiffré avec différentes clés publiques utilisant toutes le même exposant publique, et que le nombre de messages interceptés soit au moins égal à cet exposant.

c- *Franklin-Reiter :*

L'attaque Franklin-Reiter s'applique au cas où deux messages liés entre eux par une relation connue, sont chiffrés avec la même clé publique et envoyés à différentes personnes. La complexité de l'attaque est quadratique en e. Elle ne s'applique donc que lorsqu'un petit exposant e est utilisé.

Définition : *Les messages x_1 et x_2 sont liés si :*

– ils vérifient une relation polynomiale connue P de la forme $x_2 = P(x_1)$, $deg(P) = \delta$

– les deux chiffrés correspondant y_1 et y_2 sont connus

Dans ce cas $z - x_1 \pmod N$ est une racine commune des deux équations polynomiales :

$$\boldsymbol{z^e - y_1 = 0 \pmod N \text{ et } (P(z))^e - y_2 = 0 \pmod N}$$

de sorte qu'avec forte probabilité on retrouve x_1 avec :

$$pgcd(z^e - y_1,(P(z))^e - y_2) = z - m_1 \pmod N$$

Le cas limite est le même N pour envoyer le même message x à des utilisateurs différents. Si les clés e_1 et e_2 sont des entiers relativement premiers en utilisant le Théorème Bachet-Bézout il existe u_1, u_2 tels que $1 = u_1e_1 + u_2e_2$. Les chiffrés du x sont $y_1 = x^{e1}$, $y_2 = x^{e2}$ La cryptanalyse de x : $y_1^{u1} + y_2^{u2} = x^{u1e1+u2e2} \equiv x \pmod N$.

Boneh a généralisé les deux attaques Hastad et Franklin-Reiter en supposant que pour chacune des personnes $P_1, \ldots, P_k$, Bob a déterminé, a lié les messages avec un polynôme constant et public $f_i \in Z_{Ni}[X]$. Il a donc diffusé le chiffré de $f_i(x)$ à la personne P_i.

Eve peut obtenir les chiffrés $y_i = f_i(x)^{ei} \pmod{N_i}$ pour $i = 1, \ldots, k$. La nouvelle attaque permet de retrouver x à partir des chiffrés y_i s'il y en a suffisamment.

Théorème. **[version améliorée Boneh]**

Soient $N_1, \ldots, N_k$ des entiers relativement premiers. Soit $N_{min} = min(N_i)$. Soient les polynômes $g_i \in Z_{Ni}[x]$ de degré maximum $\delta \leq k$. S'il existe un unique $x_0 < N_{min}$ tel que $g_i(x_0) \equiv 0 \pmod{N_i}$ pour chaque $i = 1, \cdots, k$, alors on peut retrouver efficacement x_0 à partir des N_i et des polynômes g_i.

II-1-2-3-5- Attaques sur les implémentations de RSA :

Nous allons d'abord nous intéresser aux attaques concernant les implémentations de RSA. Ces attaques montrent qu'une implémentation de RSA ne doit pas seulement se protéger des attaques mathématiques du chiffrement RSA.

a- Attaques par fautes (Random Faults) :

C'est en septembre 1996 que trois chercheurs de Bellcore, Dan Boneh, Richard DeMillo et Richard Lipton, proposent un nouveau modèle d'attaque physique sur les cartes à microprocesseur, qu'ils baptisent « cryptanalysis in the presence of hardware faults ». Ce modèle d'attaque est alors dirigé contre plusieurs algorithmes cryptographiques à clé publique : le système RSA et les schémas d'authentification de Fiat-Shamir et de Schnorr.

Illustrons l'attaque dans le cas où l'algorithme RSA est implémenté en utilisant la technique des « restes chinois » (qui présente l'intérêt d'accélérer les calculs d'un facteur environ 4 par rapport à une implémentation classique). Pour le calcul de $\boldsymbol{x = y^d \bmod n}$, l'idée est de calculer séparément $\boldsymbol{x_p = x \bmod p}$ et $\boldsymbol{x_q = x \bmod q}$, puis de reconstituer x à partir de x_p et x_q en utilisant le théorème des restes chinois. Pour cela, on suppose que sont stockées dans la carte à puce les valeurs $\boldsymbol{d_p = d \bmod (p-1)}$ et $\boldsymbol{d_q = d \bmod (q-1)}$. Les valeurs x_p et x_q sont calculées au moyen des formules suivantes : $\boldsymbol{x_p = y^{dp}{}_p \bmod p}$ et $\boldsymbol{x_q = y^{dq}{}_q \bmod q}$, où $\boldsymbol{y_p = y \bmod p}$ et $\boldsymbol{y_q = y \bmod q}$.

L'attaquant lance alors le calcul deux fois. La première fois sans introduire de perturbation. Il obtient alors les bonnes valeurs pour $\boldsymbol{x_p}$ et $\boldsymbol{x_q}$ et le théorème des restes chinois donne la valeur correcte x. La seconde fois, il perturbe le calcul (par exemple au moyen de rayonnements électromagnétiques ou en faisant varier l'alimentation électrique...). Il obtient par exemple la bonne valeur pour $\boldsymbol{x_p}$, mais une valeur erronée (disons $\boldsymbol{x'_q}$) pour $\boldsymbol{x_q}$.

Le théorème des restes chinois donne alors une valeur ***x'*** erronée, qui vérifie ***x' ≡ x mod p***, mais ***x' ≠ x mod q***. Cela montre que ***x' – x*** est divisible par p mais pas par q. On peut alors calculer ***PGCD(x' – x, n)***, qui est alors égal à p. La factorisation de n est donc trouvée, et le système est cassé !

Notons qu'un moyen simple d'éviter ce type d'attaque consiste à vérifier systématiquement le calcul avant de sortir le résultat. Pour le RSA, c'est particulièrement commode, puisqu'il suffit par exemple de s'assurer que $x^e = y \bmod n$.

b- *Attaque de Bleichenbacher (attaque de texte chiffré adaptative) :*

Soit N un module RSA à n bits et X un message à m bits avec m<n. Avant d'appliquer le cryptage RSA, il est naturel de compléter le message X à n bits en y ajoutant des bits aléatoires. Une ancienne version d'un standard connu sous le nom de Public Key Cryptography Standard 1 (PKCS #1) utilise cette approche.
Après le remplissage, le message se présente comme suit :

02	**Random**	**00**	**X**

Le message résultant a une longueur de n bits et est directement chiffré à l'aide de RSA. Le bloc initial contenant « 02 » a une longueur de 16 bits et est là pour indiquer qu'une marge (pad) aléatoire a été ajouté au message. Lorsqu'un message PKCS#1 est reçu par la machine de Bob, une application (par exemple, un navigateur Web) le déchiffre, vérifie le bloc initial et les bandes du pavé aléatoire. Cependant, certaines applications vérifient le bloc initial « 02 » et s'il n'est pas présent, elles renvoient un message d'erreur indiquant « invalid ciphertext ». Bleichenbacher a montré que ce message d'erreur peut avoir des conséquences désastreuses : en utilisant le message d'erreur, un attaquant peut déchiffrer les textes chiffrés de son choix.
Supposons que l'attaquant intercepte un texte chiffré Y destiné à Bob et veuille le déchiffrer. Pour monter l'attaque, Marvin choisit un $r \in Z_N^*$ aléatoire, calcule ***Y'= rY mod N*** et envoie ***Y'*** à la machine de Bob. Une application exécutée sur la machine de Bob reçoit ***Y'*** et tente de le déchiffrer. Il répond par un message d'erreur ou ne répond pas du tout (si ***Y'*** est correctement formaté). Ainsi, l'attaquant apprend si les 16 bits les plus significatifs du décryptage de ***Y'*** sont égaux à **02**. En effet, l'attaquant a un oracle qui teste pour lui si les 16bits les plus significatifs du décryptage de ***rY mod N*** sont égaux à ***02***, pour tout r de son choix. Bleichenbacher a montré qu'un tel oracle est suffisant pour décrypter Y.

c- *Attaque par chronométrage (Timing Attacks) :*

L'idée consiste à étudier le temps nécessaire à l'ordinateur, stockant la clé privée, de déchiffrer (ou de signer) plusieurs messages. Cette attaque se base sur le fait que la plupart des implémentations utilisent un même algorithme (ou alors un algorithme connu) afin d'effectuer le déchiffrement, et on peut donc en déduire le nombre d'opérations effectuées et ainsi petit à petit récupérer des informations sur d. Par exemple, il est courant d'utiliser l'exponentiation modulaire pour implémenter notre fonction de déchiffrement, comme nous avons vu précédemment, cependant une amélioration de cette dernière se base sur la représentation binaire de la clé (et donc de d), ce qui nous permet après plusieurs opérations de déchiffrement de faire des analyses statistiques sur les informations recueillies pour déterminer d. Or, en général, une amélioration en temps est souvent cruciale en cryptographie, ceci est donc largement utilisé.

Tout d'abord, regardons l'amélioration de l'exponentiation modulaire :

*Soit **d** notre exposant dans l'expression **f'(x)=x^d mod n** avec **x** notre message chiffré. On peut écrire **d**, sous forme de représentation binaire :*

$$d = \sum_{i=0}^{b-1} a_i 2^i$$

*avec **a** représentant un bit (soit 0 soit 1), et **b** le nombre de bit pour représenter **d**.*

*On a donc **x^d** qu'on représente ainsi :*

$$x^d = \prod_{i=0}^{b-1} (x^{2^i})^{a_i}$$

Cette représentation binaire permet de faire des opérations extrêmement rapides dans la plupart des langages de programmation grâce aux opérateurs bit à bit, en C par exemple on a les opérateurs >> et << pour effectuer des décalages (ou shift en anglais) sur des nombres binaires (ceci offre notamment un gain énorme de temps sur des opérations comme les puissances).

L'attaque par chronométrage consistera dans notre cas, à observer le temps que met l'ordinateur pour déchiffrer un certain message afin de trouver petit à petit chaque bit de d. Tout d'abord, d par définition est forcément impair, on conclut donc que le bit 0 de d sera d_0=1 (plus d'infos : bit de poids faible). Pour trouver les autres bits, on va émettre des hypothèses sur la valeur de a_i, qui peut être soit 1, soit 0 (a_i n'est autre que le bit i de d). S'il est égal à 0, le résultat de $(b^{2i})^{ai}$ sera forcément 1, et l'opération sera alors bien plus rapide et différente en termes de temps qu'avec a_i=1, ce qui nous donne des informations sur des bits de d. Il est possible d'utiliser ce principe afin de découvrir d en entier, simplement en demandant à l'ordinateur de déchiffrer des messages bien spécifiques.

Cette attaque ne s'applique pas uniquement à RSA, et peut être un aspect important de la sécurité d'une implémentation. Pour s'en protéger, on peut par exemple effectuer des délais dans le programme afin d'avoir un temps fixe pour chaque opération nécessaire, ou encore utiliser une technique d'aveuglement.

Pour cette technique, avant de déchiffrer le message x, l'ordinateur va prendre au hasard un nombre entier r et calculer $x' = x \times r^e \bmod n$, puis faire $y' = x'^d \bmod n$, et enfin $y = y' \cdot r^{-1} \bmod n$. Ces opérations sont en réalité un simple chiffrement/déchiffrement, mais en utilisant une variable intermédiaire r qui rend alors impossible l'attaque par chronométrage car r est choisi aléatoirement par l'ordinateur.

Sachez qu'il y a des attaques dans la même idée, mais se basant cette fois sur la consommation électrique de l'ordinateur qui peut varier en fonction des opérations effectuées lors du déchiffrement.

II-1-2-3-6- <u>Conclusion :</u>

Deux décennies de recherche sur l'inversion de la fonction RSA ont produit des attaques perspicaces, mais aucune attaque dévastatrice n'a jamais été trouvée. Les attaques découvertes jusqu'à présent illustrent principalement les écueils à éviter lors de la mise en œuvre de RSA. À l'heure actuelle, il semble qu'une mise en œuvre appropriée puisse être fiable pour assurer la sécurité dans le monde numérique. Ces dernières attaques illustrent qu'une étude de la structure mathématique sous-jacente est insuffisante. Il est aussi important de remarquer que les

nombreuses attaques peuvent être vaincues en rembourrant correctement le message avant le chiffrement ou en le signant.

II-2- Les méthodes de collecte des informations :

Il existe de nombreuses méthodes différentes pour recueillir des données.

TABLEAU 1. OPTIONS DE COLLECTE (DONNEES PRIMAIRES) ET DE REGROUPEMENT (DONNEES SECONDAIRES) DE DONNEES

Option	Sources et Méthodes	Exemples
Recherche de documents et de données existants	• Documents officiels relatifs aux politiques, plans de mise en œuvre du programme et rapports • Statistiques officielles • Données de suivi du programme • Dossiers du programme	• Examen des documents relatifs à la planification du programme, comptes rendus de réunions, rapports d'étape • Situation politique, socioéconomique et/ou sanitaire du pays ou du lieu spécifique dans lequel le programme a été mis en œuvre
Collecte de données auprès de groupes ou d'individus	• Entretiens avec des informateurs clés, individus, groupes, groupes de discussion, méthodes projectives • Questionnaires ou enquêtes : par courriel, sur Internet, en face à face, données mobiles • Méthodes spécialisées (p. ex., dotmocracy [méthode participative d'aide à la décision], classement hiérarchique par cartes, calendriers saisonniers, méthodes projectives, histoires)	• Entretiens d'informateurs clés avec des représentants des ministères compétents, organisations non gouvernementales et/ou la communauté de développement au sens large • Entretiens avec des responsables, agents d'exécution et personnes chargées du suivi régulier du programme • Entretiens, discussions de groupe (telles que des groupes de discussion) et/ou questionnaires auprès de participants au programme
Observation	• Structurée ou non structurée • Avec ou sans participants	• Observations des activités du programme et des interactions avec les participants

	• Participative ou non participative • Enregistrée à l'aide de notes, photos ou vidéos	
Mesures physiques	• Mesures biophysiques • Informations géographiques	• Poids des nourrissons • Sites fortement touchés par l'infection à VIH

La collecte des données utiles pour réaliser cette étude s'est faite autour d'une seule technique parmi celles citées plus haut. Il s'agit de la recherche documentaire.

✓ **La recherche documentaire :**

La recherche documentaire est une méthode très importante dans le cadre d'une étude scientifique et est « nécessaire pour l'exploration du sujet ».
En effet, la recherche documentaire ou l'étude documentaire est un ensemble d'actions, de méthodes et de procédures ayant pour objet de retrouver dans des fonds documentaires, les références des documents pertinents.
Dans le cadre de cette étude, elle nous a permis de mieux comprendre la portée de la réalisation de ce projet, l'explication des concepts utilisés et d'affiner notre problématique.
Pour ce qui concerne les outils numériques, notamment la Collection Microsoft, elle nous a aidé à accéder à quelques définitions quoi que générales et certains articles menés sur l'informatisation.
Le dernier pan de l'étude documentaire a concerné les outils virtuels qui ont d'ailleurs constitué l'essentiel de notre documentation spécialisée.
A travers des sites Internet spécialisés, nous avons pu consulter des mémoires et articles dans leur intégralité sur RSA.

Outils numériques : généralement constitués des dictionnaires et encyclopédies

Outils virtuels : toute la documentation accessible via Internet.

II-3-Les méthodes d'analyse des résultats :

II-3-1- Portée et Objet :

L'analyse des données est le processus qui consiste à examiner et à interpréter des données afin d'élaborer des réponses à des questions. Les principales étapes du processus d'analyse consistent à cerner les sujets d'analyse, à déterminer la disponibilité de données appropriées, à décider des méthodes qu'il y a lieu d'utiliser pour répondre aux questions d'intérêt, à appliquer les méthodes et à évaluer, résumer et communiquer les résultats.

Les résultats analytiques soulignent l'utilité des sources de données en jetant de la lumière sur les sujets pertinents. L'analyse des données joue également un rôle clé dans le processus d'évaluation de la qualité des données en indiquant les problèmes liés à la qualité des données dans une enquête particulière. Ainsi, l'analyse peut influer sur les améliorations futures au processus d'enquête.

L'analyse des données est essentielle pour comprendre les résultats des études pour obtenir des renseignements sur les lacunes en matière de données, pour concevoir et remanier les enquêtes, pour planifier et pour formuler des objectifs en matière de qualité.

II-3-2- Principes :

L'analyse est le principal outil permettant d'obtenir de l'information à partir des données. Les données d'une enquête peuvent être utilisées à des fins d'études descriptives ou analytiques. Les études analytiques peuvent servir à expliquer le comportement de caractéristiques ou les relations entre elles.

Pour être efficace, l'analyste doit comprendre les questions pertinentes (tant celles qui sont actuelles que celles qui sont susceptibles d'émerger à l'avenir) et comment présenter les résultats au public. L'étude du contexte de l'analyse permet à l'analyste de choisir les sources de données et les méthodes statistiques appropriées. Toutes les conclusions présentées dans une analyse, y compris celles qui peuvent avoir une incidence sur les politiques publiques, doivent être appuyées par les données analysées.

Avant de procéder à une étude analytique, il faut se pencher sur les questions suivantes :

- Objectifs : Quels sont les objectifs de cette analyse ? Quel est le sujet abordé ? Quelles sont la (ou les) question(s) auxquelles il s'agit de trouver une réponse ?
- Justification : Pourquoi cette question est-elle intéressante ? Comment ces réponses contribueront-elles à la somme des connaissances existantes ? Quelle est la pertinence de cette étude ?
- Données : Quelles données sont utilisées ? Quelle est la meilleure source de données pour cette analyse ? Y a-t-il des limites ?
- Méthodes d'analyse : Quelles techniques sont appropriées ? Permettront-elles d'atteindre les objectifs ?
- Public : Qui s'intéresse à cette question, et pourquoi ?

II-3-3- Interprétation des résultats

En étudiant les changements survenus au fil du temps, veiller à examiner les tendances à court terme en considérant également les tendances à moyenne et à long terme. Les tendances à court terme ne représentent souvent que de légères fluctuations d'une tendance plus importante à moyen ou à long terme.

II-3-4- Présentation des résultats

- Mettre l'accent sur les variables et les sujets importants. Lorsque le sujet abordé est trop vaste, l'impact principal du message se trouve souvent atténué.
- Structurer les idées de façon logique, en fonction de leur pertinence ou de leur importance.
- Rédiger le texte en langage aussi simple que le sujet le permet.

- Insérer des graphiques en complément du texte et des tableaux pour communiquer le message. Toujours commenter l'information fournie dans les tableaux et les graphiques afin de permettre une meilleure compréhension.
- Lorsque des tableaux sont insérés, la présentation générale doit contribuer à la clarté des données qu'ils contiennent et prévenir les erreurs d'interprétation. Cela comprend l'espacement, la formulation, l'emplacement et l'apparence des titres, les titres de lignes et de colonnes etc.
- Fournir des renseignements sur les sources de données utilisées ainsi que toutes lacunes dans les données ayant pu avoir une incidence sur l'analyse.
- Fournir des renseignements sur les méthodes analytiques et les outils utilisés.
- S'assurer que toutes les références sont exactes, uniformes et font l'objet de renvois dans le texte.
- S'assurer qu'il n'y a pas d'erreurs dans le document. Vérifier les détails, par exemple la cohérence des chiffres, les tableaux et les graphiques, ainsi que l'exactitude des données externes et des calculs arithmétiques simples.
- S'assurer que ce qui est annoncé dans l'introduction est effectivement exprimé dans le reste du document. S'assurer que les conclusions sont cohérentes avec les résultats de l'analyse.
- Faire réviser le document par d'autres personnes pour en vérifier la pertinence, l'exactitude et l'intelligibilité, peu importe où il doit être diffusé.
- Comme bonne pratique, envisager de présenter les résultats à des pairs avant de mettre la dernière main au texte.

II-3-5- Indicateurs de qualités :

Principaux éléments de la qualité *: pertinence, intelligibilité, exactitude, accessibilité*

Un produit analytique est pertinent s'il y a un public qui s'intéresse (ou qui s'intéressera) aux résultats de l'étude.

Pour que le degré d'intelligibilité d'un document analytique soit élevé, le style de rédaction doit être adapté au public cible. En outre, l'article doit fournir suffisamment de détails pour permettre à une autre personne à laquelle l'accès aux données serait accordé de reproduire les résultats.

Pour qu'un produit analytique soit exact, il faut utiliser les méthodes et les outils appropriés pour produire les résultats.

Pour qu'un produit analytique soit accessible, il doit être mis à la disposition des personnes auxquelles les résultats de la recherche seraient utiles.

II-4- Simulation de l'attaque par factorisation :

Cette partie va être consacrée aux aspects pratiques pour la réalisation de notre application « RSA Attack Simulator » (Simulation de l'attaque sur RSA). Son but étant de simuler une attaque par factorisation sur le module de RSA lors des transferts des messages, puis d'examiner le problème lié à l'intégrité et à l'authentification du message.

II-4-1- Ressources utilisées :

II-4-1-1- Ressources Matérielles :

- Processeur Intel CORE™ i5
- Une mémoire vive de 12Go

II-4-1-2- Ressources Logicielles :

- Netbeans 8.2
- Langage de programmation : Java Swing
- Système d'exploitation : Windows 10

II-4-2- Conception :

Le but principal de notre application est de faire une simulation de l'attaque sur RSA par la méthode de factorisation, pour sa réalisation nous passerons par plusieurs étapes :

- Génération des clefs publiques et privées pour Alice et Bob.
- Chiffrement du message lors de son transfert
- Déchiffrement du message durant l'envoi
- Attaque sur le module n de RSA afin de retrouver les deux facteurs p et q qui permet de calculer la clef privée.
- Interception et déchiffrement des messages transmis entre Alice et Bob.

II-4-3- Présentation de l'application :

a- Objectif :

L'objectif de notre travail est de réaliser une application de simulation d'une attaque par factorisation sur le système cryptographique RSA.

b- Description de l'application :

Notre application simule l'attaque par factorisation sur le système cryptographique RSA, en mettant en point plusieurs scénarios et cas possibles.

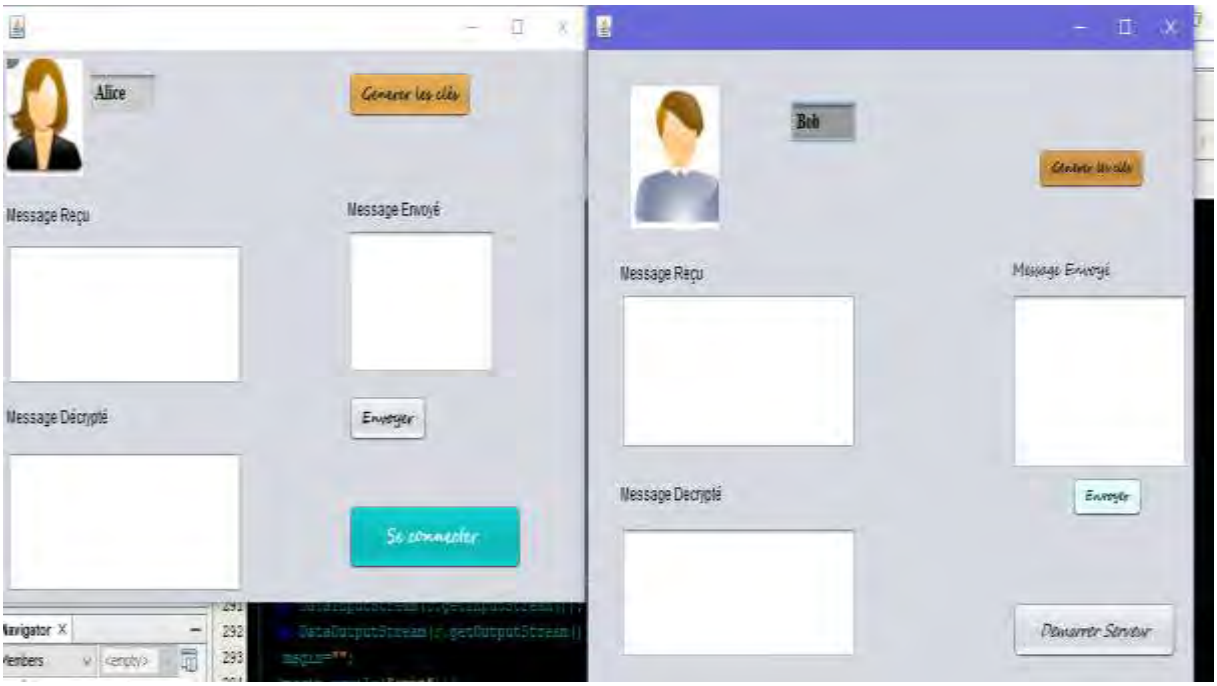


FIGURE 14. APPLICATION AU PREMIER LANCEMENT

c- ***Scénarios de l'utilisation :***

1- Pour la communication entre Alice et Bob, il est nécessaire de mettre en place une application Serveur/Client

2- Il faut d'abord démarrer le serveur et ensuite Alice s'y connecte pour pouvoir échanger avec Bob

FIGURE 15. INFORMATION DE DEMARRAGE DU SERVEUR

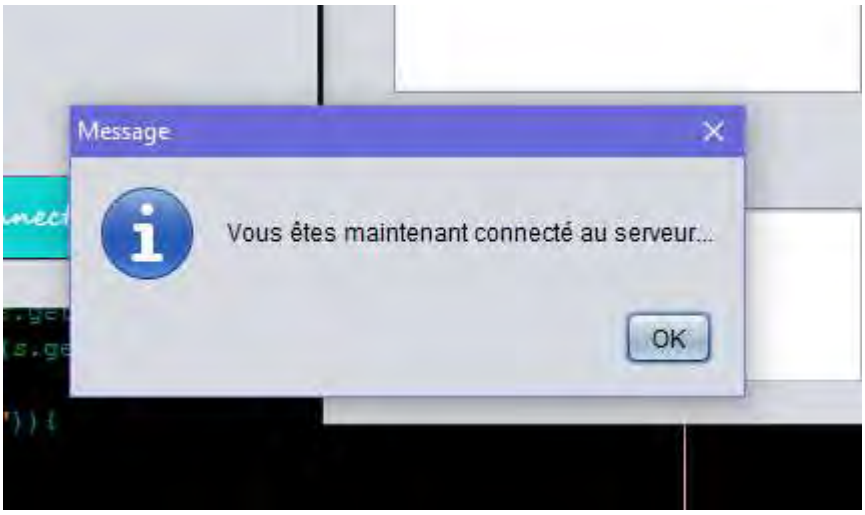


FIGURE 16. INFORMATION DE CONNEXION

3- Après connexion, l'application nécessite d'abord une génération de clés (clé privée et clé publique) pour Alice ou pour Bob.

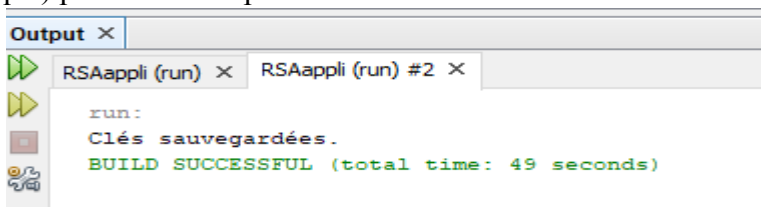


FIGURE 17. SAUVEGARDE DES CLES ET ECHANGE

4- Après cette étape de la génération de clés, les messages peuvent être envoyés.

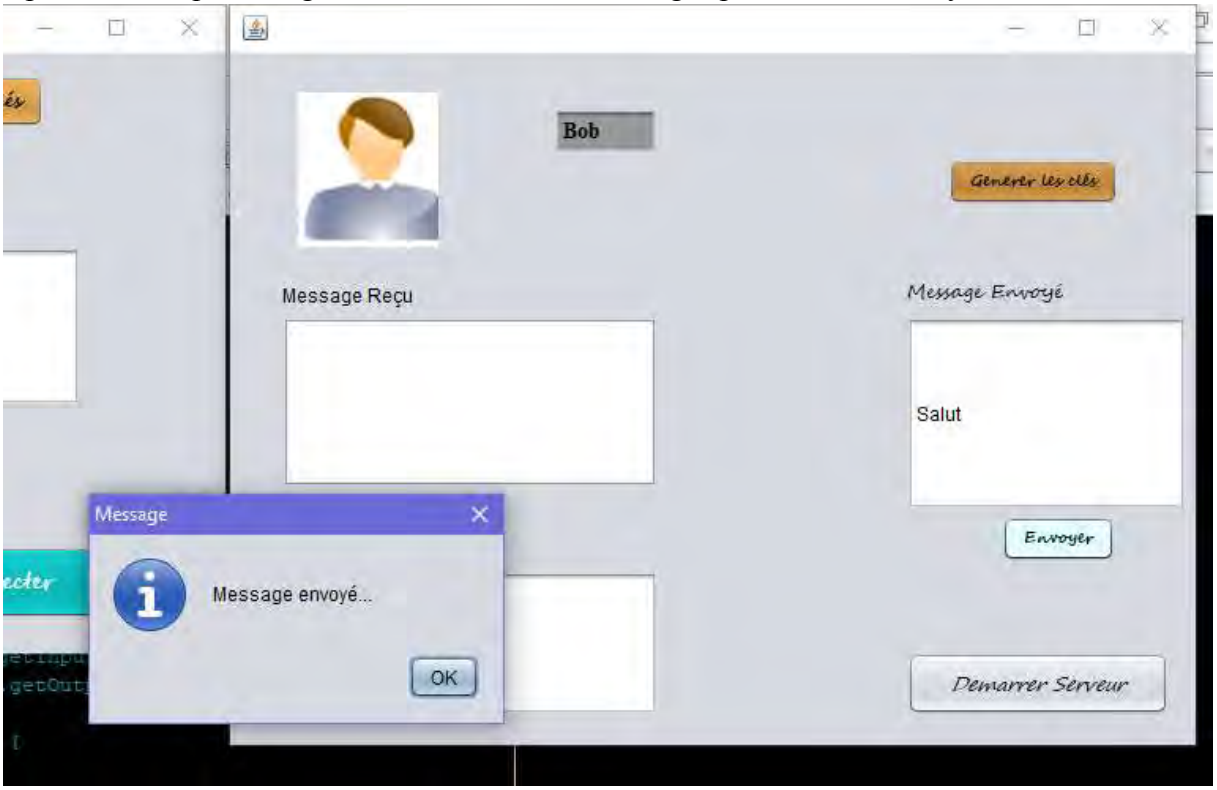


FIGURE 18. ENVOI DU MESSAGE SALUT A ALICE

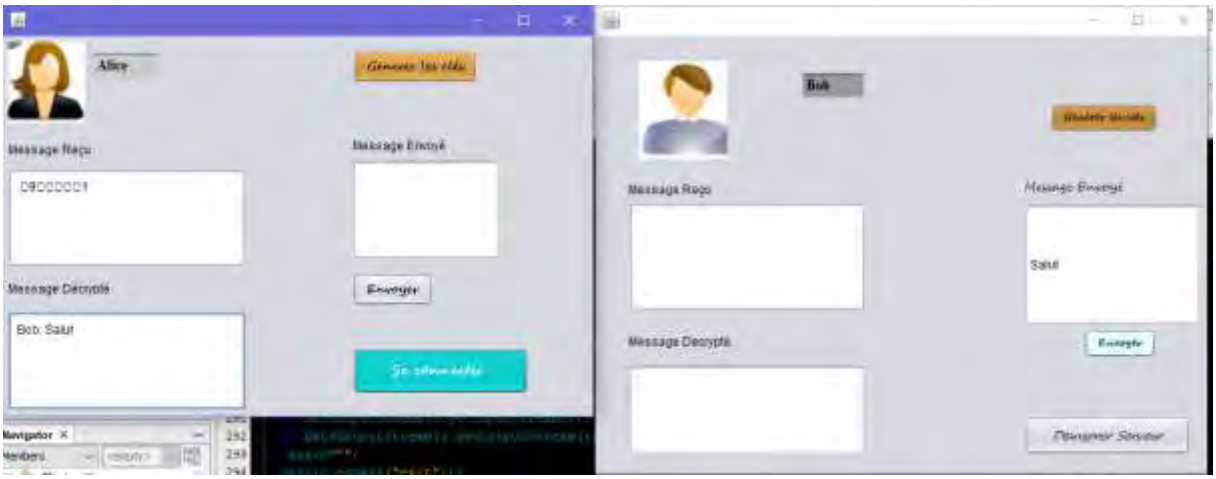


FIGURE 19. MESSAGE RECU CHIFFRE ET DECHIFFRE PAR ALICE GRACE A LA CLE PRIVEE

5- L'attaquant pourra ensuite réaliser une attaque par factorisation sur la clé privée de Alice et Bob.

FIGURE 20. INTERFACE DE L'ATTAQUANT

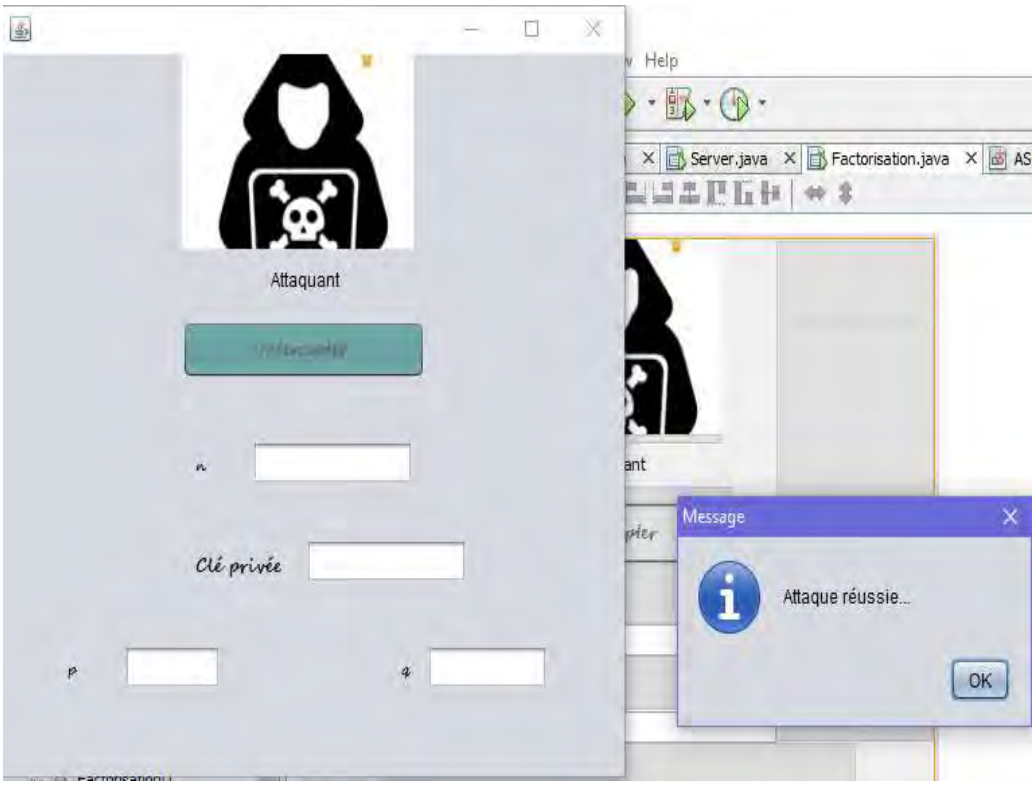


FIGURE 21. ATTAQUE REUSSIE APRES AVOIR CLIQUER SUR LE BOUTON INTERCEPTER

FIGURE 22. RECUPERATION ET AFFICHAGE DES DONNEES

6- Après l'obtention de la clé publique publiée dans le serveur, l'attaquant est en mesure de récupérer n pour la factorisation et cela grâce à une fonction de crible quadratique (quadratic sieve) implémentée en java et lancer sur la machine locale. Les valeurs de p et q sont sauvegardés dans un fichier, ils lui permettront de calculer l'exposant de déchiffrement d.

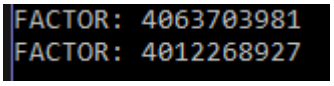


FIGURE 23. LES FACTEURS DE N (P ET Q)

Il est à noter qu'il a fallu 11000 millisecondes pour factoriser n. Une Clé Publique de 64bits.

7- Une fois que l'attaquant a la clé privée, il pourra intercepter et déchiffrer les messages transmis entre Alice et Bob.

```
// Decrypt message
public byte[] decrypt(byte[] message)
{
    return (new BigInteger(message)).modPow(d, N).toByteArray();
}
```

Figure 24. Fonction de Dechiffrement_1

```
private static String bytesToString(byte[] encrypted)
{
    String test = "";
    for (byte b : encrypted)
    {
        test += Byte.toString(b);
    }
    return test;
}
```

FIGURE 25. FONCTION DE DECHIFFREMENT_2

```
byte[] decrypted = rsa.decrypt(encrypted);
System.out.println("Decrypting Bytes: " + bytesToString(decrypted));
System.out.println("Decrypted String: " + new String(decrypted));
```

FIGURE 26. FONCTION DE DECHIFFREMENT_3

CADRE ANALYTIQUE

Chapitre III : Cadre Analytique

Il s'agira d'étudier et constater les résultats obtenus, ensuite d'essayer de repérer les solutions temporaires et enfin évaluer les solutions en tenant compte de leurs fondements et préceptes.

III-1- Analyse :

III-1-1- Méthode de factorisation utilisée :

III-1-1-1- Introduction :

La factorisation des entiers est non seulement un problème fondamental en théorie des nombres, mais a trouvé un nouvel intérêt avec l'arrivée de la cryptographie moderne. De nombreux algorithmes de factorisation existent comme cité dans la partie II (attaques de RSA-Factorisation de grands entiers). Les algorithmes les plus efficaces pour différentes tailles de nombres à factoriser sont : le crible algébrique et le crible quadratique. L'algorithme le plus répandu pour factoriser des nombres produits de deux facteurs premiers est le crible algébrique car ce dernier reste le plus performant à ce jour. Le crible quadratique est un autre membre de la famille d'algorithmes de factorisation, qui a prouvé sa performance pour factoriser des entiers de taille moyenne (jusqu'à quelques centaines de bits) et qui ne demande pas de connaissances particulières en théorie des nombres. Et nous présentons, ci-dessous, le crible quadratique qui est celui qui a été utilisé dans l'implémentation de notre application.

III-1-1-2- Crible quadratique :

La méthode du crible quadratique **[2]** a été inventée par C. Pomerance en 1981. C'est aujourd'hui la méthode la plus couramment utilisée pour factoriser des entiers n qui n'ont pas de diviseurs premiers significativement plus petits que $\sqrt{n}$. Elle ne dépend en fait que de la taille de n, de son nombre de chiffres décimaux, et non de propriétés arithmétiques particulières de ses diviseurs premiers. Avec cette méthode on parvient, avec des moyens informatiques adéquates, à factoriser tout entier ayant jusqu'à cent vingt chiffres décimaux.

Nous nous contentons de l'illustrer sur un exemple.

Soit

$$n = 21311 = 101 * 211$$

le nombre à factoriser. On choisit un entier m proche de la racine carrée de n

$$m = [n^{1/2}] = 146$$

On forme des congruences modulo n en observant que pour tout entier a,

$$(m + a)^2 \equiv (m^2 - n) + a^2 + 2am \pmod{n} = 5 + a^2 + 292a \pmod{21311},$$

où l'on note que $m^2 - n$ est de l'ordre de $\sqrt{n}$. On se donne une borne $B = 13$ et l'on cherche de petits entiers a tels que $5 + a^2 + 292a$ soit B-friable. Par exemple pour a compris entre −60 et 60 on trouve :

a	$5 + 292a + a^2$
-27	$-2.5^2 .11.13$
-5	$-2.5.11.13$
-1	$-2.11.13$
0	5
60	$5^3 .13^2$

On porte dans une matrice la parité des valuations :

	-1	2	5	11	13
-27	1	1	0	1	1
-5	1	1	1	1	1
-1	1	1	0	1	1
0	0	0	1	0	0
60	0	0	1	0	0

On forme des carrés à partir des lignes annulées par cette matrice. L'ensemble de ces lignes est un espace vectoriel dont une base est donnée par les trois lignes de la matrice suivante

-27	-5	-1	0	60
1	0	1	0	0
1	1	0	1	0
1	1	0	0	1

La première ligne du tableau donne la congruence

$(2.5.11.13)^2 \equiv (146 - 27)^2 . (146 - 1)^2 \pmod{21311}$.

On calcule alors le plus grand diviseur commun de $2.5.11.13-(146-27).(146-1) = -15825$ et de 21311. On trouve le facteur non trivial p = 211 de n = 21311 et son cofacteur 101. Un test de primalité prouve aisément que p et q sont premiers. La seule différence entre cet algorithme et les autres réside dans la manière de trouver des relations de congruences. Dans le crible quadratique le nombre supposé friable est de l'ordre de $\sqrt{n}$. La probabilité de succès est donc bien plus grande.

III-1-2- Résultats :

III-1-2-1- Factorisation de n :

Lors de la génération des clefs, l'attaquant va tenter de factoriser le module de RSA n pour trouver p et q pour ensuite calculer les clefs privées pour déchiffrer des messages interceptés.

Le tableau suivant montre le temps nécessaire pour factoriser le module n de différentes tailles.

N	**P**	**Q**	**Temps(millisecondes)**
33337	901	37	3000
72563	149	487	1000
84043	229	367	1000
90745	18149	5	2000
121037	7	17291	3000

761581	5479	139	3000

TABLEAU 2. TEMPS NECESSAIRE POUR FACTORISER LE MODULE N

III-1-2-2- Signature :

Dans le cas où le message est signé, il doit être d'abord haché avec l'algorithme SHA1. Une fois le message haché, il sera chiffré avec la clef privée de l'émetteur pour former une signature. L'attaquant doit d'abord factoriser le module n de RSA, puis calculer la clef privée d avec les deux facteurs p et q, retrouvés par la factorisation, il devra ensuite signer le nouveau message pour usurper l'identité de l'expéditeur et contourner l'authenticité.

III-1-2-3- Discussion :

On remarque que le temps de calcul par factorisation est proportionnel à la valeur de n constituant le module RSA.

On remarque aussi que si la clef privée est cassée, l'identité de l'expéditeur peut être usurpé et ainsi contourner la règle de l'authenticité.

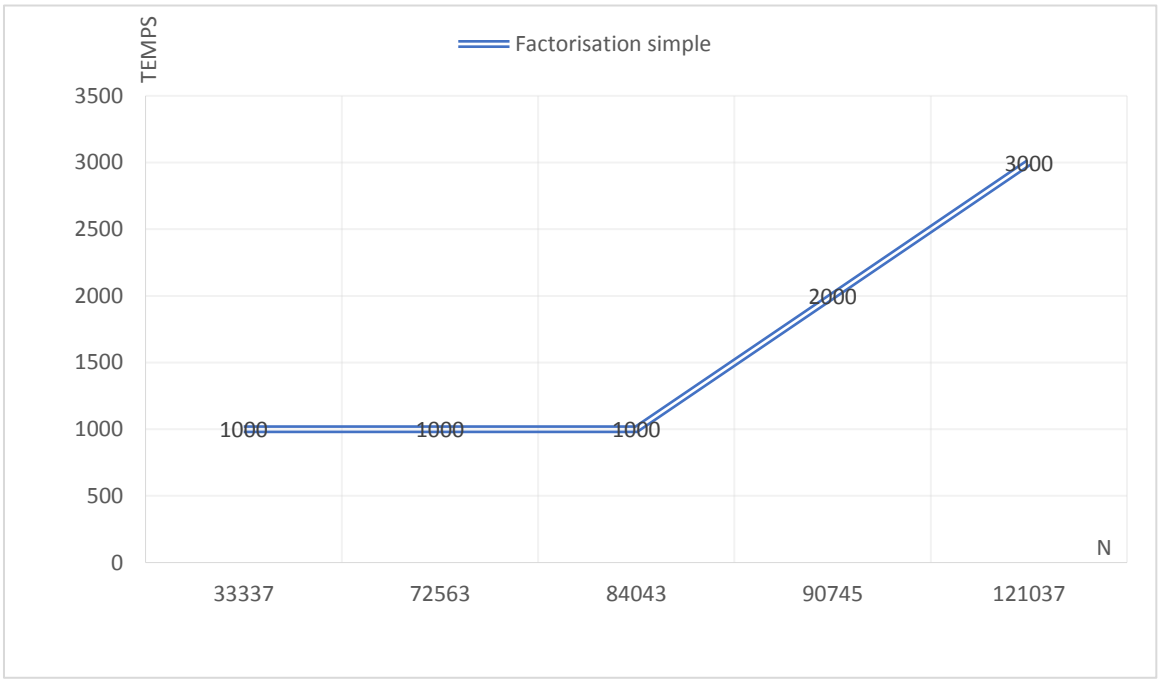


FIGURE 27. GRAPHE DU TEMPS CALCULE A LA FACTORISATION

III-1-2-4- Conclusion :

Les conclusions tenues par les différents scénarios testés montrent que la sécurité du système cryptographique RSA est associée au choix de la clef. Plus la taille de la clef est grande plus la difficulté de l'attaque augmente et plus le risque d'attaque diminue.

III-2- Identification des solutions alternatives :

Dans notre vie quotidienne, nous voulons de plus en plus que la sécurité des services auxquels nous avons accès soit garantie. Ceci implique l'usage intensif de systèmes sur puces dédiés, ayant la capacité de réaliser toutes les opérations cryptographiques nécessaires à cette tâche. La difficulté pour un utilisateur de retenir des clefs de chiffrement impose aux concepteurs de tels systèmes de stocker ces données confidentielles en leur sein. Ces données secrètes deviennent donc la cible prioritaire de pirates voulant s'octroyer l'accès à ces services.

Les méthodes à leur disposition sont multiples et nombreuses. La plupart d'entre elles ne demandent pas d'investir dans des équipements de pointe et excessivement coûteux. Nous savons vu que chaque attaque est rendue plus difficile par une méthode de protection appropriée. Ces protections ne les rendent pas impossibles, et régulièrement les attaques sont affinées pour passer outre les protections existantes, nécessitant ainsi le développement de nouvelles protections.

Au vu du risque que représentent ces attaques pour la sécurité des applications basées sur de tels circuits, il est nécessaire de mettre au point des mécanismes de protection. Leur but est de tenter de rendre ces attaques impossibles ou au moins très difficiles. Les protections mises au point sont propres aux attaques qu'elles limitent.

Par exemple :

- L'augmentation de la taille des clés permet un calcul qui s'étend dans le temps. Et pour un attaquant cela prendra beaucoup plus de temps pour trouver l'essence des échanges et donc la perte de sa valeur dans le temps car l'information a été transmise depuis des années.
- Nous pouvons aussi penser à éviter la dépendance des données manipulées, réduire le temps d'exécution.
- Les contre-mesures de masquage arithmétique sont basées sur la dissimulation des données manipulées pour éviter toute corrélation avec la puissance consommée pendant l'opération. Il est ensuite nécessaire en fin d'opération de démasquer le résultat.
- Mise en place du coprocesseur en remplacement au processeur habituel, qui sera uniquement réservé à la sécurité. « On ne peut se fier à un processeur généraliste » tel est l'avertissement répété inlassablement par nos chercheurs. Les spécialistes s'accordent sur cette nécessité de se munir d'un processeur ou d'un coprocesseur pour traiter les tâches cryptographiques.
- Etc.

Mais les protections ainsi développées sont efficaces contre l'attaque qu'elles ciblent, mais qu'en est-il vis-à-vis des autres types d'attaques ? Se protéger d'un côté n'ouvre-t-il pas de brèche sur un autre front ?

III-3- Evaluation des solutions : Critères et recommandations :

RSA est considéré comme sûr grâce à sa complexité algorithmique. En effet, RSA serait officiellement plus efficace et sûr en respectant certaines contraintes de longueur de clés et d'usage. Il est recommandé de l'utiliser brièvement au début d'un échange pour transmettre des clés secrètes de session d'un algorithme efficace à clés privées.

A l'heure actuelle, la factorisation connait de lentes améliorations au cours des années. La meilleure amélioration possible reste l'optimisation des algorithmes. Excepté un changement dramatique, le RSA-1024 restera sûr pour les prochaines années. D'après les projections, une clé de 2048 bits est sensée tenir jusqu'en 2079 si on tient compte de la loi de Moore. Mais ces valeurs sont correctes uniquement si on respecte les propriétés de e, d, p et q.

Depuis sa création le système RSA a résisté à toutes les attaques de manière satisfaisante si l'on choisit bien p, q et e, mais il faut constamment augmenter la taille de pq. Nous sommes partis de 512 bits, et même 389 bits pour les cartes à puces, à 2048 bits. Cette augmentation de taille est nécessitée par l'augmentation de la puissance des ordinateurs et par les progrès mathématiques des algorithmes de factorisations des entiers.

Pour évaluer et tester la sécurité du code RSA, il faut en particulier

- Évaluer la rapidité des algorithmes de factorisations de grands nombres entiers
- Démontrer que la clef secrète de déchiffrement d ne peut pas être obtenue sans factoriser n = pq et plus généralement qu'elle ne peut pas être calculée en temps polynomial en fonction de n.
- Montrer que l'on ne peut pas décoder un message sans la clef secrète

Dons les solutions proposées peuvent parfaitement correspondre à ce que recherche RSA pour garantir une sécurité parfaite de ses données.

Même si nous ne savons pas aujourd'hui si casser RSA est aussi difficile que factoriser n., nous pouvons toujours garder espoir sur la sûreté de RSA du moment que depuis 2500ans personne n'a trouvé une solution rapide au problème de factorisation…

CONCLUSION ET RECOMMANDATIONS :

Nonobstant les avantages qui enveloppent la cryptographie, elle a une possibilité maximum concernant sa sécurité et donc ne doit pas être considéré comme garant des autres mesures de sécurité. Si elle est utilisée proprement (chiffrement), elle est très puissante et peut bénéficier aux objectifs réels de l'entité.

Et le problème dominant demeure la gestion des clefs qui est aussi le talon d'Achille de la cryptographie. Il serait donc inéluctable de fiabiliser chaque étape y intercédant : la génération, l'échange, le stockage, la longévité, ou encore la destruction des clés.

Il est aussi important de ne pas oublier de mentionner la répercussion que peuvent avoir les avancées mathématiques dans ce domaine. En effet, un nombre impossible à factoriser aujourd'hui par les calculs informatiques, pourra l'être demain en quelques minutes à peine. Tout repose sur l'idée révolutionnaire d'un chercheur décidé à résoudre le problème. Toutefois, les quelques 20 ans de travaux réalisés sur les algorithmes du système RSA sont un véritable gage de sécurité, rendant cette éventualité encore lointaine.

Aujourd'hui, l'élaboration de compétition visant à tester des algorithmes est très populaire. Cette expérimentation est usuellement appliquée en vue de déceler l'infaillibilité d'un algorithme et ainsi favoriser l'évolution des recherches sur la cryptographie.

Dans ce mémoire, nous avons eu à faire une ébauche de différentes attaques possibles sur RSA et à esquisser une vue d'ensemble de la cryptanalyse qui est une branche très importante de la cryptographie. Et pour mieux exposer notre sujet, nous avons décrit l'attaque par factorisation sur lequel nous avons fait une simulation.

Et grâce aux tests effectués, nous avons pu montrer que le succès de l'attaque relève de la taille de la clé privée et de l'efficacité des ressources matérielles.

Cependant, pour des éventuels travaux sur l'algorithme de RSA, nous proposons l'inclusion de nouveaux processus d'attaque tout en comparant et analysant les résultats obtenus et en tenant bien en compte l'arrivée des ordinateurs quantiques et des possibilités qu'ils offrent (ex. algorithme de Shor qui semble résoudre le problème de la complexité des polynômes).

Bernard Baruch disait : « Un spéculateur, c'est un homme qui observe le futur et agit avant qu'il n'arrive ». Alors nous disons pour une évolution probable et appréciable de la cryptographie, surtout en matière de sécurité, nous nous devons d'être des spéculateurs très avertis.

REFERENCES

Bellare, M., & Rogaway, P. (1994). Optimal Asymetric Encryption. *EUROCRYPT '94, Lecture Notes in Computer Science*(950), 92-111.

Bleichenbacher, D. (1998). Chosen CipherText Attacks against protocols based on the RSA based on the RSA Encryption Standard PKCS 1. *CRYPTO '98, Lecture Notes in Computer Science*(1462), 1-12.

[1] Boneh, D. (1996). *Twenty Years of Attacks on the RSA Cryptosystem.*

Coppersmith, D. (1997). Small Solutions to polynomial equations, and lox exponents RSA vulnerabilities. *Journal of Cryptology*(10), 233-260.

Coppersmith, D., Franklin, M., Patarin, J., & Reiter, M. (1996). Low Exponent RSA with Related messages. *EUROCRYPT '96, Lecture Notes in Computer Science*(1070), 1-9.

[2] Enge, A. (s.d.). *Factorisation par le crible quadratique*. Récupéré sur http://www.enseignement.polytechnique.fr/profs/informatique/Eric.Goubault/Cours09/qs.pdf

[3] GROSSHANS, F., & GRANGIER, P. (s.d.). Récupéré sur Optique Quantique: https://www.photoniques.com/articles/photon/pdf/2014/03/photon201471p34.pdf

[4] Haroche, S. (2001). *Les algorithmes quantique*. Récupéré sur Chaire de physique quantique: https://www.college-de-france.fr/media/serge-haroche/UPL55031_SHaroche_260202.pdf

Hastad, J. (1988). Solving Simultaneous modular equations of low degrees. *SIAM J. of Computing*(17), 336-341.

Kocher, P. (1996). Timing Attacks on Implementations of Diffie Hellman, RSA, DSS, and other systems. *CRYPTO '96, Lecture Notes in Computer Science* (1109), 104-113.

[5] Mkhinini, A. (2018). *Implantation matérielle de chiffrements homomorphiques - Page 27*. Récupéré sur HAL- Archives Ouvertes : https://tel.archives-ouvertes.fr/tel-01772355/document

[6] Porumbel, D. (s.d.). *Algorithmes Heuristiques et Techniques d'apprentissage*. Récupéré sur HAL - Archives-Ouvertes: https://tel.archives-ouvertes.fr/tel-00481253/document

[7] Sacco, L. (s.d.). *Ordinateur Quantique*. Récupéré sur Futura-Sciences: https://www.futura-sciences.com/sciences/definitions/physique-ordinateur-quantique-4348/

[8] Technologies, N. I. (2015). *Secure Hash Standards*. Récupéré sur https://csrc.nist.gov/publications/detail/fips/180/4/final

Wiener, M. (1990). Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information Theory*(36), 553-558.

[9] Wikipédia. (s.d.). *Algorithme de Shor*. Récupéré sur Wikipédia, l'Encyclopédie libre: https://fr.wikipedia.org/wiki/Algorithme_de_Shor

[10] Wikipédia. (s.d.). *Congruence sur les entiers*. Récupéré sur Wikipédia, l'Encyclopédie libre: https://fr.wikipedia.org/wiki/Congruence_sur_les_entiers

[11] Wikipédia. (s.d.). *Cryptographie*. Récupéré sur Wikipédia, l'Encyclopédie libre: https://fr.wikipedia.org/wiki/Cryptographie#:~:text=La%20cryptographie%20est%20une%20des,souvent%20de%20secrets%20ou%20cl%C3%A9s.

[12] Wikipédia. (s.d.). *Distribution quantique de clés*. Récupéré sur Wikipédia, l'encyclopédie libre: https://fr.wikipedia.org/wiki/Distribution_quantique_de_cl%C3%A9

[13] Wikipédia. (s.d.). *Hypothèse calculatoire*. Récupéré sur Wikipédia, l'Encyclopédie libre: https://fr.wikipedia.org/wiki/Hypoth%C3%A8se_calculatoire#:~:text=En%20cryptographie%2C%20une%20hypoth%C3%A8se%20de,la%20robustesse%20des%20primitives%20cryptographiques.&text=L%27%C3%A9valuation%20de%20la%20difficult%C3%A9,est%20%C3%A9tudi%C3%A9e%20par%2

[14] Wikipédia. (s.d.). *Intrication Quantique*. Récupéré sur Wikipédia, l'Encyclopédie libre: https://fr.wikipedia.org/wiki/Intrication_quantique#:~:text=En%20m%C3%A9canique%20quantique%2C%20l%27intrication,la%20distance%20qui%20les%20s%C3%A9pare.

[15] Wikipédia. (s.d.). *Réseau de Feistel*. Récupéré sur Wikipédia l'encyclopédie libre: https://fr.wikipedia.org/wiki/R%C3%A9seau_de_Feistel

[16] Wikipédia. (s.d.). *Test de Primalité MIller - Rabin*. Récupéré sur Wikipédia, l'encyclopédie libre: https://fr.wikipedia.org/wiki/Test_de_primalit%C3%A9_de_Miller-Rabin

[17] Wikipédia, l. l. (2021, Avril 18). *Informatique quantique*. Récupéré sur Wikipédia, l'Encyclopédie libre: https://fr.wikipedia.org/wiki/Informatique_quantique#:~:text=L%27informatique%20quantique%20est%20le,%27informatique%20dite%20%C2%AB%20classique%20%C2%BB.

[18] Wikipédia, l. l. (s.d.). *Cryptographie Post - Quantique*. Récupéré sur Wikipédia: https://fr.wikipedia.org/wiki/Cryptographie_post-quantique

WEBOGRAPHIE :

- **Comment strusturer et écrire un bon mémoire de Master ès sciences en Sciences d'Informations,** Unil, HEC Lausanne, URL : https://www.unil.ch/
- **Projet de recherche d'un mémoire : que mettre dans votre proposition de recherche ?** Justine Debret, 17/07/2018, URL : https://www.scribbr.fr/memoire/projet-de-recherche-memoire/
- **Introduction de votre mémoire**, Justine Debret, 18/04/2018, URL : https://www.scribbr.fr/memoire/introduction
- **Conception du cadre théorique et méthodologique de l'étude,** Institut Numérique, 2013, URL : https://www.institut-numérique.org/
- **Qu'est-ce-que le cadre théorique d'un mémoire ? ;** Bas Swaen, 08/02/2016, URL : https://www.scribbr.fr/memoire/cadre-theorique/
- **Cadre Méthodologique**, Afric Mémoire, URL : https://www.africmemoire.com/
- **Histoire de la cryptologie**, Wikipédia l'encyclopédie libre, URL : https://fr.wikipédia.org/wiki/Histoire_de_la_cryptologie/
- **La Cryptographie à clé publique – Principe de fonctionnement**, Cryptographie et Codes Secrets, URL : https://www.bibmath.net/
- **Cryptographie asymétrique**, Wikipédia l'encyclopédie libre, URL : https://fr.wikipédia.org/wiki/cryptographie_asymétrique/
- **La cryptographie à clé publique**, Cryptage, URL : https://www.cryptage.org/
- **Triple DES**, Bookwiki, URL : https://boowiki.info/
- **l'AES : Advanced Encryption Standard**, SoGoodToBe, 04/11/2001**,** URL : https://www.securiteinfo.com/cryptographie/aes/
- **AES 128 bits**, JM Dutertre, 2011, Adresse URL : https://www.emse.fr/
- **Introduction au chiffrement avec DES**, Comment ça marche, URL : https://www.commentçamarche.com/
- **Chiffrement RSA,** Wikipédia l'encyclopédie libre, URL : https://fr.wikipédia.org/wiki/chiffrementRSA/
- **Cryptosystème RSA**, URL : https://www.nymhomath.ch/
- **Cryptographie – Introduction à RSA**, URL : https://web.maths.unsuw.edu.au/
- **RSA**, Thibault Allançon, 31/05/14, URL : https://haltode.fr/
- **Cryptanalyse**, Wikipédia l'encyclopédie libre, URL : https://fr.wikipédia.org/wiki/Cryptanalyse/
- **Cryptanalyse**, Pablo Rauzy, URL : https://pablo.rauzy.name/teaching/sese
- **Mise en place d'un cryptosystème pour la sécurité des données et la détection d'intrusion**, Landry Ndjate, 2014, URL : https://www.memoireonline.com/
- **RSA : Théorie et Attaque**, URL : https://perso.crans.org/genest/RSA.pdf
- **Attaque de Hastad**, Cryptanalyse RSA, Adresses URL : https://zweisamkeit.fr/
- **Quand le chiffrement des données est mis à mal par des mathématiciens**, Sébastien Gavois, 04/08/14, URL : https://www.nextinpact.com/
- **Protection des systèmes d'information contre les attaques entrées-sorties**, Cryptographie et Sécurité, INSA de Toulouse, Fernand Lone Sang, 2012,
- **Comment la cryptographie se prépare à faire aux cyberattaques quantiques**, François Manens, 09/01/2020, URL : https://cyberguerre.numerama.com

- **Comprendre l'informatique quantique – cryptographie**, Olivier Ezratty, 03/09/18, Internet – Quantiques – Startups, URL : https://www.oezratty.net/
- **Cryptographie Post-Quantique**, Wikipédia l'encyclopédie libre, URL : https://fr.wikipédia.org/wiki/Cryptographie_post_quantique//
- **Cryptographie Quantique**, Wikipédia l'encyclopédie libre, URL : https://fr.wikipédia.org/wiki/Cryptographie_quantique//