Cartes à puce

Louis Goubin

27 février 2005

Chapter 9

Cartes à puce

Chaque fois qu'un Gonda désirait quelque chose de nouveau, des vêtements, un voyage, des objets, il payait avec sa clé. Il pliait le majeur, enfonçait sa clé dans un emplacement prévu à cet effet et son compte, à l'ordinateur central, était aussitôt diminué de la valeur de la marchandise ou du service demandés.

René Barjavel, *La nuit des temps* (1967)

This application will put a sophisticated information-security device in the wallet or purse of practically every person in the industrialized world, and will therefore be the most extensive application ever made of cryptographic schemes.

Gustavus Simmons (1992)

9.1 L'invention de la carte à puce

En 1967, dans *La nuit des temps*, titre d'un roman de science-fiction du romancier français René Barjavel, on trouve l'histoire d'un peuple mythique, les Gondas - une civilisation vieille de milliers d'années mais très avancée - qui utilise un anneau magique ayant des moyens de mémorisation et de communication. On peut y voir l'origine (au moins littéraire) de la carte à puce.

L'idée d'utiliser un composant électronique contenu dans une carte de crédit apparaît presque au même moment, et de nombreux brevets commencent à être publiés. Aux États-Unis, Pomeroy (1967), puis Jules Ellingboe (1970), qui décrit concrètement un moyen de paiement électronique sur une carte de crédit à contacts, et John Halpern (1972), avec son stylo électronique sécurisé de paiement. Au Japon, Kunitaka Arimura (1970), qui propose une méthode d'authentification dynamique réalisée à l'aide d'un dispositif d'identification. En Allemagne, Jürgen Dethloff (1977). Et en France, Roland Moreno (1974), Michel Ugon (1977) et Louis Guillou (1979), ainsi que de nombreux autres.

La carte de Roland Moreno était une simple *carte à mémoire*, dite aussi *carte à logique câblée* et n'était pas programmable. Dès 1977, Michel Ugon – à qui on avait confié l'étude du

problème chez CII-Honeywell Bull – se rend compte que seule la présence d'un microprocesseur peut donner à la carte suffisamment de fonctionnalités, notamment pour assurer la sécurité au moyen d'algorithmes cryptographiques. Il devient alors clair que la carte doit être intelligente.

C'est ainsi que prend naissance la *carte à microprocesseur*, dite aussi *carte à microcalculateur*, dont le premier exemplaire est baptisé CP8. C'est à l'époque une carte bi-puces, qui présente de ce fait des faiblesses sécuritaires évidentes, car un attaquant peut connaître le contenu des informations qui transitent entre la mémoire et le microprocesseur. Pour cette raison, en 1981, en collaboration avec Motorola, voit le jour le premier calculateur *monolithique* pour la carte à puce, appelé SPOM (*Self Programmable One-chip Microprocessor*).

En 1984, la carte à puce est choisie par le GIE Cartes Bancaires comme élément principal de la sécurité du réseau de paiement électronique français (plutôt que la logique câblée) qui se déploie à partir de 1986. En 1992, Gustavus Simmons prophétise dans [103] :

> "Cette application [la carte à microprocesseur] mettra un dispositif de sécurité de l'information dans le portefeuille ou le porte-monnaie de pratiquement tout le monde dans le monde industrialisé, et constituera de ce fait l'application la plus étendue jamais mise en œuvre pour les schémas cryptographiques."

Et de fait, à partir de cette date, la carte à puce se généralise à de nombreux autres domaines: la télévision à péage, le porte-monnaie électronique, la téléphonie mobile, etc[1].

9.2 Fonctionnement d'une carte à microprocesseur

9.2.1 Description physique

Une carte à microprocesseur[2] est constituée d'un *micro-module* (appelé aussi *puce*) inséré dans rectangle de plastique au format "carte de visite" sur lequel sont en général inscrites des informations liées à l'identité du possesseur de la carte. Par ailleurs, la carte peut aussi comporter une piste magnétique (c'est le cas notamment des cartes bancaires actuelles). Conformément au standard ISO/IEC 7816-1 [52], les dimensions de la carte à microprocesseur doivent vérifier :

- $85.47 \text{ mm} \leq \text{Largeur} \leq 85.72 \text{ mm}$;
- $53.92 \text{ mm} \leq \text{Hauteur} \leq 54.03 \text{ mm}$;
- $\text{Épaisseur} = 0.76 \pm 0.08 \text{ mm}$

Comme spécifié dans le standard ISO/IEC 7816-2 [53], le micro-module comporte huit contacts, dont six sont effectivement connectés à la puce elle-même (qui est en général invisible).

[1] Pour plus de détails sur les débuts de la carte à microprocesseur, on peut consulter [43, 44, 45, 105].
[2] Ce paragraphe s'appuie en partie sur des données disponibles dans [71, 93, 90].

Les contacts sont utilisés pour l'alimentation (V_{cc} et V_{pp})[3], la masse (GND), l'horloge (CLK), le signal Reset (RST)[4] et le contact I/O d'entrée sortie par lequel transitent en mode *half-duplex* toutes les données échangées entre la carte et le monde extérieur.

Le plus souvent, la carte possède un processeur 8 bits, mais il existe des modèles à base de processeurs 16 et 32 bits. On dispose même de processeurs à architecture RISC [30]. Dans la plupart des cartes actuelles, le microprocesseur est construit autour d'un cœur Motorola 6805 ou Intel 8051, avec une horloge à 5 MHz. Même s'il est toujours possible de les augmenter, les fréquences d'horloge sont limitées par deux facteurs : la consommation du processeur, qui ne doit pas dépassser une certaine limite, et la conformité au standard ISO/IEC 7816-3 [54], qui oblige à respecter une certaine plage de fréquence, pour rester compatible avec les matériels déjà en place (notamment les lecteurs).

En revanche, dans les cartes à microprocesseur récentes, un système de multiplicateur de fréquence permet d'obtenir une horloge interne fonctionnant jusqu'à à 40 MHz, ce qui permet d'effectuer des calculs cryptographiques beaucoup plus rapidement. Par ailleurs, la carte contient souvent un *coprocesseur* cryptographique (ou *crypto-processeur*), généralement associé à un *générateur de nombres aléatoires* (RNG, *Random Number Generator*).

La carte à microprocesseur comporte en général trois types de mémoire :

- La ROM (*Read Only Memory*), qui n'a pas besoin d'être alimentée pour conserver l'information qu'elle contient. On l'utilise pour stocker le système d'exploitation de la carte, ainsi que les données permanentes. Ces éléments sont inscrits dans la carte dans la phase dite de "masquage", et ne peuvent plus être modifiés ensuite.
- L'E^{2}PROM (ou EEPROM, *Electrical Erasable Programmable Read Only Memory*) qui peut, comme la ROM, préserver son information même quand la carte n'est plus sous tension. La différence avec la ROM est qu'elle peut être modifiée par une application. Un problème de cette mémoire est sa durée de vie limitée en nombre de cycles d'écriture (typiquement de l'ordre de 100000 cycles) et en temps (10 ans), ce qui nécessite de changer régulièrement la carte pour éviter des dysfonctionnements. Un autre inconvénient est sa lenteur d'accès : l'E^{2}PROM est en effet en moyenne 1000 fois plus lente en écriture que la RAM.
- La RAM (*Random Access Memory*), qui est utilisée comme espace de stockage temporaire grâce à la rapidité des temps d'accès. Elle possède un caractère non persistent : dès que la carte n'est plus sous tension, la RAM perd son contenu. En revanche, elle peut être lue et écrite indéfiniment.

9.2.2 Communication avec la carte

La carte à microprocesseur contient un port de communication série (*via* une liaison asynchrone), pour échanger les données et les informations de contrôle avec le monde extérieur. La

[3] V_{cc} correspond à la tension d'alimentation en lecture, alors que V_{pp} correspond à la tension à appliquer, sur demande de la carte, pour programmer la mémoire de données (tension en écriture).

[4] Une tension appliquée sur ce contact déclenche l'initialisation physique et logique du composant.

vitesse de transmission est généralement de 9600 bits par seconde, mais le standard ISO/IEC 7816-3 autorise jusqu'à 115200 bits par seconde.

Le protocole suivant lequel la carte communique avec l'extérieur est également spécifié par le standard ISO/IEC 7816-3, qui définit deux possibilités : la première (appelée $T = 0$) prend l'octet comme structure élémentaire, alors que la seconde (dite $T = 1$) s'appuie sur le bit comme élément d'information[5].

La tension électrique, le traitement des erreurs et la fréquence d'horloge imposent l'utilisation d'un dispositif matériel pour dialoguer avec la carte. Ce dispositif, appelé CAD (*Card Acceptance Device*), est l'équivalent d'un UART (*Universal Asynchronous Receiver/Transmitter*), avec des fonctionnalités plus sophistiquées. Il comporte typiquement :

- une interface mécanique : le *connecteur* ;
- une interface électronique : le *coupleur* ;
- un boîtier contenant ces deux éléments : le *lecteur de carte*.

Les lecteurs les plus simples sont comparables à des modems, et gèrent le protocole de communication de façon élémentaire, sans interagir avec le système d'exploitation de la carte. Ils peuvent en principe fonctionner avec n'importe quelle carte à microprocesseur compatible avec les standards ISO/IEC 7816.

Il existe des lecteurs plus complexes, qui peuvent être en partie reprogrammés et contenir des données (comme des clés), des fichiers et des programmes. Ils peuvent exécuter des algorithmes cryptographiques, disposer de clavier, d'écran, d'un langage de programmation spécifique. Ces lecteurs ne sont en général plus universels : ils sont dédiés à certaines cartes.

9.2.3 Format logique des commandes

Pour fonctionner avec une carte à microprocesseur, le lecteur doit pouvoir exécuter les fonctions suivantes :

- mettre la carte sous tension, ou hors tension ;
- initialiser (physiquement et logiquement) la carte ;
- lire des données de la carte (commande *get*) ;
- écrire des données dans la carte (commande *put*).

Chaque commande *get* et *put* contient un en-tête, qui indique à la carte comment traiter les données présentes dans le reste de la commande. Plus précisément, l'en-tête est constitué de cinq octets appelés CLA, INS, P1, P2 et LEN, qui contiennent respectivement la classe,

[5]Remarquons que le standard prévoit en fait jusqu'à 14 protocoles, $T = 14$ correspondant à un protocole de communication propriétaire.

l'instruction, le premier paramètre, un second paramètre, et la longueur des données à traiter. La carte renvoie un octet d'accusé de réception au début de la commande, ainsi que deux octets d'état SW1 et SW2 à la fin de la commande.

Le standard ISO/IEC 7816-4 [55] vise à assurer une inter-opérabilité. Il spécifie le contenu des messages entre la carte et le lecteur. Pour cela, il utilise le protocole APDU (*Application Protocol Data Units*) pour les commandes et les réponses. Le standard définit également les structures des fichiers et des données :

- l'accès à ces données ;
- l'architecture de sécurité ;
- la sécurisation des communications.

9.3 Performances pour la signature électronique

Pour les implémentations sur PC, une étude approfondie des performances des algorithmes de signature a eté menée par le projet européen NESSIE [74]. Nous évoquons ici le cas des cartes à microprocesseur.

9.3.1 Multiplications et exponentiations modulaires

La multiplication et l'exponentiation modulaire sont à la base de nombreux algorithmes à clé publique, et leur temps de calcul est crucial pour les schémas tels que RSA, Schnorr, ElGamal, DSA, ainsi que ECDSA ou ECNR si le corps fini choisi est $\mathbb{F}_p$ (avec p premier). De nombreuses méthodes ont été proposées pour effectuer ces calculs, notamment par Montgomery [70], Barrett [9], Sedlak [99], De Waleffe et Quisquater [29], ou encore Dhem et Quisquater [32].

Les performances de ces méthodes sont comparables, comme le montrent les chiffres donnés par David Naccache et David M'Raïhi dans [71] ou par Helena Handschuh et Pascal Paillier dans [47, 48]. En revanche, le choix de l'une ou l'autre méthode a un impact important sur l'architecture interne d'un éventuel coprocesseur cryptographique (voir [79], ou bien [60] pages 248 à 250).

9.3.2 Temps de calcul sur une carte à microprocesseur

Les figures 1 et 2 donnent une évaluation des temps de calcul pour générer et vérifier une signature sur une carte à microprocesseur. Les algorithmes considérés sont RSA [92], Rabin-Williams [89, 107], Fiat-Shamir [35] (dans sa version signature), GQ [42] et GQ2 [85] (dans leur version signature), DSA [75, 76], ElGamal [34], ACE-Sign [98], ESIGN [36], ECDSA [106, 51], QUARTZ [26, 27], SFLASH [24, 25], NTRUSign [49], SP (*Small Primes*) [72], PKP (*Permuted Kernel Problem*) [100] et IP (*Isomorphisms of Polynomials*) [83].

La fréquence d'horloge (interne) est 10 MHz. Dans la figure 1, on considère une carte "bas de gamme" sans coprocesseur cryptographique. La figure 2 montre l'impact de la présence

Algorithme de signature	Taille de la signature (en octets)	Taille de la clé publique (en octets)	Temps de signature (en ms)	Temps de vérification (en ms)
RSA 1 024 bits	128	128	73 200	285
Rabin-Williams 1 024 bits	128	128	73 200	143
Fiat-Shamir	500	512	71	71
GQ	64	128	7 140	3 570
GQ2	64	128	7 140	3 570
DSA	40	128	8 660	17 300
ElGamal	128	128	26 700	53 500
ACE-Sign	de 425 à 705	620	$\simeq$ 37 100	$\simeq$ 43 700
ESIGN	144	145	$\simeq$ 3 260	$\simeq$ 384
ECDSA	48 (peut être réduit à 41)	48	825	1 610
QUARTZ	16	71 000	$>$ 1 min	$\simeq$ 10
SFLASH	33	15 400	59	$\simeq$ 15
NTRUSign - 251	220	128	256	288
SP	5 000	1 000	866	17 300
PKP	5 000	128	1 430	714
IP	250	256	355	71

Figure 9.1: Temps de calcul pour une carte à microprocesseur à 10 MHz (sans coprocesseur)

d'un coprocesseur sur les performances de RSA et Rabin-Williams 1024 bits, ainsi que de l'algorithme ECDSA lorsque le corps fini est $\mathbb{F}_p$, où p est un nombre premier de 163 bits.

Algorithme de signature	Taille de la signature (en octets)	Taille de la clé publique (en octets)	Temps de signature (en ms)	Temps de vérification (en ms)
RSA 1 024 bits	128	128	119	7
Rabin-Williams 1 024 bits	128	128	119	4
ECDSA	48 (peut être réduit à 41)	48	51	99

Figure 9.2: Temps de calcul pour une carte à microprocesseur à 10 MHz (avec coprocesseur)

9.4 Les attaques physiques

The vast majority of security failures occur at the level of implementation detail.

Ross Anderson (1993)

The great practicality and the inherent availability of physical attacks threaten the very relevance of complexity-theoretic security. Why erect majestic walls if comfortable underpasses will always remain wide open ?

Silvio Micali et Leonid Reyzin (2003)

Dans la conception traditionnelle de la *cryptologie*, la sécurité est vue de manière abstraite : pour attaquer un cryptosystème, l'attaquant se borne à échanger des messages avec celui-ci, et espère en les utilisant pouvoir mettre en défaut les objectifs visés (confidentialité, intégrité, authenticité, ...) par différentes techniques de *cryptanalyse*. La *cryptographie* classique s'efforce donc de construire des schémas avec si possible des preuves relatives de sécurité contre ce type d'attaque, en admettant la difficulté de certains problèmes mathématiques au sens de la *théorie de la complexité*.

Dans un modèle de sécurité plus étendu, on tient compte également, depuis quelques années, des *attaques physiques*. Ce nouveau concept prend en considération non seulement la sécurité des cryptosystèmes au sens *mathématique*, mais aussi les aspects liés à la nature *physique* des calculs. Ces attaques nouvelles sont particulièrement menaçantes pour les *systèmes embarqués* tels que les *cartes à microprocesseur*, contre lesquels l'adversaire peut mobiliser des moyens d'analyse de plus en plus sophistiqués.

Dans ce domaine, de nombreux travaux ont été effectués soit pour mettre en évidence de nouvelles stratégies d'attaques physiques, soit pour proposer des *contre-mesures*, avec dans certains cas des *preuves de sécurité* en établissant un modèle convenable de l'adversaire.

Dans tout ce paragraphe, les attaques physiques sont considérées en tant que menaces sur la sécurité des protocoles et algorithmes cryptographiques dans les cartes à microprocesseur. Néanmoins, tous les systèmes embarqués utilisant la cryptographie en sont aussi potentiellement la cible[6].

9.4.1 Classification des attaques physiques

Afin d'évaluer le niveau de résistance de ses produits, IBM a proposé en 1991 une taxonomie des attaquants potentiels [1] :

- Les "amateurs éclairés" (classe I) : ils sont souvent très intelligents, mais ont une connaissance imparfaite du système. Ils ont accès uniquement à du matériel de sophistication

[6] Ainsi l'analyse de la vulnérabilité spécifique des FPGA (Field Programmable Gate Arrays) a été récemment abordée par T. Wollinger, C. Paar [108] et S.B. Örs, E. Oswald, B. Preneel [80].

moyenne. Ils essaient souvent de tirer avantage de faiblesses existantes du système, plutôt que d'en créer de nouvelles.

- Les "attaquants experts" (classe II) : ils ont reçu une solide formation technique et ont de l'expérience. Ils ont une compréhension variable des parties du système, mais ont un accès potentiel à toutes ces parties. Ils possèdent souvent des outils et des instruments d'analyse hautement sophistiqués.

- Les "organisations financées" (classe III) : elles sont capables de rassembler des équipes de spécialistes ayant des capacités complémentaires, soutenues par des financements importants. Elles sont capables d'analyser en profondeur le système, de mettre au point des attaques complexes, et d'utiliser les outils d'analyse les plus modernes. Des attaquants de la classe II peuvent faire partie de ces équipes.

De même, on a pris l'habitude de classer les attaques physiques selon deux critères : les attaques *invasives* ou *non-invasives*, et les attaques *actives* ou *passives*.

Attaques invasives ou non-invasives

Une attaque *invasive* nécessite la suppression de l'enveloppe du micro-module, afin de pouvoir accéder directement à sa structure interne. Un exemple typique consiste à brancher une dérivation sur un *bus de données* afin d'intercepter les transferts de données.

Au contraire, dans une attaque *non-invasive*, l'adversaire n'utilise que les informations disponibles à l'extérieur du système, comme les temps d'exécution, la puissance électrique consommée, le rayonnement électromagnétique, *etc.*

On parle aussi parfois d'attaque *semi-invasive* [104], lorsque l'attaquant enlève l'enveloppe de la puce pour accéder à sa surface, mais garde intacte la couche de protection du micro-module (ces attaques ne nécessitant pas l'établissement d'un contact électrique avec la surface de métal).

Les cartes à microprocesseur sont munies de mécanismes de protection pour contrecarrer les attaques invasives. La technologie actuelle utilisée met ainsi en jeu (entre autres) plusieurs couches métalliques de protection, des détecteurs d'intrusion, ou encore un stockage des données sous des formats particuliers qui rendent très difficile leur interprétation. En revanche, les attaques non-invasives sont par définition indétectables. Comme elles sont en général également les moins onéreuses, ce sont elles qui constituent le principal danger pour la sécurité des cartes à puce.

Attaques actives ou passives

Dans une attaque *active*, on tente de perturber le fonctionnement normal du système. Ainsi les attaques par *injection de fautes* ont pour objectif de provoquer des erreurs au cours de certains calculs effectués par le système (voir le paragraphe 3).

Une attaque active implique une modification de l'environnement physique de la carte pour la placer dans des conditions anormales de fonctionnement. Plusieurs moyens sont à la disposition de l'attaquant[7] :

- L'*alimentation* : selon le standard ISO/IEC 7816-2 [53], le micro-module doit pouvoir supporter une tension d'alimentation V_{cc} comprise entre 4,25 et 5,25 volts. Pour ces valeurs, la carte doit fonctionner normalement. En revanche, si une variation brusque de l'alimentation (appelée *spike*) fait sortir V_{cc} de l'intervalle de tolérance, cela peut provoquer un résultat faux, à supposer que la carte soit capable de finir complètement le calcul.

- L'*horloge* : de façon analogue, le standard ISO/IEC 7816-2 définit une fréquence de référence pour l'horloge externe ainsi qu'un intervalle de tolérance. L'utilisation d'une fréquence anormalement haute ou basse peut également résulter en des erreurs[8].

- La *température* : placer la carte dans des conditions de température extrêmes est un moyen potentiel de provoquer des fautes, même s'il est assez peu utilisé aujourd'hui dans la pratique.

- Les *rayonnements* : le folklore présente souvent les attaques par injection de faute comme les "attaques au micro-ondes" (l'attaquant plaçant la carte à puce dans un four à micro-ondes pour lui faire calculer des résultats erronés). Au delà de cette vision un peu caricaturale, il est reconnu que des rayonnements correctement dirigés peuvent influencer le comportement de la carte.

- La *lumière* : Skorobogatov et Anderson [104] ont récemment observé que l'illumination d'un transistor peut le faire basculer temporairement dans son état conducteur, provoquant ainsi une erreur. En appliquant une source de lumière intense (produite par une lampe flash d'appareil photographique, amplifiée par un microscope), ils ont pu changer la valeur de bits individuels dans une mémoire SRAM (*Static* RAM). Par la même technique, ils ont pu également interférer avec les instructions `jump`, perturbant ainsi des sauts conditionnels.

- Les *courants de Foucault* (*Eddy currents*) : Quisquater et Samyde [88] ont également montré récemment que les courants de Foucault induits par un champ magnétique dans une bobine peuvent produire par exemple provoquer des erreurs dans une cellule de mémoire (qu'elle soit de type RAM, EPROM, EEPROM ou Flash).

À l'inverse, dans une attaque *passive*, l'adversaire se contente d'observer le comportement de la carte dans son fonctionnement normal. Là encore, plusieurs types d'informations de nature physique sont potentiellement utilisables par l'attaquant :

- Le *temps d'exécution* : le temps pris par un système pour exécuter un algorithme est parfois variable. Cela peut être dû à certaines instructions dont le temps d'exécution

[7] Nous empruntons certains éléments de ce paragraphe à l'étude [86] de J.-J. Quisquater et F. Koeune, ainsi qu'à celle coordonnée par E. Oswald dans [73].

[8] Blömer et Seifert [13] remarquent à ce sujet : "a finely tuned clock glitch is able to completely change a CPU's execution behavior including the omitting of instructions during the executions of programs".

dépend des données, ou bien à des optimisations du compilateur, ou encore à l'existence de plusieurs branches dans l'algorithme. L'idée des *attaques temporelles* (*timing attacks* qui en découlent a été publiée par Paul Kocher en 1996 [61], avant d'être appliquée à de nombreux cryptosystèmes tels que DES [50], AES [95], RSA [31, 94] ou les schémas à base de courbes elliptiques [66, 65, 77].

- La *consommation électrique* : la plupart des micro-modules actuels s'appuient sur une logique CMOS (*Complementary Metal-Oxyde Semiconductor*), dont on peut caractériser la consommation ainsi : à chaque coup d'horloge, les portes logiques changent d'état simultanément, provoquant le chargement ou le déchargement des capacités internes, ce qui se traduit par une variation de l'intensité du courant, mesurable de l'extérieur. En pratique, l'attaquant utilise soit une carte d'acquisition de données, soit un oscilloscope numérique pour collecter les données. L'intensité du courant peut être mesurée soit directement avec une sonde, soit en branchant une résistance en série avec la masse ou l'entrée de l'alimentation de la carte.

- Le *rayonnement électromagnétique* : le chargement et le déchargement des capacités des portes logiques a également pour conséquence la création d'un champ électromagnétique. On distingue les émanations directes, dues au courant qui circule lors de l'exécution de l'algorithme, et les émanations indirectes, provoquées par des effets de couplage entre des composants très proches[9]. Des mises en œuvre concrètes d'attaques électromagnétiques sont décrites dans [37, 87, 2].

9.4.2 Attaques par injection de fautes

Si les *timing attacks* sont les attaques passives les plus faciles à mettre en œuvre[10], c'est en essayant de provoquer des erreurs de calcul que l'on obtient les attaques actives les plus simples à mener (et les moins coûteuses). Ces attaques constituent d'ailleurs une menace non seulement pour les algorithmes cryptographiques, mais aussi pour d'autres composantes logicielles, comme les *machines virtuelles* Java[11], ou plus globalement le *système d'exploitation* dans son ensemble, pour lequel L. Goubin a récemment proposé une méthode générique de protection semi-automatique avec M.-L. Akkar et O. Ly [4].

En ce qui concerne les cryptosystèmes, c'est en septembre 1996 que trois chercheurs de Bellcore, Boneh, DeMillo et Lipton proposent un nouveau modèle d'attaque physique sur les cartes à microprocesseur, qu'ils baptisent "cryptanalysis in the presence of hardware faults" [15, 16, 17]. Ce modèle d'attaque est dirigé contre plusieurs algorithmes cryptographiques à clé publique : le schéma de signature RSA et les schémas d'authentification de Fiat-Shamir ou de Schnorr.

Dans le cas de la signature RSA, les auteurs montrent que :

- si l'implémentation utilise le théorème des restes chinois (CRT), une signature erronée et la signature correcte correspondante suffisent à factoriser le module ;

[9] La miniaturisation et la complexification des technologies CMOS ne font qu'accentuer ce phénomène.

[10] D. Boney et D. Brumley [14] ont même montré récemment qu'elles peuvent s'appliquer "à distance", en attaquant (au travers d'un réseau local) un serveur utilisant le protocole SSL.

[11] Voir à ce sujet l'article récent de S. Govindavajhala et A.W. Appel [40].

- si l'implémentation n'utilise pas les restes chinois, l'attaque peut fonctionner avec un nombre de signature erronées de l'ordre du nombre de bits du module.

Très peu de temps après, Lenstra, puis Joye et Quisquater [64, 56] remarquent que, pour une implémentation du RSA avec CRT, il suffit d'avoir un message et une signature erronée de ce message pour retrouver la factorisation du modulo. Le cas du RSA, ainsi que des algorithmes de signature ElGamal, Schnorr et DSA, est également étudié dans [8, 57]. Pour RSA, des contre-mesures ont été proposées dans [101, 102, 110, 7].

Biham et Shamir s'intéressent ensuite au cas des algorithmes cryptographiques à clé secrète. Ils montrent [11], que l'algorithme DES est aussi potentiellement vulnérable aux attaques "à la Bellcore", qu'ils rebaptisent *Differential Fault Analysis* (DFA). Il faut pour cela réussir, pour 200 exécutions de l'algorithme, à perturber à chaque fois un bit, et à récupérer les 200 messages chiffrés erronés. Pour un modèle d'attaque légèrement différent, Anderson et Kuhn [5, 6] réduisent à 10 le nombre de messages erronés nécessaires pour retrouver la clé secrète du DES. Ils supposent que l'on peut perturber la carte de manière à ce qu'une instruction assembleur bien choisie ne soit pas exécutée par le microprocesseur.

Tous les algorithmes cryptographiques sont potentiellement vulnérables aux attaques par injection de fautes. Ainsi Biehl, Meyer, Müller [10] et Ciet, Joye [21] ont montré que les attaques du type *Differential Fault Analysis* peuvent également être efficaces contre les cryptosystèmes à base de courbes elliptiques. Le cas de l'AES a quant à lui été étudié par Blömer, Seifert [13], Giraud [38], Dusart, Letourneux, Vivolo [33] et Piret, Quisquater [84].

Par ailleurs, les attaques DFA ont été généralisées dans plusieurs directions. Ainsi Biham et Shamir [11] montrent que dans certains cas on peut retrouver la clé secrète, ceci même sans connaître la spécification de l'algorithme implémenté dans la carte. Paillier a poursuivi l'étude de ce modèle d'attaque dans [82]. Par ailleurs Joye, Quisquater, Yen et Yung montrent que vérifier la justesse du résultat des calculs cryptographiques n'est pas toujours suffisant [109], et que cela peut même parfois aider l'attaquant [58].

9.4.3 Attaques par analyse de consommation électrique

Les attaques passives, définies au paragraphe 2.2, présentent l'avantage (pour l'attaquant) de ne pas perturber le fonctionnement du système. Certaines d'entre elles ont connu depuis quelques années un retentissement tout particulier. Il s'agit des attaques par *analyse de consommation électrique* (*power analysis* en anglais).

Dans [39], L. Goubin et J. Patarin ont étudié en détail les attaques utilisant le principe d'*analyse différentielle de consommation* (*Differential Power Analysis* – ou DPA – en anglais).

Analyse élémentaire de la consommation

On appelle généralement *trace* la courbe de consommation électrique (en fonction du temps) obtenue pour une exécution d'un algorithme cryptographique. Le principe de l'*analyse élémentaire de consommation* (*Simple Power Analysis* – ou SPA – en anglais) consiste à

essayer d'obtenir des informations sur la clé secrète de l'algorithme à partir d'une seule trace. Ceci n'est possible que si les variations de consommation électrique sont suffisamment importantes pour être décelées visuellement de manière directe. En outre, cela suppose qu'il y ait un lien simple et exploitable entre les informations observées et la clé secrète elle-même. C'est pourquoi ce type d'attaque vise particulièrement les implémentations qui utilisent des branchements dépendant de la clé.

Pour mener à bien une attaque SPA, on utilise souvent le fait que les valeurs de la consommation électrique sont fortement corrélées au *poids de Hamming* des données manipulées par les instructions assembleur. Cette idée a été développée par Biham et Shamir [12] pour le DES, par Mangard [67] pour l'AES, ou encore par Rao, Rohatgi et Scherzer [91] pour l'algorithme COMP128-1 [41, 19] utilisé dans le standard GSM de téléphonie mobile[12]. Par ailleurs Schramm, Wollinger et Paar ont récemment montré que le principe de la SPA pouvait également être utilisé pour détecter des collisions internes, notamment dans l'algorithme DES : ce sont les *attaques par collision* (*collision attacks*) [97].

Les algorithmes asymétriques sont aussi potentiellement vulnérables face aux attaques de type SPA. En particulier pour les algorithmes (tels RSA) faisant intervenir l'*exponentiation* d'une valeur connue, par un exposant secret, un scénario d'attaque utilisant à nouveau la notion de *poids de Hamming* est décrit par Klíma et Rosa dans [59]. De même, la *multiplication scalaire* d'un point connu d'une courbe elliptique, par un scalaire secret, donne lieu à des attaques : ainsi Oswald [81] utilise le *modèle de Markov* pour retrouver la clé dans le cas d'une implémentation avec des *chaînes d'addition-soustraction*. Clavier et Joye [22] ont établi un modèle général de ces attaques SPA, ouvrant la voie à des protections génériques et prouvées sûres pour ce type d'algorithmes asymétriques.

Analyse différentielle de la consommation

L'*analyse différentielle de consommation* (*Differential Power Analysis* – ou DPA – en anglais) a été introduite par Kocher, Jaffe et Jun en 1998 [62] et publiée en 1999 [63]. L'idée est d'exploiter les *corrélations* éventuelles entre les données manipulées par le microprocesseur et les valeurs instantanées de consommation électrique. Comme ces corrélations sont souvent très faibles, il faut avoir recours à des méthodes *statistiques* pour en tirer le maximum d'information.

Dans une attaque de type DPA, le principe consiste à comparer des valeurs mesurées lors du fonctionnement du *véritable* dispositif physique (par exemple la carte à microprocesseur) avec des valeurs calculées grâce à un modèle *hypothétique* de ce dispositif (les hypothèses portant notamment sur la nature de l'implémentation, et surtout sur une partie de la clé secrète). En comparant ces deux ensembles de valeurs, on s'efforce ensuite de retrouver tout ou partie de la clé secrète.

Les cibles initiales des attaques DPA étaient les algorithmes symétriques. La vulnérabilité du DES – mise en évidence par Kocher, Jaffe, Jun [62, 63] – a été étudiée plus en détail dans

[12] L'algorithme COMP128-1 avait de toute façon déjà succombé à la cryptanalyse de Briceno, Goldberg et Wagner [18, 46] en 1998.

mon article [39] écrit avec Patarin. Sur ce thème on peut citer aussi Messerges, Dabbish, Sloan [68] et Akkar, Bevan, Dischamp, Moyart [3]. L'application de ces attaques a été aussi largement prise en compte au cours du processus de sélection de l'AES, notamment par Biham, Shamir [12], Chari, Jutla, Rao, Rohatgi [20] et Daemen, Rijmen [28].

Mais les algorithmes asymétriques ne sont pas à l'abri non plus : L. Goubin et J. Patarin ont ainsi montré dans [39] (tout comme Messerges, Dabbish, Sloan dans [69]) comment appliquer la DPA sur l'algorithme RSA, et le cas des courbes elliptiques a été analysé par Coron [23], Okeya, Sakurai [78], puis par beaucoup d'autres (voir paragraphe 6).

Exemple de l'algorithme DES

Considérons, pour illustrer les attaques par analyse différentielle de consommation, l'exemple de l'algorithme de chiffrement DES. Celui-ci s'exécute en 16 étapes, appelées *tours* (ou *rounds* en anglais). Lors de chacun de ces tours, une transformation F est appliquée sur 32 bits. Cette fonction F utilise elle-même huit transformations non-linéaires de 6 bits sur 4 bits, chacune d'entre elles étant stockée dans une table appelée *boîte*-S (S-*box*). On peut alors mettre en œuvre une attaque DPA de la manière suivante, telle que Goubin et Patarin l'ont décrite dans [39] (le nombre 1000 est donné à titre d'exemple).

Première étape : On mesure la trace de consommation électrique du premier tour, ceci pour 1000 exécutions du DES. On désigne par E_1, ..., E_{1000} les valeurs d'entrée de ces 1000 calculs, et par C_1, ..., C_{1000} les 1000 courbes de consommation mesurées au cours de ces calculs. À partir de là, on calcule la *courbe moyenne* MC de ces 1000 courbes de consommation.

Deuxième étape : On se focalise par exemple sur le premier bit de sortie de la première boîte-S, lors du premier tour. Soit b la valeur de ce bit. Il est facile de voir que b ne dépend que de 6 bits de la clé secrète. L'attaquant fait alors une hypothèse sur les 6 bits impliqués. Il peut calculer – à partir de ces 6 bits et des E_i – la valeur (théorique) attendue pour b, et ainsi séparer les entrées E_1, ..., E_{1000} en deux catégories : celles qui donnent $b = 0$ et celles qui donnent $b = 1$.

Troisième étape : On calcule maintenant la courbe moyenne MC' des courbes correspondant aux entrées de la première catégorie (*i.e.* celles pour lesquelles $b = 0$). Si MC et MC' présentent, en un certain point, une différence appréciable (au sens statistique, *i.e.* une différence nettement plus grande que l'écart-type du bruit mesuré), l'attaquant en déduit que la valeur choisie pour les 6 bits de clé étaient corrects. En revanche, si MC et MC' ne présentent pas de différence notable, on recommence la deuxième étape avec un autre choix pour les 6 bits[13].

[13] En pratique, pour chaque choix des 6 bits, on trace la courbe représentant la différence entre MC et MC'. On obtient ainsi 64 courbes, parmi lesquelles une est supposée présenter des "pics" de consommation, aisément reconnaissables par rapport aux autres courbes.

Quatrième étape : On répète les étapes 2 et 3, avec un bit "cible" b dans la deuxième boîte-S, puis dans la troisième boîte-S, ..., jusqu'à la huitième boîte-S. En tout, cela permet de retrouver 48 bits de la clé secrète.

Cinquième étape : Les 8 bits manquants peuvent être trouvés par recherche exhaustive (ou bien en considérant le deuxième tour, avec exactement la même méthodologie d'attaque)

Cette attaque ne nécessite aucune connaissance *a priori* sur les valeurs de consommation individuelles de chaque instruction, ni sur la position dans le temps de ces instructions. La DPA peut s'appliquer à partir du moment où l'attaquant connaît les entrées (ou les sorties) de l'algorithme, ainsi que les courbes de consommation correspondantes. Elle s'appuie uniquement sur l'hypothèse suivante, que Goubin et Patarin ont mise en évidence dans [39] :

Hypothèse fondamentale : *Il existe une variable intermédiaire, qui apparaît au cours de l'exécution de l'algorithme, telle que la connaissance d'un petit nombre de bits de clé (en pratique moins de 32 bits) permet de décider si deux entrées (respectivement deux sorties) de l'algorithme donnent ou non la même valeur pour cette variable.*

Références

[1] D.G. Abraham, G.M. Dolan, G.P. Double, J.V. Stevens, *Transaction Security System*. IBM Systems Journal, Vol. 30, No 2, pp 206-229, 1991.

[2] D. Agrawal, B. Archambeault, J.R. Rao, P. Rohatgi, *The EM Side-Channel(s)*. In Proceedings of CHES'2002, LNCS 2523, pp. 29-45, Springer-Verlag, 2002.

[3] M.-L. Akkar, R. Bevan, P. Dischamp, D. Moyart, *Power Analysis: What is now Possible*. In Proceedings of ASIACRYPT'2000, LNCS 1976, pp. 489-502, Springer-Verlag, 2000.

[4] M.-L. Akkar, L. Goubin, O. Ly, *About an Automatic Fault Injection Protection System*. In Proceedings of E-Smart'2003, Nice, 2003.

[5] R.J. Anderson, M.G. Kuhn, *Improved Differential Fault Analysis*. Manuscrit, 20 novembre 1996. Disponible sur `http://www.ftp.cl.cam.ac.uk/ftp/users/rja14/dfa`

[6] R.J. Anderson, M.G. Kuhn, *Low Cost Attacks on Tamper Resistant Devices*. In Proceedings of Security Protocols, 5th International Workshop, Paris, France, April 7-9, 1997, LNCS 1361, pp. 125-136, Springer-Verlag, 1997.

[7] C. Aumüller, P. Bier, W. Fischer, P. Hofreiter, J.-P. Seifert, *Fault Attacks on RSA with CRT: concrete results and practical countermeasures*. In Proceedings of CHES'2002, LNCS 2523, pp. 260-275, Springer-Verlag, 2002.

[8] F. Bao, R.H. Deng, Y. Han, A. Jeng, A. Narasimhalu, T. Ngair, *Breaking Public-Key Cryptosystems on Tamper-Resistant Devices in the Presence of Transient Faults*. In Proceedings of Security Protocols, 5th International Workshop, Paris, France, April 7-9, 1997, LNCS 1361, pp. 115-124, Springer-Verlag, 1997.

[9] P. Barrett, *Implementing the Rivest, Shamir and Adleman Public-Key Encryption Algorithm on a Standard Digital Signal Processor*. In Proceedings of CRYPTO'86, LNCS 263, pp. 311-323, Springer-Verlag, 1987.

[10] I. Biehl, B. Meyer, V. Müller, *Differential Fault Attacks on Elliptic Curve Cryptosystems*. In Proceedings of CRYPTO'2000, LNCS 1880, pp. 131-146, Springer-Verlag, 2000.

[11] E. Biham, A. Shamir, *Differential Fault Analysis of Secret Key Cryptosystems*. In Proceedings of CRYPTO'97, LNCS 1294, pp. 513-528, Springer-Verlag, 1997.

[12] E. Biham, A. Shamir, *Power Analysis of the key scheduling of the AES candidates*. In Proceedings of the Second Advanced Encryption Standard Conference, NIST, 1999.

[13] J. Blömer, J.-P. Seifert, *Fault based cryptanalysis of the advanced encryption standard (AES)*. IACR ePrint Archive, 2002/075. Disponible sur `http://eprint.iacr.org/2002/075/`.

[14] D. Boneh, D. Brumley, *Remote timing attacks are practical*. À paraître in Proceedings of the 14th USENIX Security Symposium. Disponible sur `http://crypto.stanford.edu/~dabo/abstracts/ssl-timing.html`

[15] D. Boneh, R.A. DeMillo, R.J. Lipton, *New Threat Model Breaks Crypto Codes*. Bellcore Press Release, September 25th, 1996.

[16] D. Boneh, R.A. DeMillo, R.J. Lipton, *On the Importance of Checking Cryptographic Protocols for Faults*. In Proceedings of EUROCRYPT'97, LNCS 1233, pp. 37-51, Springer-Verlag, 1997.

[17] D. Boneh, R.A. DeMillo, R.J. Lipton, *On the Importance of Eliminating Errors in Cryptographic Computations*. Journal of Cryptology, Vol. 14, n°2, pp. 101-119, 2001.

[18] M. Briceno, I. Goldberg, D. Wagner, *GSM Cloning*. Disponible sur `http://www.isaac.cs.berkeley.edu/isaac/gsm-faq.html`

[19] M. Briceno, I. Goldberg, D. Wagner, *An implementation of the GSM A3A8 algorithm (specifically COMP128)*. Disponible sur `http://www.mirrors.wiretapped.net/security/cryptography/algorithms/gsm/a3a8.txt`

[20] S. Chari, C.S. Jutla, J.R. Rao, P. Rohatgi, *A Cautionary Note Regarding Evaluation of AES Candidates on Smart-Cards*. In Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference, NIST, 1999.

[21] M. Ciet, M. Joye, *Elliptic Curve Cryptosystems in the Presence of Permanent and Transient Faults*. IACR ePrint Archive, 2003/028. Disponible sur `http://eprint.iacr.org/2003/028/`

[22] C. Clavier, M. Joye, *Universal Exponentiation Algorithm – A First Step towards Provable SPA-Resistance*. In Proceedings of CHES'2001, LNCS 2162, pp. 300-308, Springer-Verlag, 2001.

[23] J.-S. Coron, *Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems*. In Proceedings of CHES'99, LNCS 1717, pp. 292-302, Springer-Verlag, 1999.

[24] N.T. Courtois, L. Goubin, J. Patarin, *FLASH, a fast multivariate signature algorithm*. In Proceedings of CT-RSA'2001, LNCS 2020, pp. 298-307, Springer-Verlag, 2001.
Note: Le schéma de signature SFLASH a été révisé depuis, voir [25].

[25] N.T. Courtois, L. Goubin, J. Patarin, *SFLASH, a fast asymmetric signature scheme for low-cost smartcards*. Version révisée, octobre 2001. Disponible sur `http://www.cryptonessie.org`

[26] N.T. Courtois, L. Goubin, J. Patarin, *QUARTZ, 128-bit long digital signatures*. In Proceedings of CT-RSA'2001, LNCS 2020, pp. 282-297, Springer-Verlag, 2001.
Note: Le schéma de signature QUARTZ a été révisé depuis, voir [27].

[27] N.T. Courtois, L. Goubin, J. Patarin, *QUARTZ, an asymmetric signature scheme for shot signatures on PC*. Version révisée, octobre 2001. Disponible sur `http://www.cryptonessie.org`

[28] J. Daemen, V. Rijmen, *Resistance Against Implementation Attacks: A Comparative Study of the AES Proposals*. In Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference, NIST, 1999.

[29] D. de Waleffe, J.-J. Quisquater, *CORSAIR, A Smart Card for Public-Key Cryptosystems*. In Proceedings of CRYPTO'90, LNCS 537, pp. 503-513, Springer-Verlag, 1990.

[30] J.-F. Dhem, *Design of an Efficient Public-Key Cryptographic Library for RISC-based Smart Cards*. Ph.D Thesis, UCL, 1998.

[31] J.-F. Dhem, F. Koeune, P.-A. Leroux, P. Mestré, J.-J. Quisquater, J.-L. Willems, *A practical implementation of the timing attack*. In Proceedings of CARDIS'98, LNCS 1820, pp. 167-182, Springer-Verlag, 1998. Disponible sur `http://www.dice.ucl.ac.be/crypto/techreports.html`

[32] J.-F. Dhem, J.-J. Quisquater, *Recent results on modular multiplications for smart cards*. In Proceedings of CARDIS'98, LNCS 1820, pp. 336-352, Springer-Verlag, 1998. Disponible sur `http://users.belgacom.net/dhem/papers/mulmod_cardis98.pdf`

[33] P. Dusart, G. Letourneux, O. Vivolo, *Differential Fault Analysis on A.E.S.*. IACR ePrint Archive, 2003/010. Disponible sur `http://eprint.iacr.org/2003/010/`

[34] T. ElGamal, *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*. IEEE Transactions on Information Theory, Vol. 31, pp. 469-472, 1985.

[35] A. Fiat, A. Shamir, *How to Prove Yourself: Practical Solutions of Identification and Signature Problems*. In Proceedings of CRYPTO'86, LNCS 263, pp. 186-194, Springer-Verlag, 1987.

[36] E. Fujisaki, T. Kobayashi, H. Morita, H. Oguro, T. Okamoto, S. Okazaki, *ESIGN: Efficient digital signature scheme (submission to NESSIE)*. Primitive submitted to NESSIE by NTT Corp., septembre 2000. Disponible sur `http://www.cryptonessie.org`

[37] K. Gandolfi, C. Mourtel, F. Olivier, *Electromagnetic Attacks: Concrete Results*. In Proceedings of CHES'2001, LNCS 2162, pp. 251-261, Springer-Verlag, 2001.

[38] C. Giraud, *DFA on AES*. IACR ePrint Archive, 2003/008. Disponible sur `http://eprint.iacr.org/2003/008/`

[39] L. Goubin, J. Patarin, *DES and Differential Power Analysis*. In Proceedings of CHES'99, LNCS 1717, pp. 158-172, Springer-Verlag, 1999.

[40] S. Govindavajhala, A.W. Appel, *Using Memory Errors to Attack a Virtual Machine*. IEEE Symposium on Security and Privacy, Oakland, 2003. Disponible sur `http://www.cs.princeton.edu/~sudhakar/papers/memerr.pdf`

[41] GSM Association, *Functional Description of the One Way Function COMP128 for Subscriber Authentication and Key Generation in the Paneuropean Mobile Communication System*. Document classifié, mais dont le contenu a été divulgué dans [19].

[42] L.C. Guillou, J.-J. Quisquater, *A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory*. In Proceedings of EUROCRYPT'88, LNCS 330, pp. 123-128, Springer-Verlag, 1988.

[43] L.C. Guillou, M. Ugon, *Smart Card, a Highly Reliable and Portable Security Device*. In Proceedings of CRYPTO'86, LNCS 263, pp. 464-479, Springer-Verlag, 1987.

[44] L.C. Guillou, M. Ugon, J.-J. Quisquater, *The smart card, a standardized security device dedicated to public cryptography*. In [103], chapitre 13, pp. 561-613, IEEE Press, Piscataway, 1992.

[45] L.C. Guillou, M. Ugon, J.-J. Quisquater, *Cryptographic authentication protocols for smart cards*. Computer Networks 36(4), pp. 437-451, 2001.

[46] H. Handschuh, P. Paillier, *Reducing the collision probability of Alleged Comp128*. In proceedings of CARDIS'98, LNCS 1820, pp. 366-371, Springer-Verlag, 1998.

[47] H. Handschuh, P. Paillier, *Smart Card Crypto-Coprocessors for Public-Key Cryptography*. In Cryptobytes, vol. 4, n°1, RSA Laboratories, 1998. Disponible sur `ftp://ftp.rsasecurity.com/pub/cryptobytes/crypto4n1.pdf`

[48] H. Handschuh, P. Paillier, *Smart Card Crypto-Coprocessors for Public-Key Cryptography*. In Proceedings of CARDIS'98, LNCS 1820, pp. 386-394, Springer-Verlag, 2000.

[49] J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. Silverman, W. Whyte, *NTRUSign: Digital Signatures Using the NTRU Latice*. In Proceedings of CT-RSA'2003, LNCS 2612, Springer-Verlag, 2003.

[50] A. Hevia, M.A. Kiwi, *Strength of two data encryption standard implementations under timing attacks*. ACM Transactions on Information and System Security (TISSEC), Vol. 2, pp. 416-437, 1999.

[51] IEEE P1363-2000, *Standard Specifications for Public-Key Cryptography*, août 2000. Disponible sur `http://standards.ieee.org/catalog/olis/busarch.html`

[52] ISO/IEC 7816-1, *Information Technology – Security Techniques – Integrated circuit(s) cards with contacts – Part 1: Physical characteristics*, 1998.

[53] ISO/IEC 7816-2, *Information Technology – Security Techniques – Integrated circuit(s) cards with contacts – Part 2: Contact locations and minimum size*, 1998.

[54] ISO/IEC 7816-3, *Information Technology – Security Techniques – Integrated circuit(s) cards with contacts – Part 3: Electronic signals and transmission protocols*, 1989.

[55] ISO/IEC 7816-4, *Information Technology – Security Techniques – Integrated circuit(s) cards with contacts – Part 4: Interindustry commands for interchange*, 1995.

[56] M. Joye, A.K. Lenstra, J.-J. Quisquater, *Chinese Remaindering Based Cryptosystems in the Presence of Faults*. Journal of Cryptology, Vol. 12, n. 4, pp. 241-245, 1999.

[57] M. Joye, J.-J. Quisquater, F. Bao, R.H. Deng, *RSA-type Signatures in the Presence of Transient Faults*. In Proceedings of the 6th IMA International Conference on Cryptography and Coding, Cirencester (U.K.), 17-19th December 1997, LNCS 1355, pp. 155-160, Springer-Verlag, 1997.

[58] M. Joye, J.-J. Quisquater, S.-M. Yen, M. Yung, *Observability analysis: Detecting when improved cryptosystems fail*. In Proceedings of CT-RSA'2002, LNCS 2271, pp. 17-29, Springer-Verlag, 2002.

[59] V. Klíma, T. Rosa, *Further Results and Considerations on Side Channel Attacks on RSA*. In Proceedings of CHES'2002, LNCS 2523, pp. 244-259, Springer-Verlag, 2002.

[60] D. Knuth, *The Art of Computer Programming*, vol. 2, Seminumerical Algorithms, Addison-Wesley, Reading MA, 1969.

[61] P.C. Kocher, *Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems*. In Proceedings of CRYPTO'96, LNCS 1109, pp. 104-113, Springer-Verlag, 1996.

[62] P. Kocher, J. Jaffe, B. Jun, *Introduction to Differential Power Analysis and Related Attacks*. Technical Report, Cryptography Research Inc., 1998. Disponible sur `http://www.cryptography.com/dpa/technical/index.html`

[63] P. Kocher, J. Jaffe, B. Jun, *Differential Power Analysis*. In Proceedings of CRYPTO'99, LNCS 1666, pp. 388-397, Springer-Verlag, 1999.

[64] A.K. Lenstra, *Memo on RSA Signature Generation in the Presence of Faults*. Manuscrit, 28 septembre 1996. Disponible auprès de l'auteur.

[65] C.H. Lim, H.S. Hwang, *Fast Implementation of Elliptic Curve Arithmetic in GF(p^m)*. In Proceedings of PKC'2000, LNCS 1751, pp. 405-421, Springer-Verlag, 2000.

[66] J. López, R. Dahab, *Fast Multiplication on Elliptic Curves over GF(2^m) without Precomputation*. In Proceedings of CHES'99, LNCS 1717, pp. 316-327, Springer-Verlag, 1999.

[67] S. Mangard, *A simple power-analysis (SPA) attack on implementations of the AES key expansion*. In Proceedings of ICISC'2002, LNCS 2587, pp. 343-358, Springer-Verlag, 2002.

[68] T.S. Messerges, E.A. Dabbish, R.H. Sloan, *Investigations of Power Analysis Attacks on Smartcards*. In Proceedings of the USENIX Workshop on Smartcard Technology, pp. 151-161, mai 1999. Disponible sur `http://www.eecs.uic.edu/~tmesserg/papers.html`

[69] T.S. Messerges, E.A. Dabbish, R.H. Sloan, *Power Analysis Attacks of Modular Exponentiation in Smartcards*. In Proceedings of CHES'99, LNCS 1717, pp. 144-157, Springer-Verlag, 1999.

[70] P.L. Montgomery, *Modular Multiplication without Trial Division*. Mathematics of Computations, vol. 44 (170), pp. 519-521, 1985.

[71] D. Naccache, D. M'Raïhi, *Cryptographic Smart Cards*. IEEE Micro, pp. 14-24, juin 1996.

[72] D. Naccache, D. M'Raïhi, W. Wolfowicz, A. di Porto, *Are Crypto-Accelerators Really Inevitable ?* In Proceedings of EUROCRYPT'95, LNCS 1440, pp.404-409, Springer-Verlag, 1995.

[73] NESSIE, *NESSIE Security Report*. Rapport NES/DOC/ENS/WP5/D20, version 2.0, février 2003. Disponible sur `http://www.cryptonessie.org`

[74] NESSIE, *Performance of Optimized Implementations of the NESSIE Primitives*. Rapport NES/TOC/TEC/WP6/D21, version 2.0, février 2003. Disponible sur `http://www.cryptonessie.org`

[75] NIST, *Digital Signature Standard (DSS)*. Federal Information Processing Standards Publication (FIPS PUB) 186, National Institute of Standards and Technology, novembre 1994.

[76] NIST, *Digital Signature Standard (DSS)*. Federal Information Processing Standards Publication (FIPS PUB) 186-2, National Institute of Standards and Technology, janvier 2000. Disponible sur `http://csrc.nist.gov/publications/fips/fips186-2/fips186-2.pdf`

[77] K. Okeya, H. Kurumatani, K. Sakurai, *Elliptic Curves with the Montgomery-Form and Their Cryptographic Applications*. In Proceedings of PKC'2000, LNCS 1751, pp. 238-257, Springer-Verlag, 2000.

[78] K. Okeya, K. Sakurai, *Power Analysis Breaks Elliptic Curve Cryptosystem even Secure against the Timing Attack*. In Proceedings of INDOCRYPT'2000, LNCS 1977, pp. 178-190, Springer-Verlag, 2000.

[79] J. Omura, *A Public Key Cell Design for Smart Card Chips*. IT Workshop, Hawaii, USA, pp. 983-985, novembre 1990.

[80] S.B. Örs, E. Oswald, B. Preneel, *Power-Analysis Attacks on an FPGA – First Experimental Results*. In Proceedings of CHES'2003, LNCS 2779, pp. 35-50, Springer-Verlag, 2003.

[81] E. Oswald, *Enhancing simple power-analysis attacks on elliptic curve cryptosystems*. In Proceedings of CHES'2002, LNCS 2523, pp. 82-97, Springer-Verlag, 2002.

[82] P. Paillier, *Evaluating Differential Fault Analysis of Unknown Cryptosystems*. In Proceedings of PKC'99, LNCS 1560, pp. 235-244, Springer-Verlag, 1999.

[83] J. Patarin, *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): two new Families of asymmetric Algorithms*. In Proceedings of EUROCRYPT'96, LNCS 1070, pp. 33-48, Springer-Verlag, 1996.

[84] G. Piret, J.-J. Quisquater, *A Differential Fault Attack Technique Against SPN Structures, with Application to the AES and KHAZAD*. In Proceedings of CHES'2003, LNCS 2779, pp. 77-88, Springer-Verlag, 2003.

[85] J.-J. Quisquater, L.C. Guillou, *The new Guillou-Quisquater scheme*. In Proceedings of RSA'2000.

[86] J.-J. Quisquater, F. Koeune, *Survey of side channel attacks*. Rapport CRYPTREC numéro 1047, 2002. Disponible sur `http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1047_Side_Channel_report.pdf`

[87] J.-J. Quisquater, D. Samyde, *ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards*. In Proceedings of E-Smart'2001, LNCS 2140, pp. 200-210, Springer-Verlag, 2001.

[88] J.-J. Quisquater, D. Samyde, *Eddy current for magnetic analysis with active sensor*. In Proceedings of E-Smart'2002, Nice, 2002.

[89] M.O. Rabin, *Digitized Signatures and Public-Key Functions as Intractable as Factorization*. Technical Report LCS/TR-212, M.I.T. Laboratory for Computer Science, 1979.

[90] W. Rankl, W. Effing, *Smart card handbook*. John Wiley & Sons, 1997.

[91] J.R. Rao, P. Rohatgi, H. Scherzer, *Partitioning Attacks: Or How to Rapidly Clone Some GSM Cards*. IEEE Symposium on Security and Privacy, Oakland, 2002.

[92] R.L. Rivest, A. Shamir, L.M. Adleman, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*. Communications of the ACM, vol. 21, n°2, 1978, pp. 120-126.

[93] D. Sauveron, *La Technology Java Card^{TM} – Présentation de la carte à puce – La Java Card*. Rapport Interne RR-1259-01, LaBRI, Université de Bordeaux I, 2001.

[94] W. Schindler, *A combined timing and power attack*. In Proceedings of PKC'2002, LNCS 2274, pp. 263-279, Springer-Verlag, 2002.

[95] W. Schindler, J.-J. Quisquater, F. Koeune, *Improving divide and conquer attacks against cryptosystems by better error detection correction strategies*. In Proceedings of the 8th IMA International Conference on Cryptography and Coding, LNCS 2260, pp. 245-267, Springer-Verlag, 2001.

[96] B. Schneier, A. Shostack, *Breaking Up is Hard to Do: Modeling Security Threats for Smart Cards*. USENIX Workshop on Smart Card Technology, pp. 175-185, USENIX Press, octobre 1999. Disponible sur `http://www.counterpane.com/smartcard-threats.pdf`

[97] K. Schramm, T. Wollinger, C. Paar, *A new class of collision attacks and its application to DES*. In Proceedings of FSE'2003, LNCS 2887, Springer-Verlag, 2003.

[98] T. Schweinberger, V. Shoup, *ACE: The advanced cryptographic engine*. Primitive submitted to NESSIE, septembre 2000. Disponible sur `http://www.cryptonessie.org`

[99] H. Sedlak, *The RSA Cryptographic Processor : The First High Speed One-Chip Solution*. In Proceedings of EUROCRYPT'87, LNCS 293, pp. 95-105, Springer-Verlag, 1988.

[100] A. Shamir, *An Efficient Identification Scheme Based on Permuted Kernels*. In Proceedings of CRYPTO'89, LNCS 435, pp. 606-609, Springer-Verlag Verlag, 1990.

[101] A. Shamir, *How to check modular exponentiation*. Présenté à la Rump Session d'EUROCRYPT'97, Konstanz, Allemagne. Cette méthode est brevetée [102].

[102] A. Shamir, *Method and apparatus for protecting public key schemes from timing and fault attacks*. United States Patent 5991415, 23 novembre 1999.

[103] G.J. Simmons, *Contemporary Cryptology, The Science of Information Integrity*. IEEE Press, Piscataway, 1992.

[104] S. Skorobogatov, R.J. Anderson, *Optical fault induction attacks*. In Proceedings of CHES'2002, LNCS 2523, pp. 2-12, Springer-Verlag, 2002.

[105] M. Ugon, *L'Odyssée de la carte à puce*. Article paru dans *Le Monde*. Disponible sur `http://perso.wanadoo.fr/f1my/axtcp/cartapuc.htm`

[106] S.A. Vanstone, *Responses to NIST's proposal*. Communications of the ACM, vol. 35, pp. 50-52, juillet 1992.

[107] H.C. Williams, *A Modification of the RSA Public-Key Encryption Procedure*. IEEE Transactions on Information Theory, v.IT-26, n.6, 1980, pp. 726-729.

[108] T. Wollinger, C. Paar, *How Secure are FPGAs in Cryptographic Applications ?* IACR ePrint Archive, 2003/119. Disponible sur `http://eprint.iacr.org/2003/119/`

[109] S.-M. Yen, M. Joye, *Checking before output may not be enough against fault-based cryptanalysis*. IEEE Transactions on Computers, Vol. 49, No. 9, pp. 967-970, septembre 2000.

[110] S.-M. Yen, S. Kim, S. Lim, S. Moon, *RSA speedup with residue number system immune against hardware fault cryptanalysis*. In Proceedings of ICISC'2001, LNCS 2288, pp. 397-413, Springer-Verlag, 2001.