Contributions à la cryptographie à base de couplage.

Nadia El Mrabet

Université Paris 8 Vincennes Saint Denis

Laboratoire d'Informatique Avancé de Saint Denis

Mémoire d'habilitation

pour obtenir le diplôme d'

Habilitation à Diriger des Recherches

Discipline : **Informatique**

présentée et soutenue publiquement par

Nadia El Mrabet

le 8 décembre 2017

Contributions à la cryptographie à base de couplage

JURY

Bruno Martin, Univ. Nice	Rapporteur
Emmanuel Prouff, ANSSI	Rapporteur
Arab Ali Chérif, Univ. Paris 8	Rapporteur
Jean-Claude Bajard, Univ. Paris 6	Examinateur
Laurent-Stéphane Didier, Univ. Toulon	Examinateur
Sylvain Duquesne, Univ. Rennes	Examinateur
Wolfgang Schmid, Univ. Paris 8	Examinateur

Table des matières

Remerciements

Je viens de recevoir l'autorisation de soutenir ma HDR, je peux donc prendre ma plume, mon clavier en fait, pour remercier toutes les personnes qui ont rendu cette soutenance possible.

En premier lieu, mes remerciements s'adressent aux membres du jury : merci à Bruno et Emmanuel d'avoir accepté de rapporter mon manuscrit ; à Ali Chérif d'avoir accepté de présenter mon dossier auprès de la commission restreinte de l'Université Paris 8 ; à Laurent-Stéphane et Wolfgang d'avoir accepté de faire partie de mon jury et à Jean-Claude et Sylvain pour m'avoir permis de découvrir le monde de l'enseignement et de la recherche.

Dans un deuxième temps, je remercie les équipes de recherche et collègues académiques ou administratifs qui m'ont accueillie : l'équipe Arith de l'Université Montpellier 2, l'équipe algorithmique de l'Université de Caen, le LIASD à l'Université Paris 8 et SAS à l'École des Mines de Saint-Étienne.

Ensuite dans un troisième temps, ma soutenance n'aurait pu avoir lieu sans mes co-auteurs ou co-organisateurs de conférence, je ne peux tous les citer, mais chacune et chacun d'entre eux ont été source d'inspiration et de motivation. En particulier les doctorants avec lesquels j'ai eu la chance de travailler.

Enfin dans un quatrième et dernier temps (le plus important !) : ma famille. Mes parents, frères, soeurs, tantes, oncles, nièces, par alliance ou de naissance. Merci pour votre constance ! Je garde le meilleur pour la fin : mes trésors petits et grand, qui font ma joie et mon bonheur au quotidien[1].

[1] $1 + 1 + 162 + 163 + 124 = 1$

Chapitre I

Animation de la recherche

I.1 Présentation

Agrégée de mathématiques (2005), j'ai soutenu une thèse de doctorat en informatique, à l'université Montpellier 2, en 2009, sous la direction conjointe de Sylvain Duquesne et de Jean-Claude Bajard, suivie d'une année en post-doc, au sein des laboratoires GREYC (Groupe de Recherche en Informatique, Image, Automatique et Instrumentation de Caen) et LMNO (Laboratoire de Mathématiques Nicolas Oresme), à l'université de Caen, en 2009-2010.

Je suis devenue maître de conférence à l'université Paris 8 Vincennes-Saint-Denis dans le Laboratoire d'Informatique Avancée de Saint-Denis (LIASD) en septembre 2010, puis, en décembre 2014, j'ai été recrutée comme maître assistant de l'École des Mines de Saint-Étienne, au sein de l'équipe Systèmes et Architectures Sécurisés (SAS) du Campus George Charpak Provence à Gardanne.

Mes travaux de recherche s'articulent entre les théories mathématiques pour la cryptographie, les aspects d'implémentation logicielle et matérielle, ainsi que les descriptions et réalisations d'attaques par canaux auxiliaires. Je suis ainsi à l'interface entre les mathématiques, l'informatique et la micro-électronique.

I.2 Vie du laboratoire

J'ai toujours participé activement à la vie de mes laboratoires d'accueil, en particulier via l'animation de séminaires de recherche ou de groupes de travail. Ainsi j'ai co-organisé

- le séminaire de l'équipe Arith au LIRMM (Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier) durant mon doctorat avec Valérie Berthé, Thierry Monteil et Pascal Giorgi ;
- le séminaire des doctorants de l'I3M (Institut de Mathématiques et de Modélisation de Montpellier) à l'Université Montpellier 2 ;
- le séminaire hebdomadaire Crypto de l'Université de Caen commun au GREYC et LMNO avec Fabien Laguillaumie et Pierre Castel ;

- j'ai lancé, et animé, le séminaire hebdomadaire de l'équipe SAS.

J'ai participé à l'organisation des Doctiss'07 (`http://www.lirmm.fr/doctiss07/`), journée des doctorants de l'école doctorale I2S, de l'université Montpellier 2 le 19 avril 2007. Cette manifestation regroupait tous les doctorants de l'I2S pour une journée de vulgarisation scientifique, soit environ 150 doctorants.

I.3 Conférence et comité de sélection

I.3.1 Organisation de workshop

J'ai co-organisé le workshop Physic 2016 : Workshop on Practical Hardware Innovations in Security Implementation and Characterization, avec mes collègues du CEA (`http://phisic2016.emse.fr`). Le workshop regroupe des industriels et académiques autour de 2 jours d'exposés sur le thème de la sécurité des circuits imprimés. L'objectif est de permettre l'échange et le partage d'idées sur des projets de recherche avec l'ambition d'aboutir à des projets collaboratifs de type ANR, EU FP7, FUI, H2020. L'audience se compose d'environ 120 chercheurs et industriels.

Je fais partie du comité d'organisation du workshop Crypto'puce. La session 2017 (`https://cryptopuces2017.sciencesconf.org`) s'est déroulée en mai 2017 et a regroupé une trentaine de chercheurs juniors (doctorants et post-doctorants), chercheurs et industriels. Crypto'Puces est une rencontre Université-entreprises qui prend la forme d'un séminaire au cours duquel les doctorants peuvent présenter leurs travaux et nouer des contacts avec le monde de l'entreprise. Réciproquement, des entreprises, aussi bien grands groupes que PME, peuvent présenter leurs problématiques de R&D et rencontrer des chercheurs du monde académique.

I.3.2 Comité de sélection et review

Je suis membre des comités scientifiques ou de sélections des conférences suivantes : YACC 2016 ; Information Assurance and Security 2015 ; Codes, Cryptology and Information Security 2017 ; Proofs 2016, 2017 ; Africacrypt 2017, 2018.

J'effectue régulièrement des reviews pour les journaux : Finite Fields and Applications ; Designs, Codes and Cryptography ; Cryptography and Communications ; Journal of Mathematical Cryptology ; Transactions on Computers ; Journal of Symbolic Computation ; Information Processing Letters ; et pour les conférences : Inscrypt 2012 ; Cosade 2012, 2013, 2014 ; SAC 2012 ; FDTC 2012 ; Africacrypt 2012, 2013 ; CANS 2013 ; CHES 2013, 2014, 2015 ; ICITS 2013 ; IMA Cryptography and Coding 2015 ; Asiacrypt 2015 ; ACNS 2015 ; CT-RSA 2016.

I.4 Valorisation de la recherche

I.4.1 Animation, pilotage et ancrage local

Je fais partie du comité de pilotage du groupe de travail C2 : Code et Cryptographie (`https://crypto.di.ens.fr/c2:main`) du GDR IM du CNRS. Le groupe de

travail C2 a pour vocation de réunir d'une part les chercheurs du domaine des codes correcteur d'erreurs et plus généralement de la théorie de l'information, et ceux dont les intérêts concernent la cryptologie d'autre part.

Je participe activement à l'animation de la recherche au sein de la région PACA à travers les journées thématiques organisées par le pôle de compétitivité Solutions Communicantes Sécurisées (SCS) et l'animation d'un groupe de travail sur la sécurité des Big Data. En particulier, j'ai présenté des exposés lors des journées thématiques Internet des Objets, Big Data du pôle SCS (25 avril, 24 mai et 27 juillet 2016) ainsi que lors de forum organisé par ARCSIS (Association pour la Recherche sur les Composants et les Systèmes Intégrés Sécurisés) le 29 avril 2016.

Je suis oratrice invitée pour la conférence SENSO 2017 (`http://www.sensoconference.com`) et le workshop AMUSEC 2017 (`http://www.amusec.fr`).

I.4.2 Invitations en séminaires, conférences et ancrage national

J'ai été invitée lors des Journées Thématiques "Cryptographie : Applications Des Mathématiques à La Cryptographie" du laboratoire de mathématiques AGM (Laboratoire Analyse Géométrie Modélisation) de l'université de Cergy-Pontoise afin de présenter l'exposé intitulé *Les couplages : de la définition mathématique à l'utilisation cryptographique*, en octobre 2011. J'ai été oratrice invitée lors des journées RAIM 2013 (Rencontres Arithmétiques de l'Informatique Mathématique) du groupe de travail Arithmétique du GDR IM. Mon intervention se faisait lors de la session cryptographie pour présenter un état de l'art des attaques par fautes en cryptographie à base de couplage à l'IHP à Paris, en mai 2013.

I.5 Vulgarisation

J'ai animé des stands lors des journées portes ouvertes de l'I3M à l'université Montpellier 2 durant mon doctorat et du LIASD à l'université Paris 8.

J'ai co-écrit le chapitre Réflexions antiques et modernes sur les nombres premiers de l'ouvrage collaboratif "Pourquoi les mathématiques ?" paru aux Editions Ellipses en 2015 [62].

J'ai accueilli en juin 2016 les finalistes régionaux, étudiants en classe de seconde générale, du concours de cryptographie Al Kindi (`http://www.concours-alkindi.fr/#/`). Ils ont visité le laboratoire et ont assisté à une démonstration d'attaque par canaux auxiliaires sur les bancs de l'équipe SAS.

I.6 Enseignement liés à la recherche

I.6.1 Cours et formation continue

Cours de niveau master

Parmi les enseignements que j'ai dispensés, certains étaient liés à mes thématiques de recherche. J'ai mis en place une option de cryptographie dans le master informa-

tique de l'université Paris 8. Je suis intervenue dans le master 2 recherche : Master Mathématiques pour la protection de l'information (MPRI) de l'université Paris 8, en donnant les cours d'algorithmique pour la cryptographie.

Responsable d'option ISMIN

Je suis responsable de l'option de troisième année Mobilité et Sécurité du cursus ISMIN (Ingénieur Spécialité Microélectronique et Informatique, de l'École des Mines de Saint-Étienne) et j'y dispense des cours introductifs à la cryptographie moderne (clé publique, clé secrète, RSA et courbes elliptiques). J'ai organisé cette option afin de permettre aux futurs ingénieurs de maîtriser les outils nécessaires pour réaliser l'implémentation de protocoles cryptographiques sur des systèmes embarqués. Ils sont aussi sensibilisés aux aspects de cryptanalyse à travers la cybersécurité et les attaques par canaux auxiliaires. Par ailleurs, un de mes rôle en tant que responsable est de conseiller les étudiants sur leur recherche de stage et leur suite d'étude ou recherche d'emploi après l'obtention de leur diplôme d'ingénieur.

Responsable du Mastère spécialisé SISA

Je suis également responsable de la (re)mise en place du Mastère Spécialisé Security of Integrated Systems Applications SISA (MS SISA). Le MS SISA sera ouvert aux étudiants en poursuite d'étude après un bac +5 ou aux ingénieurs en activité souhaitant suivre une formation dans le cadre de leur compte personnel de formation. Le MS SISA sera ouvert en collaboration avec le pôle SCS et ARCSIS. En tant que responsable du MS SISA, j'interagis avec la direction de la pédagogie numérique de l'Institut Mine Telecom. En particulier, j'ai participé au montage du projet "cursus de formation toute au long de la vie innovant pour l'Industrie du Futur".

J'ai aussi participé à la création d'une formation du futur, au sein du projet Henri Fabre porté par le pôle de compétivité SAFE cluster. Je suis responsable du module sécurité et vie privée de l'axe de formation Big Data.

I.6.2 Écoles thématiques

École d'été à l'ENSIAS

J'ai été invitée pour co-animer une école d'hiver à l'ENSIAS en mars 2010 à Rabat, Maroc, avec Pierre Louis Cayrel. J'y ai donné des cours d'introduction à la cryptographie générale et à la cryptographie à base de couplage. L'assistance était composée d'environ 50 enseignants-chercheurs et chercheurs.

Formation à l'EPO

Invitation pour co-animer une formation d'une semaine à l'Office Européen des brevet (EPO, European Patent Office) à Munich, Allemagne, avec David Pointcheval et Pierre Alain Fouque en 2011. Le but de la formation était d'exposer, devant les 20 examinateurs de brevets du domaine de la sécurité informatique de l'EPO, l'état de l'art de la cryptographie aujourd'hui. J'étais en charge des présentations

concernant les courbes elliptiques, le logarithme discret, les couplages, l'implémentation des couplages et les réseaux, notamment d'une introduction au chiffrement homomorphique.

Séminaire TRUDEVICE 2016

J'ai été invitée à participer à la formation des jeunes chercheurs lors la training school 2016 du workshop TRUDEVICE 2016 : Workshop on Trustworthy Manufacturing and Utilization of Secure Devices à Leukerbad, Suisse. Cette école était organisée par la ICT COST Action IC1204. J'ai donné un cours d'introduction à la cryptographie sur courbes elliptiques.

I.7 Encadrement (stages, stages de recherche et thèses)

I.7.1 Stages de recherche

J'ai participé à l'encadrement d'étudiants durant leur stage de projet d'étude et projet industriel cursus ISMIN, de stage de fin d'étude d'étudiants ISMIN et ISFEN, de stage de Licence 3 Mathématiques dans le cadre d'un stage d'initiation à la recherche, de Master 1 Informatique dans leur projet du type TER.

Stages de fin d'étude, École des Mines de Saint-Étienne

J'ai encadré les stages de fin d'étude de

- Véronique Blanchet en stage en Nouvelle-Zélande sur le cloud computing du 2 mars au 28 aout 2015, cycle ISMIN ;
- Marine Cammarata en apprentissage chez ST Microelectronics, cycle ISFEN (Ingénieur de Spécialité Électronique et Informatique Industrielle, École des Mines de Saint-Étienne) 2015 ;
- Anthony Pedraja en apprentissage chez Orange, cycle ISFEN 2015 ;
- Timothé Riom en stage dans l'équipe SAS, cycle ISMIN 2016.

Stages de recherche, poursuivis par une thèse

J'ai encadré des étudiantes pour leur stage recherche :

- Meriem Smache, en 2015, pour son stage de fin d'étude de cycle ingénieur, sur la sécurité dans les systèmes de smart grid ;
- Hanane Zerdoum, en 2015 également, pour son stage de recherche de M2 MPRI (Université Paris 8), sur les implémentations de protocoles à bases de courbes elliptiques sur une carte ST32 ;
- Amina Bel Korchi, en 2016, pour son stage de recherche de M2 CRYPTIS (Université de Limoges) dans l'entreprise Kontron sur la thématique de la cryptographie homomorphique.

Ces trois étudiantes ont poursuivi leurs recherches par une inscription en doctorat, dont Meriem et Amina sous ma co-direction. Leurs sujets de thèse, dans la continuité de leurs sujets de stage de recherche, sont décris au chapitre IV.

Le sujet de stage d'Hanane, que j'ai co-encadré avec Driss Aboulkassimi du CEA, concernait mes travaux en arithmétique des corps finis : Hanane a étudié les problématiques liées à l'implémentation du protocole de cryptographie RSA sur une carte STM32. Nous avons choisi l'arithmétique des grands nombres pour calculer la multiplication et l'exponentiation sur un corps fini. Hanane a rédigé un tutoriel expliquant les démarches à suivre pour exécuter son implémentation. Cette dernière est utilisée par les étudiants de 3ème année du cycle ISMIN pour mener des attaques par canaux cachés. Hanane a continué ses études par une thèse en doctorat à l'université Paris 8, sous la direction de Wolfgang Schmid de l'équipe de mathématiques pour le traitement de l'information et de l'image du LAGA (Laboratoire Analyse, Géométrie et Applications).

Accueil de doctorant

En janvier 2012, j'ai accueilli le doctorant camerounais Emmanuel Fouatsa, dans le cadre des programmes courts séjours mathématiques du Centre International de Mathématiques Pures et Appliquées (CIMPA). Emmanuel étudiait la construction de primitives cryptographiques pour la cryptographie à base de courbes elliptiques. Cette visite s'est concrétisée par une collaboration qui perdure. Nous avons publié un article de journal [56], un article dans une conférence avec acte et comité de sélection [70], et un article en cours de soumissions [80].

I.7.2 Co-direction de thèses

Damien Jauvard

J'ai co-encadré Damien avec Louis Goubin (Université de Versailles Saint-Quentin-en-Yvelines) et Jacques Fournier (CEA) de septembre 2014 à septembre 2017. Damien a travaillé sur les attaques par canaux auxiliaires contre une implémentation sur carte STM 32 d'un couplage. Ses travaux ont donné lieu à 2 publications dans des conférences internationales avec acte et comité de sélection [110, 111] et à 1 chapitre de livre [73]. Sa soutenance est prévue le 20 septembre 2017.

Amine Mrabet

Avec Sihem Mesnager (LAGA, Université Paris 8), nous avons co-encadré Amine Mrabet en thèse de doctorat en co-tutelle entre l'université Paris 8 et l'université de Monastir, Tunisie, de septembre 2014 à septembre 2017. Amine a travaillé sur l'implémentation sur cartes programmables (FPGA) de primitives cryptographiques à base de courbes elliptiques. Ses travaux ont donné lieu à 1 article de journal [150], à 4 publications dans des conférences internationales avec actes et comité de sélection [148, 149, 151, 152] et à 1 chapitre de livre [6]. Sa soutenance est prévue le 8 novembre 2017.

Aminatou Pecha

Je co-encadre Aminatou Pecha, avec Wolfgang Schmidt (UP8), dans le cadre d'une thèse en cotutelle entre l'université Paris 8 et le laboratoire d'algèbre de l'Université de Yaoundé (Cameroun), de septembre 2014 à septembre 2017. Aminatou étudie la construction de primitives cryptographiques pour les protocoles à base de courbes elliptiques. Ses travaux ont donné lieu à 2 publications en cours d'expertise dans des journaux. Elle rédige son manuscrit, la soutenance est prévue avant la fin de l'année 2017.

Meriem Smache

Je co-encadre Meriem Smache en thèse Cifre dans l'entreprise GRIDBEE Communication avec Emmanuel Riou (Gridbee), Assia Tria et Pierre-Alain Moellic, tous deux du CEA. Meriem travaille depuis février 2016 à la sécurisation des communications dans les réseaux de Smart Grid. Ses travaux ont donné lieu à 1 publication dans une conférence internationale avec actes et comité de sélection [176].

Amina Bel Korchi

Je co-encadre Amina Bel Korchi en thèse Cifre dans l'entreprise Kontron avec Serge Tissot (Kontron) et Dominique Feuillet (École des Mines de Saint-Étienne) depuis novembre 2016. Amina travaille sur la réalisation pratique de la cryptographie homomorphique sur les modules cryptographiques de Kontron.

Asma Chaouch

Je co-encadre Asma Chaouch en co-tutelle entre l'université de Toulon et l'université de Monastir avec Laurent-Stéphane Didier depuis avril 2017. Le travail d'Asma concerne le chiffrement d'images par l'emploi de courbes elliptiques.

Les sujets de thèse des doctorants sont décrits plus précisément au chapitre IV.

Safia Haloui, post-doc

J'ai encadré Safia Haloui, post-doctorante dans le cadre du projet ANR INS 2012 SIMPATIC de septembre 2014 à avril 2016. Cette collaboration a donné lieu à 1 article de journal [54], à 2 soumissions dans des journaux et à 2 chapitres de livre [57, 97].

I.8 Projets collaboratifs

Je détaille dans cette section les projets collaboratifs auxquels j'ai participé. J'ai été membre de trois projets ANR : SIMPATIC (INS 2012), THE CASCADE (INS 2013) et CLÉ (JCJC 2013) et suis membre du projet FUI TEEVA 2016.

Les participations à (comme le montage de) ces projets collaboratifs m'ont permis d'élargir mes thématiques de recherche et de collaborer avec des chercheurs

de différents laboratoires. Mes travaux sont disponibles sur ma page web et via la plateforme Hal.

I.8.1 ANR SIMPATIC

L'ANR INS 2012 **SIM** and **PA**iring **T**heory for **I**nformation and **C**ommunications security (SIMPATIC) regroupait les partenaires suivants : ENS Paris, Université de Caen, Université de Bordeaux, Université Paris 8, Orange Lab, Oberthur, INVIA, ST-ERICSON. La durée du projet était de 42 mois. Dans le cadre de ce projet, j'ai recruté une post-doctorante et financé les séjours en France de mon doctorant Amine Mrabet.

L'objectif du projet ANR SIMPATIC était de proposer les implémentations matérielles et logicielles les plus efficaces possibles d'un couplage sur une carte SIM. Les couplages permettent d'assurer la confidentialité et l'authentification des services, par la création de signatures courtes ou l'utilisation de la cryptographie basée sur l'identité.

Le projet a permis de montrer que les couplages mathématiques peuvent être intégrés dans les cartes à puce actuellement déployées auprès du grand public, en obtenant des performances bien supérieures à l'état de l'art. Les outils cryptographiques issus du projet ont par ailleurs permis d'associer fonctionnalité et efficacité dans des domaines tels que la minimisation des données personnelles dans les services sans contact, la télévision payante, ou la protection des données stockées dans un cloud qui n'est pas de confiance.

J'ai essentiellement travaillé dans la tâche 2 du projet, dont j'étais par ailleurs responsable. Cette tâche 2 était subdivisée en trois axes.

- Le premier consistant à étudier les couplages comme objets mathématiques : de nouvelles constructions permettant de réduire la complexité algébrique du calcul des couplages (nouvelles courbes, nouvelles représentations des extensions de corps finis), inspirée de la géométrie algébrique, ont été proposées ;

- Le deuxième axe concerne l'algorithmique des couplages : quel couplage est le mieux adapté au besoin des autres partenaires ?

- Enfin le dernier axe est l'étude de la sécurité des couplages vis-à-vis des attaques par canaux cachés.

Les résultats des études théoriques effectuées dans la tâche 2 ont été transmis aux partenaires impliqués dans les applications matérielles ou logicielles de nos propositions pour exploitation et valorisation. Le bilan du projet est positif puisque les implémentations ont été réalisées et les temps de calcul correspondent à ce que nous espérions [163].

Les travaux effectués dans le cadre de ce projet se sont concrétisés par 1 livre dont je suis co-éditrice [65], 5 chapitres de livre [77, 24, 23, 57, 73], 4 articles de journaux [56, 71, 54, 150] et 10 articles de conférences internationales avec actes et comité de sélection [39, 128, 69, 70, 126, 148, 149, 151, 152, 188].

I.8.2 ANR THE CASCADE

L'ANR INS 2013 **THE**orie du **C**ontrôle **A**ppliquée à la **S**ynchronisation des **C**ommunic**A**tions **D**iscr**E**tes (THE CASCADE) regroupe les partenaires suivants : l'Université de Nancy, l'Université Paris 8 et Airbus Defense & Space. La durée du projet est de 42 mois, et je suis impliquée dans ce projet à hauteur de 11 hommes-mois. Le projet à débuté en février 2014.

Je me permets de reprendre ici, et ci-dessous, la description succincte du projet THE CASCADE. Il répond à un enjeu sociétal et économique : la garantie de la sécurité des échanges dans les communications numériques qui requièrent des solutions simples à faible coût ou demandant peu de ressources : systèmes haut débit, systèmes cyber-physiques. Dans cette perspective, le projet doit proposer de nouveaux chiffreurs parmi la classe des auto-synchronisants. Ces chiffreurs sont définis sur la base de générateurs calculant des suites de chiffrement symétriques. Les générateurs doivent se synchroniser avec l'émetteur pour le chiffrement et le récepteur pour le déchiffrement.

Le projet THE CASCADE propose de nouvelles architectures auto-synchronisantes pour réaliser cet objectif. L'originalité du projet repose dans l'emploi des systèmes de communications basés sur les réseaux radio mobile, sur le modèle du bluetooth et des tags RFID. Pour ce faire, nous étudions des combinaisons d'outils de la théorie du contrôle et de la cryptographie classique. En effet, la synchronisation peut être interprétée comme la reconstruction d'état et la récupération d'informations par le récepteur, en conséquence de quoi elle peut être analysée comme un problème d'inverse à gauche. Nous étudions des générateurs construits sur des systèmes dynamiques non-linéaires.

Je suis impliquée dans la tâche 3 de ce projet pour laquelle j'étudie les spécifications de tels générateurs et analyse la sécurité de ceux-ci vis-à-vis des attaques physiques (attaques par canaux cachés).

Certains des travaux menés dans le cadre du projet sont encore en cours de soumission, et je suis l'un des co-auteurs d'un article publié dans les actes d'une conférence internationale avec comité de sélection [95].

I.8.3 ANR CLÉ

L'ANR JCJ 2013 **C**ryptography from **L**earning with **E**rrors (CLE) regroupait les partenaires académiques suivants : ENS Paris, Université de Versailles Saint)Quentin-en-Yvelines, INRIA Paris-Rocquencourt, Université Catholique de Louvain et l'Université Paris 8. Le projet, qui devait durer 48 mois, a débuté en octobre 2013.

Le problème Learning With Error (LWE) est depuis peu un outil performant pour construire des primitives cryptographiques. Néanmoins, les protocoles proposés sont potentiellement très puissants mais difficiles à réaliser en pratique. Le but du projet CLÉ était d'explorer les applications pratiques du problème LWE et de proposer au bout des 4 ans des schémas à clé publique ou clé privée facilement implémentables. En particulier, nous allions étudier des schémas de signature utilisant la transformation de Fiat et Shamir, qui produit des sorties dont nous espérions réduire la taille en utilisant le problème LWE.

Malheureusement, le porteur de l'ANR ayant quitté l'ENS pour travailler dans le secteur privé, le financement a été suspendu peu après le kick off. C'est un vrai regret personnel tant il est difficile d'obtenir des financements et parce que les sujets à traiter étaient d'un grand intérêt.

I.8.4 FUI TEEVA

Le consortium du projet FUI TEEVA (pour Trusted Execution EVAluation) regroupe les partenaires suivants : Gemalto, Trustonic, PhoneSec, CNRS-LIRMM, EMSE, Laboratoire Hubert Curien. Le projet a débuté en février 2016, pour une durée de 36 mois.

L'objectif de ce projet est d'étudier l'emploi en toute sécurité des téléphones mobiles de nouvelle génération munis de la technologie sans contact NFC. Ces téléphones mobiles représentent un marché potentiel de plusieurs milliards d'utilisateurs de services tels que le paiement sans contact, ou le paiement mobile. Par ailleurs, les données personnelles des utilisateurs sont des cibles potentielles pour les hackers. Les informations disponibles sur les téléphones portables pourraient servir pour des vols d'identité ou de données bancaires. Le projet TEEVA évaluera deux solutions alternatives de sécurisation des plateformes mobiles, l'une purement logicielle, la Whitebox Crypto, alors que l'autre (le TEE) intègre des éléments logiciels mais aussi matériels en s'appuyant sur la technologie Trustzone d'ARM disponible sur de nombreux chipsets du marché des smartphones et tablettes Android principalement.

Je suis impliquée dans la mise en œuvre d'attaques matérielles, logicielles et mixtes sur des environnements de tests constitués de plateformes de type ARM 9.

Chapitre II

Travaux de recherche

II.1 Thèmes de recherche

Mes travaux de recherche s'articulent entre les théories mathématiques pour la cryptographie, les aspects d'implémentation logicielle et matérielle, ainsi que les descriptions et réalisations d'attaques par canaux auxiliaires. À travers mes projets de recherche, j'étudie les variétés algébriques, la complexité des protocoles, l'arithmétique des corps finis, l'algorithmique et la construction de courbes elliptiques, la mise en œuvre de protocoles cryptographiques, les implémentations efficaces et sécurisées sur des cibles embarquées ainsi que les attaques par canaux cachés.

Les couplages forment le fil rouge autour duquel j'ai construit mon expertise. Les couplages sont un outil cryptographique très intéressant, dont la propriété de bilinéarité permet la simplification et la création de protocoles cryptographiques.

Les premiers couplages, de Weil et de Tate, ont en premier lieu été introduits en mathématiques en 1940 par les théoriciens des nombres, loin des préoccupations des cryptographes [187, 43].

Le premier intérêt des couplages du point de vue de la cryptographie fut leur emploi comme outils de cryptanalyse dans les années 1990. Les couplages ont permis de ramener le problème difficile du logarithme discret sur les courbes elliptiques au problème un peu moins difficile du logarithme discret sur le groupe multiplicatif d'un corps fini et ont ainsi été utilisés comme moyen d'attaque sur certaines courbes pour lesquelles le logarithme discret sur le corps fini est résoluble (attaques MOV [138] et Frey-Rück [84]).

Ce n'est qu'autour de l'an 2000 que les couplages ont été utilisés pour construire de nouveaux protocoles cryptographiques, solutions élégantes au défi de Shamir [172], comme la cryptographie basée sur l'identité [34], impossibles à réaliser auparavant, l'échange de clé tri-partite à la Diffie-Hellman [112], ou encore des schémas de signatures courtes [38].

La cryptographie basée sur l'identité permet une simplification de la gestion des clés des utilisateurs. Cette spécificité en fait un atout indéniable pour la cryptographie dans le *Cloud* et la gestion des *Big data*. Ainsi, l'engouement pour ce nouvel outil n'a cessé de croître et les algorithmes de calcul des couplages ont été améliorés au cours des dernières années.

Les couplages Ate et Twisted Ate ont été les prémices des notions de couplages optimaux et de réseaux de couplages et les implémentations actuelles des couplages sont assez efficaces pour envisager l'utilisation de ceux-ci dans des protocoles sur cartes à puce et téléphones.

D'un point de vue abstrait, un couplage est une application bilinéaire (ou bi-additive) non dégénérée notée $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, où $\mathbb{G}_1$, $\mathbb{G}_2$ et $\mathbb{G}_3$ sont trois groupes abéliens finis, de même ordre premier r.

La bilinéarité et la non-dégénérescence se traduisent comme suit (où les lois de groupes de $\mathbb{G}_1$ et de $\mathbb{G}_2$ sont notées additivement, alors que celle de $\mathbb{G}_3$ est noté multiplicativement) :

- *Bilinéarité :* $\forall P, P' \in \mathbb{G}_1, \forall Q, Q' \in \mathbb{G}_2$:

$$e(P + P', Q) = e(P, Q) \times e(P', Q),$$

$$e(P, Q + Q') = e(P, Q) \times e(P, Q').$$

- *Non-dégénérescence :*

$$\forall P \in \mathbb{G}_1 \setminus \{0\},\ \exists Q \in \mathbb{G}_2 \text{ tel que } e(P, Q) \neq 1,$$

$$\forall Q \in \mathbb{G}_2 \setminus \{0\},\ \exists P \in \mathbb{G}_1 \text{ tel que } e(P, Q) \neq 1.$$

La conséquence principale de la bilinéarité des couplages est l'égalité suivante :

$$\forall j \in \mathbb{Z},\ e([j]P, Q) = e(P, Q)^j = e(P, [j]Q)$$

où $[j]P := \underbrace{P + \cdots + P}_{j \text{ termes}}$ désigne la *multiplication scalaire* dans $\mathbb{G}_1$, et de même dans $\mathbb{G}_2$.

Dans les constructions existantes de couplages, les groupes $\mathbb{G}_1$ et $\mathbb{G}_2$ sont généralement des sous-groupes du groupe des points d'une courbe elliptique (ou hyperelliptique) et $\mathbb{G}_3$ est un sous-groupe des racines de l'unité d'un corps fini.

Illustrons l'utilisation des couplages en cryptographie avec une version simplifiée d'un protocole de génération de clé basé sur l'identité [34]. Alice et Bob souhaitent générer une clé commune de chiffrement symétrique en utilisant la cryptographie basée sur l'identité. Ils doivent se mettre en relation avec une autorité de confiance qui leur met à disposition les groupes $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ ainsi qu'un couplage $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$. Les identités d'Alice et Bob sont hashées respectivement en un élément Id_A de $\mathbb{G}_1$ et un élément Id_B de $\mathbb{G}_2$. (Pour les détails techniques, le livre [65] présente un état de l'art de la réalisation pratique d'un protocole à base de couplage.) L'autorité de confiance fournit à Alice et Bob leur clé privée respective, $P_A := [s]Id_A$ et $P_B := [s]Id_B$, en utilisant la clé maîtresse $s \in \mathbb{Z}$ du protocole (secrète, connue uniquement par l'autorité de confiance).

D'une part Alice peut évaluer $e(P_A, Id_B)$, et d'autre part Bob peut évaluer l'expression $e(Id_A, P_B)$. La bilinéarité des couplages nous permet d'affirmer que ces deux quantités sont égales, et peuvent donc constituer une clé privée de chiffrement commune, sans avoir besoin d'échanger une donnée sensible. En effet :

$$\begin{aligned} e(P_A, Id_B) &= e([s]Id_A, Id_B), \\ &= e(Id_A, Id_B)^s, \\ &= e(Id_A, [s]Id_B), \\ &= e(Id_A, P_B). \end{aligned} \tag{II.1.1}$$

Bien entendu, les protocoles basés sur l'identité reposent sur l'hypothèse que l'autorité de confiance porte bien son nom.

À travers l'étude des couplages, j'ai considéré plusieurs aspects importants de la cryptographie : les protocoles cryptographiques, les variétés algébriques, l'arithmétique, l'algorithmique, la complexité, l'implémentation efficace et la sécurité des algorithmes face aux attaques par canaux cachés. Ce sont les sujets au cœur des trois projets ANR auxquels j'ai participé, comme du projet FUI.

Dans la suite de ce chapitre je détaille mes travaux, en les classant dans mes différents axes de recherche, et en particulier de façon plus approfondie les articles [39, 54, 150, 56, 71] sur lesquels reposent ce manuscrit d'habilitation à diriger des recherches et qui se trouvent reproduits dans les annexes A, B, C, D et E.

II.2 Algorithmique des couplages

Dans cette partie sont présentés mes travaux relatifs à l'algorithmique des couplages :

- Mon premier travail [10]*[1] consistait à comparer la complexité algébrique des couplages de Weil et de Tate afin de déterminer lequel est le plus efficace à implémenter en fonction du niveau de sécurité ;
- L'algorithme de Miller est au cœur de l'implémentation des couplages et l'article [39]* propose une variante de celui-ci permettant d'étendre l'optimisation dite d'élimination des dénominateurs à toutes les tordues possibles pour les courbes elliptiques adaptées au couplage ;
- Les travaux [57, 54] sont respectivement un chapitre de livre et un article de journal dans lesquels est décrite de manière quasi-exhaustive la génération de paramètres pour un calcul efficace de couplage.

II.2.1 Complexité algébrique des couplages

Le calcul effectif des couplages est devenu intéressant pour les cryptographes dès lors que des applications concrètes ont été développées. Les groupes $\mathbb{G}_1$ et $\mathbb{G}_2$ impliqués

[1] La petite * indique mes travaux publiés dans des conférences internationales avec actes et comité de sélection.

dans le calcul du couplage de Weil ou de Tate sont des sous-groupes d'une courbe elliptique (ou hyperelliptique) définie sur un corps fini. Le résultat du couplage appartient aux racines de l'unité d'un corps fini. Des optimisations des couplages de Weil et de Tate ont été proposées, il s'agit des couplages Eta, Ate, twisted Ate, puis ont été introduites les notions de couplages optimaux ou de réseaux de couplages.

À mes débuts dans ce domaine, seuls les couplages de Weil et de Tate étaient connus et le second semblait le plus efficace des deux, en terme de complexité algébrique. En effet, le couplage de Tate se compose d'une exécution de la fonction (ou algorithme) de Miller et d'une exponentiation, alors que le couplage de Weil se compose de deux exécutions d'une fonction de Miller.

Voyons maintenant en détail comment sont réalisés les couplages de Weil et de Tate sur une courbe elliptique. Ils prennent en argument deux points d'une courbe elliptique E définie sur un corps fini $\mathbb{F}_p$ et renvoient un élément d'une extension $\mathbb{F}_{p^k}$ de ce corps fini, où p est un grand nombre premier et k le degré de plongement qui est un entier défini relativement à $E(\mathbb{F}_p)$.

Soient $E : y^2 = x^3 + ax + b$ une courbe elliptique définie sur $\mathbb{F}_p$, r un facteur premier du cardinal $\#E(\mathbb{F}_p)$ de $E(\mathbb{F}_p)$. Le degré de plongement k est le plus petit entier positif tel que r divise $(p^k - 1)$. Le point à l'infini de E est noté P_∞. Afin d'éviter que toute la r-torsion soit dans $E(\mathbb{F}_p)$, il est nécessaire de supposer que r^2 ne divise pas $\#E(\mathbb{F}_p)$.

Nous savons que les points de r-torsion de la courbe elliptique, dont l'ensemble est noté $E[r] = \{P \in E, rP = P_\infty\}$, sont inclus dans $E(\mathbb{F}_{p^k})$. Nous noterons $\mathbb{G}_1 = E(\mathbb{F}_p)[r]$ (où $E(\mathbb{F}_{p^d})[r]$ désigne l'ensemble des points de r-torsion de E à coordonnées dans une extension $\mathbb{F}_{p^d}$), $\mathbb{G}_2 \subset E(\mathbb{F}_{p^k})[r]$ et $\mathbb{G}_3 = \{\mu \in \mathbb{F}_{p^k}$ tel que $\mu^r = 1$ dans $\mathbb{F}_{p^k}\}$, autrement dit, $\mathbb{G}_3$ est le groupe des racines r-ième de l'unité.

Soient $P \in \mathbb{G}_1$ et $Q \in \mathbb{G}_2$ deux points de la courbe elliptique. À tout point R de $E(\mathbb{F}_{p^e})$, pour e un entier quelconque, et tout entier s, est associée une certaine fonction rationnelle $f_{s,R}$ appelée *fonction de Miller*.

Le couplage de Weil est alors défini comme l'application e_W :

$$\begin{aligned} e_W : \mathbb{G}_1 \times \mathbb{G}_2 &\to \mathbb{G}_3, \\ (P, Q) &\to (-1)^r \frac{f_{r,P}(Q)}{f_{r,Q}(P)}. \end{aligned}$$

Le couplage de Tate réduit est quant à lui défini comme l'application e_T telle que :

$$\begin{aligned} e_T : \mathbb{G}_1 \times \mathbb{G}_2 &\to \mathbb{G}_3, \\ (P, Q) &\to f_{r,P}(Q)^{\frac{p^k-1}{r}}. \end{aligned}$$

Ces deux couplages reposent donc sur le calcul d'une fonction rationnelle du type $f_{s,P}$, calcul effectué par l'algorithme de Miller [142]. L'exponentiation à la puissance $\frac{p^k-1}{r}$ intervenant dans le calcul du couplage de Tate est appelée *exponentiation finale*.

L'algorithme de Miller construit la fonction $f_{s,P}$ tout en l'évaluant au point Q. Il repose sur l'égalité de Miller [142] qui est la suivante :

$$f_{i+j,P} = f_{i,P} \times f_{j,P} \times \frac{l_{[i]P,[j]P}}{v_{[i+j]P}},$$

où i et j sont des entiers, $l_{[i]P,[j]P}$ désigne l'équation de la droite $([i]P[j]P)$ et $v_{[i+j]P}$ celle de la verticale au point $[i+j]P$.

L'égalité de Miller permet de calculer $f_{s,P}(Q)$ par un procédé itératif construit sur le schéma de l'exponentiation rapide à base de doublement et d'addition, et cette évaluation de la fonction $f_{s,P}$ au point Q s'effectue en $\log_2(s)$ étapes. Pour chaque étape, il suffit d'évaluer deux équations de droites au point Q. La méthode de calcul est résumée dans l'algorithme 1.

Algorithme 1 : Miller(P, Q, s)

Données : $s = (s_n \dots s_0)$(représentation binaire), $P \in \mathbb{G}_1$ et $Q \in \mathbb{G}_2$;
Résultat : $f_{s,P}(Q) \in \mathbb{G}_3(\subset \mathbb{F}^{\star}_{p^k})$;

$T \leftarrow P$;
$f_1 \leftarrow 1$;
pour $i = n-1$ **to** 0 **faire**
  $T \leftarrow [2]T$;
  $f_1 \longleftarrow {f_1}^2 \times \frac{l_1(Q)}{v_1(Q)}$;
  où l_1 est l'équation de la tangente au point T ;
  et v_1 est l'équation de la verticale au point $[2]T$;
  si $s_i = 1$ **alors**
    $T \leftarrow T + P$;
    $f_1 \longleftarrow f_1 \times \frac{l_2(Q)}{v_2(Q)}$;
    où l_2 est l'équation de la droite (TP) ;
    et v_2 est l'équation de la verticale au point $T + P$;
  retourner f_1

La complexité de l'algorithme de Miller dépend directement de la complexité des fonctions l_1, l_2, v_1 et v_2 et celles-ci dépendent du choix de la représentation de la courbe elliptique, du système de coordonnées et de l'arithmétique des corps finis. Une optimisation classique de l'algorithme de Miller est l'élimination des dénominateurs. Cette optimisation repose sur la représentation des points de $\mathbb{G}_2$ en tant qu'image de la tordue de degré pair de la courbe elliptique[2].

Le couplage de Weil requiert deux exécutions de l'algorithme de Miller plus une inversion et une multiplication dans $\mathbb{F}_{p^k}$; tandis que le couplage de Tate ne nécessite qu'une seule exécution de l'Algorithme 1 suivie d'une exponentiation.

Appelons Miller Lite l'évaluation $f_{r,P}(Q)$, et Miller Full le calcul de $f_{r,Q}(P)$. Cette distinction vient du fait que durant le calcul de $f_{r,P}(Q)$, l'arithmétique de la courbe elliptique est effectuée dans $\mathbb{F}_p$; puis on évalue les équations de droites à coefficients dans $\mathbb{F}_p$ en des éléments de $\mathbb{F}_{p^k}$. Inversement, pour le calcul de Miller Full les doublements et additions de points sur la courbe utilisent des opérations dans $\mathbb{F}_{p^k}$, et nous évaluons des équations de droites à coefficients dans $\mathbb{F}_{p^k}$ en des éléments de $\mathbb{F}_p$. La complexité algébrique de l'opération Miller Full est plus coûteuse que celle de l'opération Miller Lite : l'arithmétique dans $\mathbb{F}_{p^k}$ est plus lourde que celle de $\mathbb{F}_p$.

[2]Soient E et $\widetilde{E}$ deux courbes elliptiques définies sur $\mathbb{F}_q$. Alors, la courbe $\widetilde{E}$ est **un twist de degré d de la courbe** E s'il existe un isomorphisme Ψ_d défini sur $\mathbb{F}_{q^d}$ de $\widetilde{E}$ dans E et tel que d soit minimal.

Mon premier travail, en collaboration avec Jean-Claude Bajard [10]⋆, consista à répondre à la question : du couplage de Weil et du couplage de Tate, lequel est le plus coûteux à calculer ? *A priori*, à un niveau de sécurité donné, l'exponentiation finale est moins coûteuse qu'un calcul de Miller full. Néanmoins, pour des niveaux de sécurité élevés, cette exponentiation finale est effectuée dans un corps fini dont le nombre d'éléments est de plus en plus grand et le coût de l'arithmétique associée augmentant avec la taille, le couplage de Weil devient asymptotiquement plus efficace que le couplage de Tate [10]⋆. Ce premier travail a permis de montrer l'influence de l'arithmétique des corps finis sur la complexité algébrique et donc sur le calcul des couplages.

II.2.2 Une variante de l'algorithme de Miller

Dans le cadre de mon post-doctorat à l'université de Caen, j'ai travaillé en collaboration avec John Boxall, Fabien Laguillaumie et Phong Le Duc sur l'arithmétique de l'algorithme de Miller.

Dans l'article [39]⋆, nous avons proposé une version de cet algorithme en rupture avec ce qui se faisait habituellement puisque reposant sur une réécriture de l'égalité de Miller. Cet article a été publié dans la conférence internationale avec actes et comité de sélection Pairing 2010[3].

Nous avons montré, à une constante multiplicative près, l'égalité suivante :

$$f_{a+b} = \frac{1}{f_{-a} f_{-b} l_{-a,-b}},$$

laquelle permet de procéder à l'élimination des dénominateurs quel que soit le degré de la tordue de la courbe elliptique, alors que cette optimisation classique n'était alors employée que pour les courbes elliptiques admettant une courbe tordue de degré pair.

Je n'entre pas ici dans les détails techniques mais l'intérêt de cette nouvelle formule tient au fait qu'elle peut rendre envisageable l'utilisation de courbes elliptiques dont le degré de plongement n'est pas forcément pair. Notre article [39]⋆ est reproduit dans l'annexe A.

Ainsi, il pourrait être intéressant de chercher de nouvelles familles de courbes elliptiques et d'étudier la complexité algébrique de l'implémentation du couplage avec cette nouvelle formule. Cette interrogation s'est concrétisée sous la forme d'une thèse en co-tutelle entre l'université Paris 8 et l'Université de Yaoundé : je co-encadre Aminatou Pecha qui étudie la construction de courbes elliptiques différentes de celles recommandées pour les couplages afin de comparer ce qu'elles pourraient apporter en terme d'efficacité et de sécurité ainsi que la généralisation du couplage β-Weil. Le détail de ces travaux se trouve dans la section IV.3.

II.2.3 Sélection des paramètres

L'implémentation efficace d'un couplage repose sur de nombreux paramètres. Nous considérons dans ce paragraphe les couplages définis sur les courbes elliptiques en re-

[3]En 2010, l'engouement pour les couplages était tel qu'une conférence leur était dédiée.

présentation de Weierstrass. Les couplages peuvent être définis sur d'autres modèles de courbes (hyperelliptique ou Kummer par exemple) ou d'autres représentations de courbe (Edwards ou Jacobi par exemple). En ce qui concerne les courbes elliptiques, les records d'implémentation ont été atteints en utilisant la représentation de Weierstrass.

Les autres paramètres comme le choix du corps fini $\mathbb{F}_p$, le niveau de sécurité (les tailles en nombre de bits de r et p), le degré de plongement k, un générateur des groupes $\mathbb{G}_1$ et $\mathbb{G}_2$, ou encore le choix du couplage à implémenter, sont communs quel que soit le modèle de courbe choisi.

Les paramètres pour implémenter un couplage se déterminent en général dans l'ordre suivant :

- En premier lieu, il faut fixer le niveau de sécurité souhaité ;
- Ensuite, on détermine une valeur pour le degré de plongement k de la courbe elliptique ;
- Puis on construit une famille de courbes elliptiques admettant ce degré k de plongement en utilisant l'article [82]. Lors de cette étape, les choix de p et r sont arrêtés ;
- Avec tous ces éléments, il est possible de construire les sous-groupes $\mathbb{G}_1$ et $\mathbb{G}_2$ d'ordre r tels que $\mathbb{G}_1 \times \mathbb{G}_2$ soit l'ensemble de départ du couplage ;
- La dernière étape est le choix du couplage, essentiellement le couplage Tate, Ate ou twisted Ate ou une version optimale de ces derniers.

Pour chaque étape, de nombreuses options sont possibles. La complexité finale du calcul du couplage dépend directement des choix de tous ces paramètres. Jusqu'à l'été 2016, les niveaux de sécurité pour les couplages étaient déterminés et la table II.1 rappelle les recommandations qui étaient faites.

Niveau de sécurité en bits	Dimension de r en bits	Dimension de p^k en bits	Degré de plongement k	
			$\rho \approx 1$	$\rho \approx 2$
80	160	960 – 1280	6 – 8	3 – 4
112	224	2200 – 3600	10 – 16	5 – 8
128	256	3000 – 5000	12 – 20	6 – 10
192	384	8000 – 10000	20 – 26	10 – 13
256	512	14000 – 18000	28 – 36	14 – 18

TABLE II.1 – Évaluation des paramètres de sécurité des courbes elliptiques pour couplage.

Les courbes elliptiques en caractéristique 2 et 3 n'assurent plus un niveau de sécurité suffisants suite aux travaux [114, 13]. Les courbes elliptiques sur les corps finis de grande caractéristique sont vulnérables à l'attaque présentée dans l'article [118].

Cela induit donc une modification de la table II.1. Pour l'instant deux articles proposent une mise à jour asymptotique des niveaux de sécurité [139, 12], et tous deux proposent des dimensions approximatives pour des familles de courbes données [82]. En effet, l'attaque de Kim et Barbulescu [118] repose sur des algorithmes de calcul d'index et sa complexité est difficile à établir précisément. Les deux articles s'attachent à préciser au mieux cette complexité, mais leurs conclusions divergent comme le montre la table II.2. Cette table transcrit les recommendations des niveaux de sécurité pour des familles de courbes classiques. Les familles de courbes sont nommées d'après les initiales des chercheurs les ayant proposées. Il s'agit des courbes Barreto-Naehrig (BN), Barreto-Lynn-Scott de degré de plongement 12 (BLS12) ou 24 (BLS24) et des courbes Kachisa-Schaefer-Scott de degré de plongement 18 (KSS18) [82].

Niveau de sécurité	BN	BLS12	KSS18	BLS24
128 bits [139]	383	384	342	320
128 bits [12]	461	460	238	–
192 bits [139]	1031	1147	597	480
192 bits [12]	–	–	677	554

TABLE II.2 – Ajustement pour les paramètres de sécurité pour couplage.

La table précise les dimensions de p minimales pour assurer un niveau de sécurité de 128 et de 192 bits. Par exemple, au niveau de sécurité 128 bits, pour les deux articles la famille de courbe KSS18 semble nécessiter une dimension de p réduite : Menezes et al. [139] requiert $\log_2(p) = 342$ alors que Barbulescu et Duquesne [12] retient $\log_2(p) = 238$. Cette différence de dimension de 100 bits sur $\mathbb{F}_p$ se traduit par une différence de 1 800 bits sur $\mathbb{F}_{p^{12}}$. Des travaux supplémentaires paraissent donc nécessaires pour assurer le bon niveau de sécurité tout en permettant les implémentations les plus efficaces.

Mettons de côté la problématique du niveau de sécurité. Le choix des paramètres suivants n'en demeure pas moins primordial. Une des missions de la tâche 2 du projet ANR SIMPATIC concernait justement un transfert de savoir concernant les choix algorithmiques vers les ingénieurs implémentant les couplages. (Cette mission a donné lieu à une publication dans le journal AAECC à paraître [54] et à un chapitre de livre [57].) Ce travail fournit un tutoriel pour la génération de paramètres de couplages permettant une implémentation efficace en fonction de la cible d'implémentation.

Nous n'avons détaillé avec précision le processus que pour la famille de courbes BN, mais la méthode décrite est suffisamment générique pour permettre une application simple à toute autre famille de courbes. Le plan du tutoriel est le suivant : après un rappel du calcul du couplage optimal Ate sur les courbes BN, nous présentons les différentes options pour construire l'extension de corps $\mathbb{F}_{p^{12}}$ et l'implémentation des opérations de base. En particulier, nous décrivons comment choisir le polynôme pour construire l'extension, la tour choisie ainsi que les algorithmes de multiplications en fonction de la complexité des opérations sur $\mathbb{F}_p$. Nous expliquons comment choisir

les paramètres de la famille de courbe, puis comment construire les sous-groupes et représentations inhérents à la courbe elliptique. L'article [54] est reproduit en annexe B.

II.3 Arithmétique des corps finis

L'implémentation efficace des couplages repose pour beaucoup sur une arithmétique efficace des corps finis et des extensions (de degré fini) de ceux-ci. Plusieurs de mes travaux publiés portent sur cette thématique [74, 75, 72, 24, 149, 151, 152]*, dont un article de journal [150]. Le chapitre [24] présente l'état de l'art de l'arithmétique des corps finis pour application au couplage.

Mes travaux dans ce domaine concernent en particulier :

- La complexité bilinéaire de la multiplication sur les corps finis et leurs extensions, pour des degrés d'extensions particuliers [74]* et pour des degrés d'extensions quelconques [72]* ;
- L'utilisation de l'arithmétique en représentation AMNS (Adapted Modular Number System) pour améliorer l'efficacité de la multiplication sur les corps finis en large caractéristique [75] et sur leurs extensions [72]* ;
- Les implémentations sur des cartes reprogrammables (FPGA), avec mon doctorant Amine Mrabet, de la multiplication sur $\mathbb{F}_p$ [152]*, et [150], de l'inversion et de la division pour la cryptographie sur courbe elliptique [149]*, ainsi que l'arithmétique des courbes elliptiques [151]*.

II.3.1 Les extensions de degré 5, 6 et 7

En collaboration avec Sorina Ionica et Aurore Guillevic, nous avons proposé une multiplication efficace pour des extensions de degré 5, 6 et 7 [74]*.

Il existe différents cas d'utilisation d'extensions de degré 5 en cryptographie. Celles-ci figurent par exemple dans les travaux de Rubin et Silverberg [167] qui étudient le problème de compression de la représentation des éléments d'un corps fini en utilisant des tores algébriques $T_n(\mathbb{F}_q)$. Van Dijk et al., dans [182], développent les protocoles de type El Gamal pour le chiffrement, la signature et le vote électronique sur le tore $T_{30}(\mathbb{F}_q)$. Ils proposent entre autre une implémentation de la multiplication sur l'extension $\mathbb{F}_{q^5}$ de complexité $15M_q$+$75A_q$, où M_q et A_q représentent le coût d'une multiplication et d'une addition dans $\mathbb{F}_q$.

Les extensions de degré 5 sont également utilisés pour les couplages : pour les hauts niveaux de sécurité (192 et 256 bits) les familles de courbes de degré de plongement 30 ou 35 étaient recommandées [81, 82]. Or les tours d'extensions pour construire $\mathbb{F}_{p^{30}}$ et $\mathbb{F}_{p^{35}}$ impliquent de passer par une extension de type $\mathbb{F}_{q^5}$, q pouvant être dans le cas des couplages p ou une puissance de p.

Par ailleurs, et pour revenir aux travaux cités plus haut, il a été observé que les valeurs des couplages peuvent être représentées sous forme compressée en utilisant les tores algébriques. Cela a été fait pour les courbes supersingulières en caractéristique 3 [90] et pour les courbes BN [153].

La multiplication pour des extensions de degré 5 a déjà été étudiée. Bodrato [32] propose une méthode pour améliorer l'algorithme de Toom Cook en caractéristique 2. Dans [8], la multiplication dans $\mathbb{F}_{q^5}$ est obtenue par deux applications de la méthode de Karatsuba de sorte que la complexité soit de $14M_q + 34A_q$. Jusqu'à notre article, les meilleurs résultats ont été obtenus par Montgomery [146] : $13M_q + 62A_q$.

La complexité de notre multiplication atteint la borne inférieure (conjecturée) du nombre d'opérations élémentaires dans le corps fini de base en utilisant judicieusement les valeurs d'interpolation de la méthode de Newton. De plus, l'interpolation de Newton nécessite des divisions exactes par les différences des valeurs d'interpolation. De manière générale, la division dans un corps fini est une opération coûteuse. Il est communément admis que suivant l'algorithme utilisé, une division représente entre 10 et 50 multiplications dans le corps fini. Nous proposons un algorithme pour effectuer ces inversions avec une complexité équivalente à celle d'une soustraction. Cette astuce permet de réduire considérablement le coût de notre multiplication et d'être plus efficace que tous les autres propositions pour les extensions de degré 5, 6 et 7, en particulier celles de Montgomery [146] dès que $M_q > 18A_q$. Pour une multiplication dans $\mathbb{F}_{q^5}$ nous obtenons une complexité de $9M_q + 137A_q$.

II.3.2 La représentation AMNS

On sait que la multiplication est une opération très importante en cryptographie. Pour l'optimiser il peut être judicieux d'étudier la représentation des éléments d'un corps fini. En effet, changer de représentation pour les corps finis permet de développer de nouvelles méthodes pour implémenter la multiplication, avec parfois des gains non négligeables en termes de complexité.

Je me suis ainsi intéressée à une représentation originale des corps finis : la représentation en base adaptée (Adapted Modular Number System ou AMNS)[162]. Cette représentation d'un corps fini permet d'améliorer la multiplication dans les extensions de corps finis.

Le principe en est le suivant : un élément d'un corps fini est représenté comme un polynôme en une variable γ (fixée *a priori* dans le corps), dont les coefficients sont petits, dans le sens où il est imposé aux coefficients des polynômes d'être de taille inférieure à la taille d'un mot machine ; bien entendu la taille du mot machine dépendra du matériel mais nous pouvons considérer aujourd'hui que cette taille est de 32 ou de 64 bits.

L'intérêt de cette représentation est qu'elle manipule, lors d'une opération arithmétique, des mots de taille machine et des multiplications par des puissances de γ, et ces dernières sont, dans la base AMNS, composées de décalages et parfois d'additions.

J'ai travaillé avec Christophe Nègre pour construire un algorithme de multiplication efficace sur une extension de corps $\mathbb{F}_{p^k}$ [75]* utilisant la représentation AMNS. La multiplication dans l'extension est faite, suivant le degré de l'extension, soit avec une transformée de Fourier discrète soit avec une transformée de Fourier rapide. Dans la continuité de ces travaux, nous avons proposé, avec Nicolas Gama, une implémentation efficace de cette multiplication [72]* et une méthode pour construire plus efficacement les paramètres d'une représentation AMNS. En effet, pour obtenir

ces paramètres dans [75]⋆ il est nécessaire de réduire un réseau arithmétique, ce qui se fait plus efficacement [72]⋆ en s'inspirant des méthodes issues de la description du protocole homomorphique de Gentry. L'implémentation est en langage C, et nos résultats, représentés dans le graphique II.1, sont très prometteurs, puisque la multiplication en base AMNS est plus efficace que la multiplication dans les extensions de corps programmée en utilisant les bibliothèques optimisées NTL et GMP.

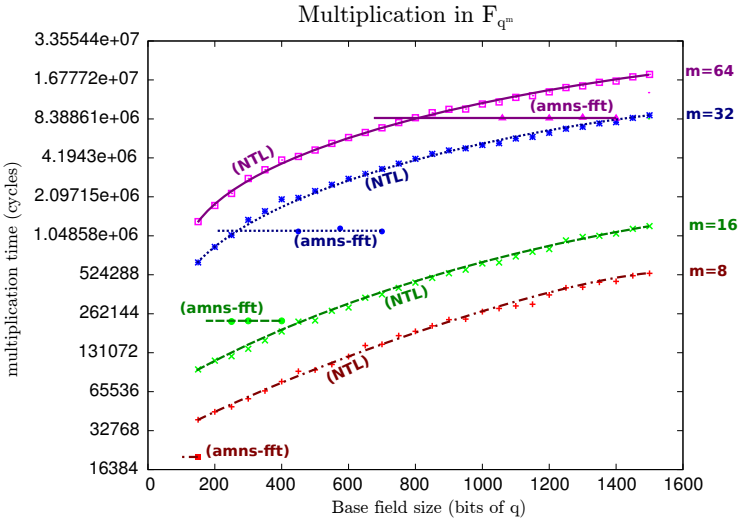

FIGURE II.1 – Résultat de notre implémentation AMNS.

II.3.3 Implémentation matérielle

J'ai co-encadré Amine Mrabet en thèse (thèse en co-tutelle avec l'université Paris 8 et l'université de Monastir en Tunisie, débutée en septembre 2014). Le sujet concerne l'étude de l'implémentation de primitives cryptographiques sur un circuit reprogrammable (FPGA). Amine a étudié l'arithmétique des corps finis afin d'améliorer les primitives pour la cryptographie à base de courbes elliptiques et en particulier pour les couplages.

Nous avons étudié la multiplication modulaire de Montgomery, opération indispensable pour l'arithmétique des corps finis. Nous avons choisi d'implémenter l'algorithme 2 dit CIOS sur une architecture FPGA dédiée. Les paramètres en entrée de l'algorithme 2 sont un nombre premier p ; w, s deux entiers positifs représentant respectivement les bits de poids faibles, et les bits de poids forts ; $p' = -p^{-1} \mod 2^w$; enfin, a et b sont les nombres modulo p dont nous cherchons le produit.

L'architecture que nous proposons, combine les circuits systoliques, se décomposant en plusieurs petites cellules élémentaires, et l'algorithme CIOS pour la multiplication de Montgomery. Les résultats obtenus sont encourageants puisque nous avons proposé une architecture peu coûteuse en surface, rapide en temps d'exécution et qui est comparativement la plus efficace connue actuellement [152]⋆. La version étendue de ce travail [150] vient d'être acceptée (août 2017) pour publication dans le journal Hardware and Systems Security, et est reproduite dans l'annexe C. La table II.3 présente la comparaison de l'implémentation de la multiplication modulaire de Montgomery avec l'état de l'art. Cette table compare les résultats obtenus

Algorithme 2 : CIOS algorithm for Montgomery multiplication [123]

Input : $p < 2^K$, $p' = -p^{-1} \mod 2^w$, w, s , $K = s \cdot w$:bit length, $R = 2^K$, $a, b < p$
Output : $a \cdot b \cdot R^{-1} \mod p$

```
1  T ← Null ;
2  for i ← 0 to s − 1 do
3      C ← 0 ;
4      for j ← 0 to s − 1 do
5          (C, S) ← T[j] + a[i] · b[j] + C
6          T[j] ← S
7      (C, S) ← T[s] + C
8      T[s] ← S
9      T[s + 1] ← C
10     C ← 0 ;
11     m ← T[0] · p′ mod 2^w
12     (C, S) ← T[0] + m · p[0]
13     for j ← 1 to s − 1 do
14         (C, S) ← T[j] + m · p[j] + C
15         T[j] ← S
16     (C, S) ← T[s] + C
17     T[s − 1] ← S
18     T[s] ← T[s + 1] + C
19 return T ;
```

dans notre travail [150] avec les résultats de la littérature les plus récents. Les lignes correspondent à la fréquence des cartes (Freq.), le nombre de cycles d'horloge nécessaires pour calculer la multiplication de Montgomery sur la cible d'implémentation (Cycles), la vitesse d'exécution de la multiplication en micro-secondes (Speed μs), le nombre de registre nécessaires pour l'implémentation (Slice Reg), le nombre de table logique (Slice LUT) et le nombre de registres mémoire de type BRAM (BRAM).

	Our works A7		**Our works V5**		[161] V5		[26] K7 and V5	
	512	**1024**	**512**	**1024**	512	1024	512 K7	512 V5
Freq.	**106**	**65**	**97**	**65**	95	130	176	123
Cycles	**66**	**66**	**66**	**66**	96	384	66	66
Speed μs	**0.622**	**1.013**	**0.680**	**1.015**	1.010	2.953	0.374	0.536
Slice Reg	**2164**	**4208**	**3046**	**6072**	3876	6642	5076	4960
Slice LUTs	**1789**	**5242**	**1781**	**5824**	–	–	8757	10877
BRAM	0	0	0	0	128	256	0	0

TABLE II.3 – Comparaison avec l'état de l'art pour la multiplication modulaire de Montgomery.

Nous avons également travaillé sur un algorithme d'inversion, modification de l'algorithme d'Euclide étendu, pouvant aisément évoluer en un algorithme de division [149]*. L'algorithme d'Euclide étendu a été modifié afin d'être exécuté en un maximum de $4N$ itérations, où N représente la longueur de la décomposition binaire des opérandes. La modification consiste à réécrire l'algorithme en mettant en évidence des blocs logiques dans le but d'optimiser le nombre d'itérations mais surtout la surface nécessaire pour la programmation de l'inversion. Les résultats sont meilleurs que les plus récents dans ce domaine comme le montre la table II.4.

L'arithmétique des corps finis que nous avons implémentée a été utilisée pour améliorer les performances de l'arithmétique sur courbe elliptique, ainsi que les tours d'extensions nécessaires pour le calcul de couplage [151]*.

II.4 Géométrie algébrique

La construction des couplages repose sur des fondements de géométrie algébrique. Je décris dans cette partie mes travaux se rapportant à ce domaine :

- J'ai présenté ci-dessus les couplages définis sur des courbes elliptiques. Dans le travail écrit en collaboration avec Sylvain Duquesne et Emmanuel Fouatsa [56],

	Slice FF	Slice LUT	slice	Freq. (MHz)	Cycles d'horloge
[25] 384 bits (Virtex5)	8113	30619	9119	127	3518
[149]* **384 bits (Virtex5)**	1573	2286	873	129	1536
[149]* **384 bits (Artix7)**	1573	3029	850	93	1536

TABLE II.4 – Implementations de l'nversion sur Xilinx FPGA.

nous avons étudié le calcul d'un couplage sur les courbes de Jacobi quartiques ;

- En collaboration avec Le-Duc Phong, nous avons proposé une méthode de construction de familles complètes de courbes de type MNT [128]* ;
- Il existe dans la littérature une construction des couplages sur les fonctions théta. J'ai étudié la sécurité de ces couplages lorsqu'ils sont soumis à des attaques par canaux cachés dans l'article [69]*.

II.4.1 Courbes de Jacobi

Les implémentations les plus efficaces des couplages ont été réalisées sur des courbes elliptiques, et plus particulièrement sur des courbes elliptiques décrites par leur équation de Weierstrass. Je l'ai déjà abordé : il existe d'autres modèle de courbes elliptiques, par exemple les courbes d'Edwards, de Huff ou celles de Jacobi. Chaque modèle de courbes possède sa propre version de l'algorithme de Miller.

Avec Sylvain Duquesne et Emmanuel Fouatsa nous nous sommes intéressés au calcul d'un couplage sur les courbes de Jacobi quartiques [56]. Ces courbes elliptiques admettent une tordue de degré 4 ce qui nous permet d'optimiser le calcul des couplages. Une tordue de degré 4 d'une telle courbe d'équation $Y^2 = dX^4 + Z^4$ définie sur l'extension $\mathbb{F}_{q^{k/4}}$ est donnée par

$$E_d^\omega : Y^2 = d\omega^4 X^4 + Z^4$$

où $\omega \in \mathbb{F}_{q^k}$ est tel que $\omega^2 \in \mathbb{F}_{q^{k/2}}$, $\omega^3 \in \mathbb{F}_{q^k} \backslash \mathbb{F}_{q^{k/2}}$ et $\omega^4 \in \mathbb{F}_{q^{k/4}}$. Autrement dit, $\{1, \omega, \omega^2, \omega^3\}$ est une base de $\mathbb{F}_{q^k}$ vu comme un $\mathbb{F}_{q^{k/4}}$-espace vectoriel.

L'équivalence birationnelle entre la courbe de Jacobi en coordonnées projective $E_d : Y^2 = dX^4 + Z^4$ et la courbe de Weierstrass $W_d : y^2 = x^3 - 4dx$ s'écrit :

$$\begin{array}{rccl} \varphi : & E_d & \longrightarrow & W_d \\ & (0:1:1) & \longmapsto & P_\infty \\ & (0:-1:1) & \longmapsto & (0,0) \\ & (X:Y:Z) & \longmapsto & \left(2\frac{Y+Z^2}{X^2}, 4\frac{Z(Y+Z^2)}{X^3}\right) \end{array} \qquad \begin{array}{rccl} \varphi^{-1} : & W_d & \longrightarrow & E_d \\ & P_\infty & \longmapsto & (0:1:1) \\ & (0,0) & \longmapsto & (0:-1:1) \\ & (x,y) & \longmapsto & (2x:2x^3-y^2:y). \end{array}$$

Tout ceci nous permet de transformer les étapes de doublement et d'addition de l'algorithme de Miller afin qu'elles soient plus performantes.

La fonction de Miller obtenue en utilisant l'équivalence birationnelle sur la courbe de Jacobi quartique $E_d : Y^2 = dX^4 + Z^4$ est donnée par l'égalité $l_{R,S}(X,Y,Z) = l_{P_1,P_2}(\varphi(X,Y,Z))$. Nous avons alors :

$$l_{R,S}(X,Y,Z) = \frac{4X_3^2X^2}{2X_3^2(Y+Z^2)-2X^2(Y_3+Z_3^2)}\left(\frac{ZY+Z^3}{X^3}-\frac{1}{2}\lambda\left(\frac{Y+Z^2}{X^2}\right)-\frac{\alpha}{4}\right).$$

avec

$$\lambda = \begin{cases} \dfrac{-2X_1^3Z_2(Y_2+Z_2^2)+2X_2^3Z_1(Y_1+Z_1^2)}{X_1X_2[-X_1^2(Y_2+Z_2^2)+X_2^2(Y_1+Z_1^2)]} & \text{si } P_1 \neq P_2, \\ \dfrac{Y_1+2Z_1^2}{X_1Z_1} & \text{si } P_1 = P_2, \end{cases} \tag{II.4.1}$$

et

$$\alpha = \begin{cases} \dfrac{-4(Y_1+Z_1^2)(Y_2+Z_2^2)(Z_2X_1-Z_1X_2)}{X_1X_2[-X_1^2(Y_2+Z_2^2)+X_2^2(Y_1+Z_1^2)]} & \text{si } P_1 \neq P_2, \\ \dfrac{-2Y_1(Y_1+Z_1^2)}{X_1^3Z_1} & \text{si } P_1 = P_2. \end{cases} \tag{II.4.2}$$

Nous utilisons la représentation du point Q de $\mathbb{G}_2$ comme image de la tordue : $(\omega X_Q : Y_Q : Z_Q)$ ou $(x_Q\omega, y_Q, 1)$ en coordonnées affine, avec X_Q, Y_Q, Z_Q, x_Q et y_Q dans $\mathbb{F}_{q^{k/4}}$.

Après simplification des équations nous obtenons :

$$\tilde{l}_{R,S}(x_Q\omega, y_Q, 1) = B\left(\frac{y_Q+1}{x_Q^2\omega^4}\right)\omega^2 + D\left(\frac{y_Q+1}{x_Q^3\omega^4}\right)\omega + A.$$

Comme le point $Q = (x_Q\omega, y_Q, 1)$ est fixé pendant les calculs, les fractions $\frac{y_Q+1}{x_Q^3\omega^4}$ et $\frac{y_Q+1}{x_Q^2\omega^4}$ peuvent être précalculées dans $\mathbb{F}_{q^{k/4}}$.

Nos complexités algébriques sont les plus performantes pour le calcul de couplage de Tate et de ses variantes sur les courbes admettant une tordue de degré 4. Nous avons amélioré la complexité algébrique théorique des étapes de doublement et d'addition de l'algorithme de Miller d'au plus 27% et 39% par rapport à [46] et à [185].

Ce travail a été publié dans le Journal of Mathematical Cryptography [56] et se trouve dans l'annexe D.

II.4.2 Courbes MNT

J'ai travaillé sur la construction de courbes adaptées au calcul de couplages. Il s'agit de la famille de courbes dites MNT, du nom de leurs inventeurs Miyaji, Nakabayashi et Takano [144]. Avec Le Duc Phong de l'Université de Singapour, nous avons proposé une méthode permettant de générer des familles complètes de courbes MNT [128]*.

La famille de courbes MNT est particulièrement adaptée aux couplages d'ordre premier et de degré de plongement = 3, 4 et 6 [144]. *A priori*, les couplages sur ces courbes n'assuraient plus un niveau de sécurité suffisant par rapport aux couplages définis sur des corps de grande caractéristique. Néanmoins, les progrès récents sur

la résolution du logarithme discret pour les courbes elliptiques à couplage en grande caractéristique pourrait relancer l'intérêt cryptographique pour ces courbes.

Scott et Barreto [169] et Galbreith et al. [86] ont étendu la notion de courbe MNT en autorisant le cardinal de la courbe à avoir un petit co-facteur $h \geq 2$ au lieu d'être premier. La méthode de Galbreith et al. permet de construire des familles explicites de courbes, alors que la méthode de Scott et Barreto n'en génère que des cas particuliers.

Nous avons généralisé la méthode de Scott et Barreto afin de construire des familles de courbes et notre méthode, alternative à celle de Galbreith et al., permet de générer un plus grand nombre de familles de courbes avec n'importe quel co-facteur.

Une version étendue de notre travail [128]*, actuellement soumis, présente une heuristique sur le nombre de courbes MNT que nous pouvons trouver.

II.4.3 Fonction théta

De manière générale, les couplages sont considérés sur des courbes algébriques de genre 1 ou 2 mais de récents travaux s'intéressent au calcul de couplages pour n'importe quelle variété abélienne, et, en particulier, David Lubicz et Damien Robert ont proposé des algorithmes des couplages de Weil et de Tate sur les fonctions theta[4] [135].

Dans l'article publié dans les actes de la conférence CAI 2013 [69]*, je m'intéresse à la sécurité de ces couplages lorsqu'ils sont soumis à des attaques par canaux cachés. J'ai ainsi explicité les formules intervenant dans le calcul des couplages, et en ai déduit quelles pouvaient être les failles de sécurité. J'ai proposé une contre-mesure reposant sur la bilinéarité des couplages, puisque contrairement aux couplages sur courbes elliptiques, l'homogénéité des coordonnées projectives n'est pas une contre-mesure suffisante pour les fonctions theta.

II.5 Attaques par canaux cachés

Au travers de mes différents travaux, j'ai compris certaines subtilités de l'arithmétique et de l'implémentation des couplages. J'ai ainsi pu m'intéresser à des sujets encore peu explorés comme par exemple la résistance des algorithmes de couplages aux attaques par canaux cachés. Ces travaux m'ont permis de développer des compétences dans le domaine de la microélectronique.

Il est possible d'attaquer un protocole cryptographique, afin de retrouver le secret utilisé, non pas en exploitant directement une faiblesse de l'algorithme mais plutôt en observant les éventuelles fuites d'information causées par la mise en œuvre du protocole. Les attaques par canaux cachés sont ainsi une solution alternative pour toute personne souhaitant intercepter et comprendre les secrets échangés par d'autres. Elles ont la particularité de déduire des informations sur le secret utilisé en

[4]Les fonctions theta sont des fonctions analytiques (complexes) définies sur le produit d'un espace vectoriel complexe g-dimensionnel par l'espace des matrices complexes carrées de taille g, symétriques, dont la partie imaginaire est définie positive, pour un entier g fixé, et vérifiant une propriété de quasi-périodicité, i.e., périodicité à un facteur multiplicatif près.

écoutant les bruits provoqués par le fonctionnement de l'algorithme. Il peut s'agir de la consommation électrique, du rayonnement électromagnétique ou du temps de calcul. Il est aussi possible de provoquer intentionnellement des erreurs durant l'exécution d'un algorithme pour en déduire des informations secrètes.

La cryptographie basée sur les couplages, en particulier la cryptographie basée sur l'identité n'échappe pas à la règle. Je m'intéresse à deux types d'attaques : mes travaux [63, 67, 69, 126, 70, 188]*, l'article de journal [71] et les chapitres [77, 73] concernent les attaques par fautes ; mes travaux [64, 95, 110, 111]* concernent les attaques par analyse de consommation.

Je me suis essentiellement intéressée aux attaques par canaux auxiliaires contre l'algorithme de Miller, excepté dans [95]* où la cible est une boîte-S. Dans le chapitre de livre [73] nous avons établi un état de l'art des attaques par canaux auxiliaires contre les algorithmes de couplage existant en 2016.

Je décris ci-après les attaques existantes contre les algorithmes de couplage, puis présente mes travaux relatifs aux attaques d'implémentations non protégées. Je récapitule les contre-mesures pour les couplages connues dans la littérature et termine cette partie par la description des attaques menées contre des implémentations protégées de couplage.

II.5.1 Description des attaques

Lorsqu'un protocole cryptographique est implémenté sur un matériel électronique, ce dernier peut être la cible d'attaque par canaux auxiliaires. Le terme de cible désigne donc l'implémentation matérielle d'un procédé cryptographique.

Dans le cas des couplages, les protocoles pouvant être sensibles à une attaque par canaux cachés sont ceux impliquant un secret, par exemple la cryptographie basée sur l'identité. Pour les protocoles basés sur l'identité, un calcul de couplage impliquant une donnée publique et une donnée secrète (ou privée) est fait systématiquement. C'est illustré par l'exemple de création de clé entre Alice et Bob déjà rencontré : Alice doit calculer $e(P_A, Id_B)$, où la donnée Id_B est publique et P_A est la clé privée d'Alice. Lors du calcul $e(P_A, Id_B)$ un attaquant peut essayer de récupérer de l'information sur P_A.

Contrairement à ce qui se passe pour les courbes elliptiques ou pour RSA, le secret n'influence pas les algorithmes de calcul des couplages utilisés, il s'agit seulement d'un des paramètres sur lequel sont appliqués les algorithmes. En pratique, le couplage se décompose en deux partie distinctes : l'algorithme de Miller et l'exponentiation finale. L'algorithme de Miller est exécuté une ou deux fois suivant que l'on calcule un couplage de type Tate ou Weil. Dans la suite je me focalise sur les couplages de type Tate par souci de concision, mais les attaques s'adaptent au couplage de Weil.

Pour un couplage de type Tate, nous commençons par appliquer au couple (P_A, Id_B) la fonction de Miller $f_{s,P_A}(Id_B)$ (ou $f_{s,Id_B}(P_A)$ pour le couplage Ate). La position du secret peut à première vue influencer la résistance de l'algorithme de couplage soumis à une attaque par canaux auxiliaire. Je reviendrai sur ce fait par la suite. Le résultat $f_{s,P_A}(Id_B)$ est ensuite soumis à l'exponentiation finale à la puissance $\frac{p^k-1}{r}$.

Les informations connues par l'attaquant sont Id_B, l'algorithme de Miller, celui de l'exponentiation finale et le résultat $e(P_A, Id_B)$. Son but est de retrouver P_A.

Il existe plusieurs sortes d'attaques par canaux auxiliaires, je me contenterai ici de décrire le principe des attaques par fautes et par analyse de courant qui sont celles que je connais le mieux. Qui plus est, à l'heure actuelle, ce sont les deux seuls types d'attaques ayant permis de récupérer le secret lors de l'exécution d'un algorithme de couplage.

Lors d'une attaque par injection de fautes, l'espion perturbe l'exécution de l'algorithme en utilisant par exemple une impulsion laser, un champ électromagnétique ou une surtension de l'alimentation. L'objectif est de perturber assez la cible pour que l'exécution de l'algorithme soit faussée mais qu'une sortie puisse être lisible. Ces attaques impliquent d'avoir accès à la cible et de pouvoir exécuter autant de fois qu'il est nécessaire le couplage avec le même secret.

Les attaques par analyse de courant recueillent les courbes de consommation de la cible. L'objectif est de classifier les courbes en utilisant des hypothèses sur le secret puis de les soumettre à un traitement statistique. Il existe différentes attaques par analyse de courant, par exemple l'attaque simple, l'analyse différentielle, l'attaque par corrélation, l'attaque template [134]. Les attaques sont différenciées par le type de traitement statistique auquel sont soumises les courbes de consommation. Je range dans la catégorie des attaques par analyse de courant, les attaques par émission électromagnétique puisque le principe reste le même, exception faite que l'analyse statistique est réalisée sur les émissions électromagnétiques de la cible au lieu des courbes de consommation.

Lorsque des attaques par canaux cachés sont menées contre un algorithme de couplage, l'attaque, pour être un succès, doit réussir à inverser les deux étapes que sont l'algorithme de Miller et l'exponentiation finale. L'attaquant doit pouvoir associer un résultat de l'algorithme de Miller au résultat final du couplage ; et des coordonnées potentielles pour le secret à partir du résultat de Miller. L'exponentiation finale est en théorie impossible à inverser (on utilise une taille d'exposant d'un minimum de 3000 bits). Il n'est pas possible de retrouver la bonne racine $\frac{p^k-1}{r}$ème de $e(P_A, Id_B)$ et tester toutes les possibilités pour la trouver n'est pas non plus envisageable. Ainsi, l'exponentiation finale était considérée comme une contre-mesure naturelle aux attaques par canaux cachés contre les couplages. Néanmoins, il est possible d'oublier l'exponentiation finale, en supposant que l'attaquant utilise une attaque de type saut d'instruction. Ces attaques nécessitent de bien connaître l'architecture interne de la cible mais ne sont pas impossibles [192, 181]. L'exponentiation finale peut aussi être inversée en utilisant deux attaques par fautes [125]. Attaquer par faute un couplage complet (Miller et exponentiation finale) requiert de pouvoir appliquer au moins deux attaques différentes, ce qui est fait par exemple dans les travaux [78].

Mes travaux se sont concentrés sur les attaques contre l'algorithme de Miller et je présente ici brièvement les attaques par analyse de courant que j'ai pu mener à bien [64, 110, 111, 95]⋆. J'ai réalisé la première attaque par analyse de courant contre les couplages durant mon doctorat [64]⋆. Puis de septembre 2014 à septembre 2017, j'ai co-encadré le doctorant Damien Jauvart, avec Louis Goubin et Jacques Fournier. Ensemble nous avons étudié la résistance d'une implémentation du couplage optimal

Ate sur les courbes BN face aux attaques par canaux cachés. En particulier, nous avons amélioré l'analyse des faiblesses des couplages en réalisant en pratique une attaque dite CPA (Correlation Power Analysis), une attaque par profilage contre une implémentation de couplage et une attaque horizontale [110, 111]⋆. Nous avons participé à la rédaction d'un chapitre de livre [73].

Dans le cadre de l'ANR THE CASCADE, j'ai travaillé avec Philippe Guillot, Brandon Dravie et Gilles Millerioux au développement d'une approche spectrale de l'attaque CPA. L'intérêt de notre approche est la réduction de la complexité de l'attaque. Nous obtenons une complexité quasi linéaire en la taille des données, alors que pour la CPA utilisant les attaques statistiques cette complexité est quadratique. Nous illustrons notre amélioration en attaquant une boîte-S du protocole AES [95]⋆.

II.5.2 Attaque contre des implémentations non protégées

Dans un premier temps j'ai mis au point des attaques contre un algorithme de Miller non protégé [63, 67, 188, 69]⋆. Les chapitres de livre [77, 73] présentent les états de l'art des attaques contre les couplages. J'ai décrit la première attaque par faute contre un algorithme de Miller [63]. Le principe de l'attaque est de perturber le nombre d'itérations effectuées durant l'exécution de l'algorithme 3.

Algorithme 3 : Miller(P, Q, s)

Données : $s = (s_n \dots s_0)$ (représentation binaire), $P \in \mathbb{G}_1$ et $Q \in \mathbb{G}_2$;
Résultat : $f_{s,P}(Q) \in \mathbb{G}_3 (\subset \mathbb{F}^{\star}_{p^k})$;

$T \leftarrow P$;
$f \leftarrow 1$;
pour $i = n-1$ **to** 0 **faire**
  $T \leftarrow [2]T$;
  $f \longleftarrow f^2 \times l_T(Q)$;
  où l_T est l'équation de la tangente au point T ;
  si $s_i = 1$ **alors**
    $T \leftarrow T + P$;
    $f \longleftarrow f \times l_{(TP)}(Q)$;
    où $l_{(TP)}$ est l'équation de la droite (TP) ;
  retourner f

Notons $F_{\tau,s,P}(Q)$ le résultat tronqué de l'algorithme de Miller après τ itérations. L'objectif est de parvenir à récupérer deux résultats tronqués distincts d'une seule itération $F_{\tau,s,P}(Q)$ et $F_{\tau+1,s,P}(Q)$. Deux cas sont possibles : soit $\tau+1$ correspond à une étape pendant laquelle seul un doublement est fait, soit un doublement et une addition sont effectués. Le cas simple est le premier, que je me contente de décrire dans la suite. La mise à jour de f s'effectue via le calcul

$$F_{\tau+1,s,P}(Q) = (F_{\tau,s,P}(Q))^2 \times l_T(Q).$$

Ainsi, le rapport $\frac{F_{\tau+1,s,P}(Q)}{(F_{\tau,s,P}(Q))^2}$ correspond tout simplement à l'évaluation de la droite tangente à la courbe elliptique au point T courant évaluée au point Q : $l_T(Q)$.

Il ne reste plus à l'attaquant qu'à déterminer l'équation de l_T et de l'évaluer au point Q. L'attaquant dispose d'une part d'une décomposition théorique de $l_T(Q)$, et d'autre part il a la valeur dans $\mathbb{F}_{p^k}$ de $l_T(Q)$. Une identification des coefficients de la décomposition dans la base choisie de $\mathbb{F}_{p^k}$ lui permet d'obtenir un système en les coordonnées des points publics et privés. La décomposition des paramètres dépend de la réprésentation choisie pour la courbe elliptique, du corps fini et des coordonnées des points. Dans tous les cas, l'attaquant obtient un système non linéaire à résoudre en les coordonnées du point secret. La dimension de ce système dépend du degré de plongement de la courbe elliptique. En pratique, le système est surdéterminé, la dimension est suffisamment grande le résoudre. J'ai ainsi montré que quel que soit le secret, P ou Q, quel que soit le système de représentation utilisé pour la courbe et pour les coordonnées des points de la courbe, il est possible de retrouver le secret [63, 67, 188, 69]*.

La réussite de l'attaque dépend du nombre de tirages nécessaires pour obtenir deux itérations consécutives. J'ai établi une formule de récurrence pour calculer la probabilité d'obtenir deux nombres consécutifs après n tirages parmi N entiers :

$$P(n,N) = 1 - \frac{B(n,N)}{C^n_{n+N}},$$

avec

$$\begin{cases} N \leq 0, n > 0, B(n,N) &= 0, \\ \forall N, n = 0, B(n,N) &= 1, \\ B(n,N) &= \sum_{j=1}^{N} \sum_{k=1}^{n} B(n-k, j-2). \end{cases}$$

En pratique, le nombre de tirages est raisonnable surtout lorsqu'il est comparé aux nombres de courbes nécessaires pour mener à bien une attaque par analyse de courant verticale[5]. Par exemple, pour une architecture 8 bits il suffit de 15 tirages pour trouver deux itérations consécutives, et pour une architecture 16 bits il en suffit de 45.

Le couplage de Miller doit donc être protégé contre les attaques par fautes (et les attaques par analyse de courant).

Plusieurs contre-mesures ont été proposées dans la littérature. Certaines reposent sur la construction de l'algorithme de Miller. L'attaque portant sur la réduction du nombre d'itérations, une contre-mesure évidente serait d'utiliser un compteur afin de vérifier que l'algorithme effectue le bon nombre d'itérations [158, Section 5.3]. Il est aussi possible de tester qu'en sortie de l'algorithme $T = [s]P$ dans le cas du calcul de $f_{s,P}(Q)$ (les rôles de P et Q pouvant être interchangés) [77].

D'autres contre-mesures reposent sur la représentation des éléments intervenant dans le calcul du couplage : il s'agit des contre-mesures de masquages des coordonnées. Pour des raisons d'efficacité, l'algorithme de Miller est implémenté en utilisant des coordonnées homogènes de type projectives ou jacobiennes pour le point T. Par exemple, en coordonnées projectives, le point $T = (X_T, Y_T, Z_T)$ est équivalent au point $T = (\lambda X_T, \lambda Y_T, \lambda Z_T)$, pour tout λ non nul. Ainsi, si le secret est le point sur lequel l'algorithme de Miller est construit, alors pour un surcoût assez minime, la

[5]Pour une attaque par analyse de courant le nombre de courbes nécessaires se comptent par milliers[94].

représentation du point peut être changée à chaque exécution de l'algorithme, ce qui rend a priori impossible l'attaque par faute. Si le secret est le point en lequel les équations de droites sont évaluées, le point Q dans le cas du calcul $f_{s,P}(Q)$, alors il est possible de réécrire l'équation de la ligne pour utiliser une représentation homogène du point Q. Cela implique un surcoût plus important mais néanmoins moins coûteux que les contre-mesures construites en calculant deux couplages, comme ci-après.

Ce dernier type de contre-mesures repose sur la bilinéarité des couplages :

- calculer les deux couplages $R_1 = e(P,Q)$, $R_2 = e(aP,bQ)$ afin de vérifier l'égalité $R_2 = R_1^{ab}$ [77] ;
- choisir a et b tels que $ab = 1 \mod (r)$ afin de calculer $e(P,Q) = e(aP,bQ)$ [159] ;
- choisir un point aléatoire R tel $S = e(P,R)^{-1}$ est défini puis calculer $e(P,Q) = e(P,Q+R)S$.

Ces différentes contre-mesures induisent un surcoût parfois non négligeable pour le calcul du couplage. Ce sont les mêmes pour les attaques par fautes comme pour les attaques par analyse de consommation. Néanmoins ces contre-mesures ne résistent pas toutes aux attaques par canaux cachés.

II.5.3 Attaque contre des implémentations protégées

Dans mes travaux [126, 70]* et l'article de journal [71] je m'intéresse à la résistance des contre-mesures soumises à des attaques par fautes.

Dans le travail [126]* co-écrit avec R. Lasherme, M. Paindavoine, J. Fournier et L. Goubin nous étudions les attaques par fautes contre des implémentations protégées des couplages. Nous validons expérimentalement les attaques par faute contre l'algorithme de Miller ; jusqu'à notre article les attaques étaient uniquement décrites théoriquement. Nous avons en particulier réalisé l'attaque par faute par saut d'instruction. Le modèle de faute modifie le nombre d'itérations effectuées par l'algorithme de Miller pendant l'exécution d'un couplage Ate sur courbe BN, implémenté en C. La cible est un microcontrôleur de type Cortex-M3 32-bit à une fréquence de 56Mhz. Les courbes BN ont un degré de plongement $k = 12$ et admettent une torsion de degré 6. L'algorithme de Miller calcule $f_{s,Q}(P)$. Les fautes sont provoquées à l'aide d'une sonde électromagnétique. La première étape de l'attaque consiste en une cartographie des émissions du microcontrôleur afin de cibler précisément le registre mémoire déterminant le nombre d'itérations de l'algorithme de Miller.

Nous avons étudié la solidité de la contre-mesure par homogénéité des coordonnées. Une analyse approfondie des équations, nous a permis de mettre en évidence que malgré l'aléa λ lié à la modification des coordonnées, nous pouvons retrouver le secret utilisé. L'inclusion de λ dans les équations de droites permet de mettre en évidence deux comportements. Lors d'une étape de doublement, nous pouvons factoriser λ de tel sorte que $l_T^{(\lambda)} = \lambda^{24i} l_T$, où l_T représente l'équation de la tangente en T, $l_T^{(\lambda)}$ est la même avec l'utilisation de l'aléa λ et i est un entier relatif aux nombres d'itérations effectuées. Lors d'une étape d'addition, nous obtenons l'égalité $l_{(TP)}^{(\lambda)} = \lambda^{(9i+12)} l_{(TP)}$, où $l_{(TP)}^{(\lambda)}$ représente l'équation de la droite (TP) avec le masquage λ. Ainsi, le résultat de l'algorithme de Miller utilisant l'homogénéité des

coordonnées peut s'écrire $f_{s,P}(Q)^{(\lambda)} = \lambda^a f_{s,P}(Q)$. Le modèle d'attaque par faute demande deux exécutions de l'algorithme de Miller, cela semble ajouter deux inconnues au système que l'attaquant doit résoudre. En pratique, cela n'ajoute qu'une seule inconnue. En effet, nous avons les égalités suivantes :

$$\begin{aligned} l_T(P)^{(\lambda)} &= \frac{f_{s,Q}^{(\lambda_1)(\tau+1)}(P)}{f_{s,Q}^{(\lambda_2)(\tau)}(P)^2} \\ &= \frac{\lambda_1^a \cdot f_{s,Q}^{(\tau+1)}(P)}{\lambda_2^b \cdot f_{s,Q}^{(\tau)}(P)^2} = \frac{\lambda_1^a}{\lambda_2^b} \cdot l_T(P). \end{aligned} \tag{II.5.1}$$

Soit $L = \frac{\lambda_1^a}{\lambda_2^b}$ la nouvelle inconnue, nous obtenons ainsi le système suivant :

$$\begin{aligned} l_T(P)^{(\lambda)} &= L \cdot ((3X_T^3 - 2Y_T^2) \cdot w^6 + 2Y_T Z_T^3 y_p \cdot w^3 \\ &\quad -3X_T^2 Z_T^2 x_p \cdot w^4), \end{aligned} \tag{II.5.2}$$

$$l_T(P)^{(\lambda)} = R_0^{(\lambda)} + R_3^{(\lambda)} w^3 + R_4^{(\lambda)} w^4. \tag{II.5.3}$$

L'élément w est la racine du polynôme de degré 6 irréductible sur $\mathbb{F}_{p^2}$ permettant de construire $\mathbb{F}_{p^{12}}$ comme une extension de $\mathbb{F}_{p^2}$. Les R_i sont les décompositions du rapport des deux résultats tronqués de l'algorithme de Miller, connus par l'attaquant. Une identification des termes dont nous connaissons le développement théorique et la valeur pratique nous permet d'obtenir le système suivant :

$$\begin{cases} R_0^{(\lambda)} = L \cdot R_0, \\ R_3^{(\lambda)} = L \cdot R_3, \\ R_4^{(\lambda)} = L \cdot R_4. \end{cases}$$

L'équation de la courbe elliptique nous permet d'ajouter une équation au système, que nous résolvons en utilisant les bases de Gröbner.

L'homogénéité des coordonnées n'étant pas une contre-mesure efficace, nous avons recommandé d'implémenter les couplages en utilisant les contre-mesures reposant sur la bilinéarité et nécessitant deux exécutions de l'algorithme de Miller. Sur ce sujet j'ai co-écrit avec R. Lasherme, J. Fournier et L. Goubin, un article paru dans la revue Cryptography and Communication [71]. Dans cette contribution nous rappelons brièvement les propriétés de la cryptographie basée sur l'identité, puis nous présentons les attaques par fautes existant dans la littérature. Nous proposons une série de contre-mesures, lesquelles devraient être résistantes aux attaques par canaux cachés existantes. Cet article est disponible dans l'annexe E.

Un autre de mes travaux, effectué avec Emmanuel Fouotsa, illustre le fait que la contre-mesure de masquage utilisée pour protéger les couplages contre les attaques par fautes n'est pas efficace [70]*. La contre-mesure ciblée consiste à utiliser un point aléatoire pour empêcher l'attaquant d'utiliser l'entrée publique. Le secret sera ici le point P, le point Q est public. Soit R un point tel que le couplage $e(P,R)$ soit défini. Le calcul de $e(P,Q)$ se fait via $\frac{e(P,Q+R)}{e(P,R)}$. Le secret étant P, l'aléa R vient perturber toutes attaques par canaux cachés, puisque l'attaquant n'est plus en

mesure de faire des prédictions sur les résultats intermédiaires. L'attaque par faute ne dépend pas de la façon dont cette contre-mesure est implémentée. Nous avons néanmoins proposé des algorithmes pour exécuter la contre-mesure de la manière la plus efficace possible. L'attaque par faute décrite ci-dessus nous amène à considérer un système non linéaire de $2k+2$ inconnues avec $2k+2$ équations. Le système admet des solutions par construction et les bases de Gröbner devraient pouvoir nous aider à le résoudre. L'inconvénient majeur est le degré des équations. Plus l'algorithme de Miller fait d'itérations, plus le degré du système à résoudre est élevé. Le nombre maximum d'itérations pour lequel nous pouvons résoudre le système est un problème ouvert au sujet duquel nous travaillons encore.

Chapitre III

Conclusion et perspectives

III.1 La cryptographie de demain

Les outils cryptographiques utilisés actuellement (comme RSA) comme les méthodes de certification ne sont plus adaptés à la multitude d'interconnexion de l'internet des objets. Les nouvelles technologies se multiplient pour faciliter notre vie quotidienne : les clés de voiture sont électroniques, le paiement sans contact se popularise. Bien que ces premiers développements soient déjà cassés (piratés), les applications sont nombreuses et cela dans une multitude de domaines tels que la santé, l'environnement, le trafic routier, ferroviaire, maritime, la domotique... Les objets connectés lancent de nouveaux challenges en cryptographie, avec toujours plus de couples clé privée-clé publique, et de très nombreuses données à stocker, manipuler, protéger. Les enjeux économiques sont aussi importants et plus personne ne conteste l'importance de la collecte et du traitement, en très grande quantité, de l'information.

La gestion des clés et des utilisateurs se complique dans le cadre d'applications à l'internet des objets. La gestion des clés par une autorité de certification externe au processus cryptographique impliquerait une croissance exponentielle du nombre de clés et rendrait les applications numériques beaucoup trop lourdes en ressources et temps de calcul. En effet, le couple clé privée-clé publique construit par l'autorité de certification doit être stocké par celle-ci, et donc les quantités à sauvegarder vont croître en fonction du nombre d'utilisateurs et de processus. Ceci n'est pas viable pour le passage à grande échelle puisque cela pose des problèmes de stockage et de ressources de calcul. De plus, l'inconvénient des certificats est la création d'un chemin de confiance pour authentifier l'utilisateur et sa clé.

La cryptographie basée sur l'identité simplifie l'infrastructure de gestion des clés. Dans les schémas basés sur l'identité, l'autorité de confiance ne doit stocker qu'un seul élément : sa clé privée. Les clés privées des utilisateurs sont construites par une exponentiation de leur clé publique par la clé privée de l'autorité de confiance. Le chemin de confiance est plus court et l'adaptabilité des schémas basés sur l'identité plus rapide. Les couplages pourraient ainsi être un outil des plus performants pour les applications de l'internet des objets. De nombreux protocoles basés sur l'identité existent et les implémentations les plus efficaces d'un couplage sont de l'ordre de la milliseconde. Il y a donc un avenir pour les couplages !

Mon expertise dans le domaine de la cryptographie basée sur les couplages m'a

permis d'être éditrice du livre "Guide to pairing-based cryptography", co-édité avec Marc Joye aux éditions CRC Press, Taylor & Francis [65]. Ce livre présente en particulier l'état de l'art de la recherche liée aux couplages en janvier 2016 et chaque chapitre est dévolu à l'un des thèmes de la cryptographie à base de couplages. J'ai personnellement participé, en tant qu'auteur, aux chapitres concernant les mathématiques pour les couplages [23], l'arithmétique [24], le choix des paramètres [57] et les attaques par canaux cachés [73].

Cependant suite à l'attaque NFS de Kim et Barbulescu [118], publiée en août 2016, les bases théoriques de l'implémentation des couplages sont à revoir. Il est évident que l'attaque induit une augmentation des dimensions des sous-groupes de la courbe elliptique ainsi que de l'extension de corps dans laquelle se trouve l'image des couplages. À l'heure actuelle, il est encore difficile de prévoir avec certitude quels couplages et quelles courbes permettront d'obtenir les implémentations logicielles ou matérielles les plus efficaces. L'analyse du choix des courbes et des algorithmes doit être réalisée de nouveau.

Mes perspectives de recherche s'inscrivent pour partie dans cette optique. Je continue de travailler sur les primitives nécessaires aux couplages à travers l'algorithmique, l'arithmétiques des corps finis et des courbes. Je continuerai à explorer les failles des couplages par rapport aux attaques par canaux cachés. C'est sur ces thématiques que je travaillerai avec Asma Chaouch, doctorante en première année. Mes thématiques de recherche s'enrichissent à travers les sujets de thèse de mes deux autres doctorantes. Avec Meriem Smache, j'étudie le déploiement de protocoles sécurisés dans les smarts grid. Avec Amina Bel Korchi, j'étudie la cryptographie homomorphique.

III.1.1 Les smart grids

J'ai initié ma recherche à la thématique des smart grid en encadrant le stage de fin d'étude de Meriem Smache en février 2015.

Les smart grids ou réseau électrique intelligents sont un réseau de distribution d'électricité ou les consommateurs et fournisseurs interagissent en temps réel pour ajuster l'offre et la demande d'électricité afin d'optimiser la production. Les technologies liées aux smart grids sont en cours de développement. Ainsi le rapprochement entre les Nouvelles Technologies de l'Information et de la Communication et les fournisseurs d'énergie fait partie de l'un des enjeux du Plan Numérique France 2012-2020. Pour être efficace, un réseau smart grid devra pouvoir fonctionner à l'échelle d'une ville, voire d'un département, ou même d'un état.

Un des axes de développement dans cette thématique concerne les architectures informatiques à déployer pour autoriser le passage à l'échelle. Ces couches logicielles supérieures de pilotage devront être efficaces et sûres. Par ailleurs, un protocole déployable sur les réseaux smart grids devra gérer l'interconnexion de plusieurs milliards d'appareils et capteurs. Ce qui est complètement différent du cadre classique de la cryptographie. Un défi important de la technologie smart grid est le développement de capteurs embarqués disposant de peu de ressources que ce soit en termes de surface matérielle ou d'énergie disponible. Le réseau devra être résistant en cas de défection d'un ou plusieurs capteurs. Il devra aussi détecter les tentatives de fraudes

ou de manipulation des capteurs.

L'objectif du stage de Meriem était justement de parvenir à étudier les failles par rapport aux attaques physiques d'un protocole de communication sécurisé et efficace. Meriem s'est en premier lieu familiarisée avec les standards CEI 62351 et les recommandations du NIST (Special Publication 800-53 et FIPS 140-02), afin de spécifier les caractéristiques d'un réseau de smart grids. Par la suite, ses travaux ont débuté par le développement d'une couche logicielle et se poursuivent par l'implémentation de simulation d'un réseau de smart grids sécurisé.

Je considère que le stage de Meriem s'est très bien passé. Je continue de la co-encadrer pour sa poursuite d'étude en thèse Cifre avec l'entreprise Gridbee Communication. Son sujet est décrit dans la section IV.4.

III.1.2 La cryptographie homomorphique

La sécurité des données est elle aussi à renforcer, surtout avec le développement des méthodes de stockage dématérialisées comme le cloud. L'ANR JCJC CLÉ m'a permis d'initier ma recherche sur la thématique du chiffrement homomorphique qui est un chiffrement adapté au stockage de données sur le cloud. Il permet de travailler sur les données chiffrées sans avoir à les déchiffrer au préalable. Néanmoins, les protocoles et schémas homomorphiques actuels sont très lourds. Le temps d'exécution de leurs implémentations sont beaucoup trop longs pour permettre leur utilisation en pratique. Le développement de schémas homomorphiques efficaces est probablement un des futurs chantiers de la cryptographie actuelle. Surtout lorsqu'en pratique tous les objets connectés sont des points potentiels de fuite d'information.

Cet intérêt pour la cryptographie homomorphique s'est concrétisé par un stage de Master 2 avec l'entreprise Kontron en mars 2016. Kontron est une entreprise internationale spécialisée dans la conception et la vente de calculateurs embarqués pour les usages industriels, défense, transport et télécommunications. Elle étudie et met en œuvre pour ces domaines les technologies numériques matérielles et logicielles conforment à l'état de l'art. La filiale française, basée à Toulon, collabore dans le cadre de cette thèse avec le CMP (Centre Microélectronique de Provence) dont je suis membre.

L'Internet des objets (IoT pour Internet of Things en anglais) est un nouvel enjeu de la sécurité numérique. En effet, l'IoT implique l'interaction de nombreux objets connectés et le stockage d'informations personnelles et, ou, sensibles. En conséquence de quoi, le chiffrement homomorphique semble être un outil prometteur pour le développement de l'IoT. L'objectif du stage d'Amina Bel Korchi était d'étudier comment les protocoles homomorphiques existants pourraient être déployés pour sécuriser les communications lors d'une utilisation dans le cadre de l'IoT.

Le stage d'Amina se poursuit actuellement par une thèse Cifre. Son sujet est décrit dans la section IV.5.

Chapitre IV

Sujet de thèse des doctorants co-dirigés

IV.1 Amine Mrabet (2014-2017)

Amine était en co-tutelle entre l'université Paris 8 et l'université de Monastir, Tunisie. Je l'ai co-encadré avec Sihem Mesnager en France. Sa thèse était financée par l'ANR SIMPATIC pour les périodes pendant lesquelles il était en France.

Amine a travaillé sur les implémentations pour cartes reprogrammables de type FPGA (Field Programmable Gate Array platforms) de Xlinx de primitives cryptographiques pour les courbes elliptiques. Il a utilisé le langage VHDL pour développer ses composants et ses architectures. Ses résultats présentent de bonnes performances en ressources et en vitesse. Il a tout d'abord développé une multiplication modulaire de Montgomery en combinant l'utilisation de l'algorithme dit CIOS et une architecture systolique. Son architecture permet d'effectuer la multiplication en réduisant le nombre de cycles d'horloge. Il a réussi à obtenir le ratio surface/latence le plus intéressant comparativement à la littérature. Il a implémenté la multiplication pour des corps finis de dimension une puissance de 2, variant de 128 à 512 bits. Ses architectures sont évolutives et dépendent uniquement du nombre et de la taille binaire des mots. Par exemple, il fournit des résultats d'implémentation pour des longs mots de 8, 16, 32 et 64 bits en 33, 66, 132 et 264 cycles d'horloge. Amine a par la suite proposé une architecture pour l'inversion et la division dans les corps finis. En fin de doctorat il a développé l'arithmétique des tours d'extensions impliquées dans le calcul du couplage sur les courbes BN.

Ses travaux ont donné lieu à 1 publication dans un journal [150], 4 articles de conférences avec acte et comité de sélection [148, 149, 151, 152] et 1 chapitre de livre [6].

IV.2 Damien Jauvart (2014-2017)

Damien avait un contrat de thèse DGA à l'université de Versailles, et il se trouvait physiquement au Campus George Charpak Provence à Gardanne. Il était co-encadré par Louis-Goubin (UVSQ), Jacques Fournier (CEA) et moi même.

Damien a travaillé à la sécurisation des algorithmes de couplages contre les attaques physiques. Avant son doctorat, les attaques par canaux auxiliaires contre des algorithmes de couplage étaient majoritairement des attaques par fautes. Damien a étudié comment les couplages résistaient aux attaques de type DPA, DEMA. Il a réalisé les attaques en utilisant les courbes de consommation du couplage optimal Ate sur les courbes BN. Il a considérablement réduit le nombre de courbes nécessaires pour réaliser une attaque par consommation en divisant par dix ce nombre. Il a de plus réalisé une attaque horizontale nécessitant uniquement une courbe de consommation pour récupérer le secret. Damien a proposé des contre-mesures pour résister à ces attaques.

Ses travaux ont donné lieu a 2 publications dans des conférences avec acte et comité de sélection [110, 111] et 1 chapitre de livre [73].

IV.3 Aminatou Pecha (2014-2017)

Aminatou est en co-tutelle entre l'Université Paris 8 et l'Université de Yaoundé, Cameroun. Je co-encadre Aminatou avec Wolfgang Schmidt en France. Ses séjours en France sont financés par le projet PRMAIS qui regroupe plus de Cinq universités en Afrique Subsaharienne pour le développement des mathématiques (Théorie des nombres, géométrie algébrique, théorie des codes) et leurs applications à la sécurité de l'information. Aminatou est d'ailleurs la responsable du pôle "femmes et mathématiques" du projet PRMAIS.

Aminatou travaille à la génération de primitives pour la cryptographie à base de couplage. Elle a étudié les courbes elliptiques dites OEF. Il s'agit de courbes elliptiques définies sur des corps finis de type $\mathbb{F}_{p^m}$, où p est un nombre premier de la taille d'un mot machine et m un entier suffisamment grand pour assurer un bon niveau de sécurité. Aminatou a décrit un algorithme qui doit lui permettre de générer de telle courbe. L'implémentation de l'algorithme étant toujours en cours, elle n'a pas de résultats concrets.

Aminatou a étudié la problématique de l'implémentation du couplage optimal Ate sur les courbes elliptiques de degré de plongement impair [80]. Cet article détermine la complexité algébrique du calcul du couplage optimal Ate pour les courbes elliptiques admettant un degré de plongement $k = 9, 15$ et 27. Aminatou a soigneusement sélectionné les paramètres pour réduire au minimum le nombre d'itérations de l'algorithme de Miller. En particulier, elle réduit d'une inversion dans $\mathbb{F}_{p^{27}}$ le coût d'une boucle de Miller. Les arithmétiques de $\mathbb{F}_{p^9}$, $\mathbb{F}_{p^{15}}$ et $\mathbb{F}_{p^{27}}$ sont détaillées, Aminatou optimise le calcul du Frobenius et des inversions dans les sous-groupes cyclotomiques des corps choisis. Enfin, Aminatou a amélioré le calcul de l'exponentiation finale pour ces courbes, elle réduit de 25% la complexité des calculs pour l'exponentiation finale. Ce travail est en cours d'expertise par un journal.

Enfin Aminatou vient d'avoir un article accepté dans le journal Imohtep. L'objet de ce travail est l'étude du couplage β-Weil.

IV.4 Meriem Smache (2016-2019)

Meriem est en thèse Cifre auprès de l'entreprise Gridbee, le CEA DPACA et l'École des Mines de Saint-Étienne. Je co-encadre Meriem avec Emmanuel Riou (Gridbee), Assia Tria (CEA) et Pierre-Alain Moellic (CEA).

L'objectif de sa thèse est d'évaluer et d'étudier les nouveaux risques d'attaques dans un réseau maillé sans fil intelligent (Smartgrid) dont l'infrastructure est basée sur l'interconnexion de modules de communication GDB1000 développés par l'entreprise Gridbee Communications. Les verrous principaux de la thèse portent d'une part sur la prise en compte de plusieurs milliers d'objets (compteurs électriques, modules de communication et autres dispositifs) communicant en temps réel, et du caractère ouvert et dynamique du réseau maillé sans fil (WMN : Wireless Mesh Network) dans la conception de solutions de cryptage. Ce dynamisme se traduit par la possibilité d'ajout et suppression de nœuds de communication. D'autre part, il s'agit d'adapter ces solutions au contexte applicatif très réglementé (i.e. contraintes sur le contenu et format de messages) des smartgrids. Pour répondre à ces défis, les pistes de recherche envisagées portent sur la conception et l'implémentation d'algorithmes de chiffrement dans un cadre de systèmes multi-agents (sous-domaine de l'intelligence artificielle distribuée). Ces derniers, grâce à leurs propriétés de flexibilité, passage à l'échelle et tolérance aux pannes, étant largement utilisés dans le contrôle et monitoring distribués de systèmes industriels de grande taille et tout particulièrement dans le domaine des smartgrids.

Meriem a publié un article dans une conférence internationale avec actes et comité de sélection [176]. Elle décrit un modèle d'attaque contre un nœud appartenant à un réseau de capteurs sans fil. Elle démontre qu'il est possible de prendre l'ascendant sur le nœud pour récupérer et déchiffrer les informations échangées dans le réseau. Elle conclut l'article par la description d'une contre-mesure reposant sur le développement d'un protocole complet.

IV.5 Amina Bel Korchi (2016-2019)

Amina est en contrat Cifre entre l'entreprise Kontron et l'École des Mines de Saint-Étienne. Je la co-encadre avec Serge Tissot (Kontron) et Dominique Feuillet (EMSE). Dans le contexte de l'internet des objets exigeant une sécurité numérique accrue, cette thèse a pour ambition d'offrir une sécurité des données de bout en bout depuis les capteurs jusqu'à l'analytique et le stockage dans le cloud. En autorisant une protection de bout en bout, cette recherche adresse les spécificités de l'internet des objets : 20 à 50 milliards d'objets en 2020, pas de présence physique d'opérateurs, une longue durée de vie des objets et passerelles réseau, et des piles de logiciel en open source. Les données sont chiffrées dès le départ dans les capteurs, puis jamais exposées en clair excepté à l'intérieur de "secure elements", tout en demeurant actionnables par l'analytique avant ou dans le cloud.

La programmation de "secure elements" au sein desquels les calculs seraient effectués sur les données en clair, dans une chaîne cohérente, avec une cryptographie intégrée homomorphe ou non, permettant de reproduire des avantages proches de

la cryptographie homomorphe pure, représente une approche très innovante adaptée à la sécurité numérique de l'internet des objets. Toutefois, l'option d'utiliser les calculs homomorphes dans un "secure element" permettrait d'exploiter les données chiffrées indifféremment dans le cloud public ou au sein de calculateurs équipés de puces de sécurité homomorphes. Amina a commencé par faire un état de l'art des algorithmes homomorphiques existants. Ensuite, elle a étudié la complexité du chiffrement/déchiffrement, avant de réaliser des calculs sur les données telles que le tri, rechercher le minimum ou le maximum, calculer une moyenne... Elle étudie la complexité de ces calculs. Elle devra, par la suite, réaliser une implémentation pratique de ces opérations en utilisant une bibliothèque open source permettant le calcul homomorphique, par exemple : https ://github.com/shaih/HElib. Cette implémentation permettra de réaliser des benchmarks sur différentes architectures de processeur (SIMD, multi-coeur, CPU, GPU) et d'étudier les possibilités d'accélération des algorithmes pour les inclure dans le "secure element".

IV.6 Asma Chaouch (2017-2020)

Asma est en thèse de co-tutelle entre l'université de Toulon et l'université de Monastir. Je la co-encadre avec Laurent-Stéphane Didier de l'université de Toulon. La communication moderne de nos jours utilise inéluctablement les réseaux informatiques. Il devient possible d'archiver et d'échanger toute sorte d'information, dont les images. En effet, il existe actuellement plusieurs équipements qui sont capables d'acquérir, d'archiver et de transmettre des images avec des coûts très raisonnables ce qui favorise leur utilisation dans plusieurs domaines. De ce fait, le champ du traitement des images devient de plus en plus vaste et il couvre actuellement un très grand nombre d'applications comme la vision par ordinateur, l'imagerie médicale, l'imagerie satellitaire, l'Internet, la compression, la sécurité, la transmission, etc. C'est dans ce contexte que s'inscrit notre travail qui consiste en la combinaison des techniques de compression et de chiffrement d'images. Ce sujet de thèse s'oriente vers la comparaison et l'étude théorique des différentes méthodes de compression et de sécurisation des images. L'objectif de cette thèse est alors la réduction du volume des données afin de désencombrer le maximum possible les réseaux publics de communication et la confidentialité en vue de garantir un niveau de sécurité optimum.

Initialement, Asma va commencer par une étude bibliographique des techniques de chiffrement d'image. Ensuite, Asma recherchera, étudiera et testera des algorithmes existants dans le cadre de cette technique (technique de cryptage symétrique et asymétrique, chiffrement chaotique, sélectif, crypto-compression). L'objectif final sera d'implémenter en langage C/C++ puis sur FPGA les algorithmes les plus performants.

Chapitre V

Publications

Éditrice

(1) N. El Mrabet and M. Joye éditeurs du livre *Guide to Pairing-Based Cryptography*, publié en 2016 aux éditions CRC Press, Taylor & Francis Group.

Chapitres de livre

(1) N. El Mrabet, D. Page and F. Vercauteren *Fault Attack in Pairing Based Cryptography : A state of the art*, in Identity Based Cryptography, Eds M. Joye and M. Tunstall, pp 221 – 236 Springer 2012.

(2) J-L. Beuchat, L. J. Dominguez Perez, S. Duquesne, N. El Mrabet, L. Fuentes-Castañeda, and F. Rodríguez-Henríquez *Arithmetic of Finite Fields* Chap. in Guide to Pairing-Based Cryptography, 2016, Eds N. El Mrabet and M. Joye, CRC Press 2016.

(3) J-L. Beuchat, N. El Mrabet, L. Fuentes-Castañeda, and F. Rodríguez-Henríquez *Mathematical Background* Chap. in Guide to Pairing-Based Cryptography, 2016, Eds N. El Mrabet and M. Joye, CRC Press 2016.

(4) S. Duquesne, N. El Mrabet, S. Haloui, D. Robert and F. Rondepierre *Choosing parameters*, Chap. in Guide to Pairing-Based Cryptography, 2016, Eds N. El Mrabet and M. Joye, CRC Press 2016.

(5) Nadia El Mrabet, Louis Goubin, Sylvain Guilley, Jacques Fournier, Damien Jauvart, Martin Moreau, Pablo Rauzy, and Franck Rondepierre *Physical Attacks* Chap. in Guide to Pairing-Based Cryptography, 2016, Eds N. El Mrabet and M. Joye, CRC Press 2016.

Journaux

(1) N. El Mrabet, S. Duquesne et E. Fouotsa *Efficient Pairings Computation on Jacobi Quartic Elliptic Curves*, dans le **Journal of Mathematical Cryptology**, Associate Eds N. Koblitz, vol. 8, Number 4, pp. 331–362, 2014.

(2) N. El Mrabet, J. Fournier, L. Goubin and R. Lashermes *A survey of Fault attacks in Pairing Based Cryptography*, dans le **journal Cryptography And Communication**, special issue On "Discrete Structures For Side Channel Attacks And Their Counter-Measures", Associate Eds C. Carlet and P-A. Fouque, vol. 7, Number 1, pp. 185–205, 2015.

(3) N. El Mrabet and L. Poinsot *Harmonic Analysis and a Bentness-Like Notion in Certain Finite Abelian Groups Over Some Finite Fields* in **Malaysian Journal of Mathematical Sciences**, Special Issue : The 4th International Cryptology and Information Security Conference 2014, vol. 9, pp. 1–20 Juin 2015 (ISSN 1823-8343).

(4) S. Duquesne, N. El Mrabet, S. Haloui and F. Rondepierre *Choosing and generating parameters for pairing implementation on BN curves*, **Applicable Algebra in Engineering, Communication and Computing**, pp. 1–35, 2017.

(5) A. Mrabet, N. El Mrabet, R. Lashermes, J-B. Rigaud, B. Bouallegue, S. Mesnager and M. Machhout. *A scalable and systolic architectures of montgomery modular multiplication for public key cryptosystems based on dsps.*, **Journal of Hardware and Systems Security** (accepté en Aout 2017).

Conférences internationales avec actes et comité de sélection

(1) J.-C. Bajard et N. El Mrabet, *Pairing in Cryptography : an Arithmetic Point of View* présenté lors de la conférence Advanced Signal Processing Algorithms, Architectures, and Implementations XVI, part of SPIE proceedings, pp 1–11, en aout 2007.

(2) N. El Mrabet *What about vulnerability to a fault attack of the Miller algorithm during an Identity Based Protocol ?*, paru dans le proceeding LNCS de ISA 2009, The 3rd International Conference on Information Security and Assurance, du 25 au 27 juin 2009, LNCS vol. 5576, pp 122–134.

(3) N. El Mrabet et C. Negre *Finite field multiplication combining AMNS and DFT approach for pairing cryptography*, paru dans le proceeding LNCS de ACISP 2009, The 14th Australian Conference on Information Security and Privacy, du 1$^{\text{er}}$ au 3 juillet 2009, à Brisbane, Australie, LNCS vol. 5594, pp 422–436.

(4) N. El Mrabet, M-L. Flottes et Giogio Di Natale *A practical Differential Power Analysis Attack against the Miller Algorithm*, actes disponibles sur le site IEEEXplore de Prime 2009, The 5th International Conference on Ph.D. Research in Microelectronics & Electronics.

(5) N. El Mrabet *Fault attacks against the Miller algorithm in Edwards Coordinates ?*, proceeding LNCS de ISA 2010, The 4th International Conference on Information Security and Assurance, Springer CCIS vol. 76, pp 72–85.

(6) J. Boxall, N. El Mrabet, F. Laguillaumie and P. Le Duc *A Variant of Miller's Formula and Algorithm*, proceedings LNCS de Pairing 2010, LNCS vol. 6487, pp 417–434.

(7) N. El Mrabet, A. Guillevic and S. Ionica *Efficient Multiplication in Finite Field Extensions of Degree 5*, proceedings LNCS de AfricaCrypt 2011, LNCS vol. 6737, pp 188–205.

(8) J. Weng, Y. Dou, C. Ma and N. El Mrabet *Fault Attacks against the Miller Algorithm in Hessian Coordinates*, proceedings LNCS de Information Security and Cryptology, Inscrypt 2011, LNCS vol. 7537, pp. 102–112, Springer 2011.

(9) N. El Mrabet and N. Gama *Efficient Multiplication over Extension Fields*, proceedings LNCS de WAIFI 2012, Workshop on Arithmetic on Finite Fields, LNCS vol. 7369, pp 136–151.

(10) N. El Mrabet *Side Channel Attacks against Pairing over Theta Functions*,proceedings LNCS de CAI 2013, Conference on Algebraic Mathematique LNCS vol. 8080, pp 132–146.

(11) A. Mrabet, B. Bouallegue, M. Mohsen, N. El Mrabet, S. Mesnager and R. Tourki *Implementation of Faster Miller over BN curves in Jacobian Coordinates*, GSCIT'2014, actes disponibles sur le site IEEE Xplore.

(12) N. El Mrabet and L. Poinsot *Harmonic Analysis and a Bentness-like Notion in Certain Finite Abelian Groups Over Some Finite Fields*, Cryptology2014, version étendue publiée dans le Malaysian Journal of Mathematical Sciences, Special Issue : The 4th International Cryptology and Information Security Conference 2014, vol. 9, pp. 1–20 Juin 2015 (ISSN 1823-8343).

(13) R. Lashermes, M. Paindavoine, N. El Mrabet, J. Fournier and L. Goubin *Practical Validation of Several Fault Attacks against the Miller Algorithm*, FDTC 2014, Fault Diagnosis and Tolerance in Cryptography, proceedings IEEE computer Society, pp. 115–122, 2014.

(14) L. Duc-Phong, N. El Mrabet and T. Chik How *On Near Prime-Order Elliptic Curves with Small Embedding Degrees*, Conference on Algebraic Informatics CAI 2015, LNCS vol. 9270, pp. 140–151, Springer 2015.

(15) N. El Mrabet and E. Fouotsa *Failure of the Point Blinding counter-measure Against Fault Attack in Pairing-Based Cryptography*, proceedings LNCS de C2SI 2015, Codes, Cryptology, and Information Security 2015, LNCS vol. 9084, pp. 259–273, Springer 2015.

(16) D. Jauvart, J. J. A. Fournier, N. El Mrabet and L. Goubin : *Improving Side-Channel Attacks Against Pairing-Based Cryptography*. proceedings LNCS de CRiSIS 2016 : pp. 199–213, LNCS 2016.

(17) A. Mrabet, N. El Mrabet, R. Lashermes, J-B. Rigaud, B. Bouallegue, S. Mesnager and M. Machhout : *High-performance elliptic curve cryptography by using the cios method for modular multiplication*, proceedings LNCS de CRISIS 2016 : 11th International Conference on Risks and Security of Internet and Systems, pp. 185–198, LNCS 2016.

(18) A. Mrabet, N. El Mrabet, R. Lashermes, J-B. Rigaud, B. Bouallegue, S. Mesnager and M. Machhout : *A scalable and systolic architectures of montgomery modular multiplication for public key cryptosystems based on dsps.*, proceedings LNCS de The Sixth International Conference on Security, Privacy and Applied Cryptographic Engineering (SPACE 2016), pp. 138–156, LNCS 2016.

(19) M Smache, N El Mrabet, A Tria, JJ GIL-QUIJANO, E Riou, G Chaput *Modeling a node capture attack in a secured Wireless Sensor Network*, proceedings IEEE du 3rd World Forum on Internet of Things, 2016.

(20) A. Mrabet, N. El Mrabet B. Bouallegue, S. Mesnager and M. Machhout. *An efficient and scalable modular inversion and division for public key cryptosystems*, proceedings IEEE de l'International Conference on Engineering and Management Information Systems ICEMIS 2017.

(21) Philippe Guillot, Gilles Millérioux, Brandon Dravie and Nadia El Mrabet : *Spectral Approach for Correlation Power Analysis*, proceedings LNCS de C2SI 2017 : pp. 238–253, LNCS 2017.

Annexe A

A Variant of Miller's Formula and Associated Algorithm

PAIRING 2010 [39]
avec John Boxall, Fabien Laguillaumie et Duc-Phong Le

Abstract : Miller's algorithm is at the heart of all pairing-based cryptosystems since it is used in the computation of pairing such as that of Weil or Tate and their variants. Most of the optimizations of this algorithm involve elliptic curves of particular forms, or curves with even embedding degree, or having an equation of a special form. Other improvements involve a reduction of the number of iterations.

In this article, we propose a variant of Miller's formula which gives rise to a generically faster algorithm for any pairing friendly curve. Concretely, it provides an improvement in cases little studied until now, in particular when denominator elimination is not available. It allows for instance the use of elliptic curve with embedding degree not of the form 2^i3^j, and is suitable for the computation of optimal pairings. We also present a version with denominator elimination for even embedding degree. In our implementations, our variant saves between 10% and 40% in running time in comparison with the usual version of Miller's algorithm without any optimization.

A.1 Introduction

Pairings were first introduced into cryptography in Joux' seminal paper describing a tripartite (bilinear) Diffie-Hellman key exchange [113]. Since then, the use of cryptosystems based on bilinear maps has had a huge success with some notable breakthroughs such as the first identity-based encryption scheme [36]. Nevertheless, pairing-based cryptography has a reputation of being inefficient, because it is computationally more expensive than cryptography based on modular arithmetic. On the other hand, the use of pairings seems to be essential in the definition of protocols with specific security properties and also allows one to reduce bandwidth in certain protocols.

Ever since it was first described, Miller's algorithm [143] has been the central ingredient in the calculation of pairings on elliptic curves. Many papers are devoted

to improvements in its efficiency. For example, it can run faster when the elliptic curves are chosen to belong to specific families (see for example [18, 20, 47]), or different coordinate systems (see for example [106, 46, 22]). Another standard method of improving the algorithm is to reduce the number of iterations by introducing pairings of special type, for example particular optimal pairings [183, 101, 100] or using addition chains (see for example [28]).

In this paper we study a variant of Miller's algorithm for elliptic curves which is generically faster than the usual version. Instead of using the formula $f_{s+t} = f_s f_t \frac{\ell_{s,t}}{v_{s+t}}$ (see Subsection A.2.2 for notation and Lemma B.2.1) on which the usual Miller algorithm is based, our variant inspired by the formula $f_{s+t} = \frac{1}{f_{-s} f_{-t} \ell_{-s,-t}}$ (see Lemma A.3.1 for a proof). An important feature is that the only vertical line that appears is f_{-1}, in other words the vertical line passing through P, and even this does not appear explicitly except at initialization. We shall see in § A.3.1 why it does not appear in the addition step. Our algorithm is of particular interest to compute the Ate-style pairings [16, 101] on elliptic curves with small embedding degrees k, and in situations where denominator elimination using a twist is not possible (for example on curves with embedding degree prime to 6). A typical example is the case of optimal pairings [183], which by definition only require about $\log_2(r)/\varphi(k)$ (where r is the group order) iterations of the basic loop. If k is prime, then $\varphi(k \pm 1) \leq \frac{k+1}{2}$ which is roughly $\frac{\varphi(k)}{2} = \frac{k-1}{2}$, so that at least twice as many iterations are necessary if curves with embedding degrees $k \pm 1$ are used instead of curves of embedding degree k.

The paper is organized as follows. In Section A.2 we recall some background on pairings, and recall the usual Miller algorithm (Figure A.1). In Section A.3 we explain and analyze generically our version of Miller's algorithm, which is resumed by the pseudocode in Figure A.2 when the elliptic curve is given in Jacobian coordinates. Section A.4 discusses a variant without denominators applicable when k is even (see Figure A.5). Section A.5 describes some numerical experiments and running times; in an example with $k = 18$ and r having 192 bits, the algorithm of Figure A.5 is roughly 40% faster than the usual Miller algorithm, and about as fast as the algorithm of [18].

Further work is needed to see whether many of the recent ideas used to improve the usual Miller algorithm can be adapted to the variant presented here. We believe that doing so should lead to further optimizations.

A.2 Background on Pairings

A.2.1 Pairings

We briefly recall the basic definitions and some examples of pairings used in cryptography. For further information, see for example [29, 44].

We let $r \geq 2$ denote an integer which, unless otherwise stated, is supposed to be prime. We let $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$ and $(\mathbb{G}_T, \cdot)$ denote three finite abelian groups, which are supposed to be of order r unless otherwise indicated. A pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ such that $e(P_1 + P_2, Q) = e(P_1, Q)e(P_2, Q)$ and $e(P, Q_1 + Q_2) = e(P, Q_1)e(P, Q_2)$ for all P, P_1, $P_2 \in \mathbb{G}_1$ and for all Q, Q_1, $Q_2 \in \mathbb{G}_2$. We say that the

pairing e is left non degenerate if, given $P \in \mathbb{G}_1$ with $P \neq 1$, there exists $Q \in \mathbb{G}_2$ with $e(P,Q) \neq 1$. The notion of a right degenerate pairing is defined similarly and e is said to be non degenerate if it is both left and right non degenerate

We recall briefly one of the most frequent choices for the groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ in pairing-based cryptography. Here, $\mathbb{G}_1$ is the group generated by a point P of order r on an elliptic curve E defined over a finite field $\mathbb{F}_q$ of characteristic different to r. Thus, $\mathbb{G}_1 \subseteq E(\mathbb{F}_q)$ is cyclic of order r but, in general, the whole group $E[r]$ of points of order dividing r of E is not rational over $E(\mathbb{F}_q)$. Recall that the embedding degree of E (with respect to r) is the smallest integer $k \geq 1$ such that r divides $q^k - 1$. A result of Balasubramanian and Koblitz [11] asserts that, when $k > 1$, all the points of $E[r]$ are rational over the extension $\mathbb{F}_{q^k}$ of degree k of $\mathbb{F}_q$. The group $\mathbb{G}_2$ is chosen as another subgroup of $E[r]$ of order r. Finally, $\mathbb{G}_T$ is the subgroup of order r in $\mathbb{F}_{q^k}^\times$; it exists and is unique, since r divides $q^k - 1$ and $\mathbb{F}_{q^k}^\times$ is a cyclic group.

Let $P \in E(\mathbb{F}_q)$ be an r-torsion point, let D_P be a degree zero divisor with $D_P \sim [P] - [O_E]$, and let f_{r,D_P} be such that $\operatorname{div} f_{r,D_P} = rD_P$. Let Q be a point of $E(\mathbb{F}_{q^k})$ (not necessarily r-torsion) and $D_Q \sim [Q] - [O_E]$ of support disjoint with D_P. Consider

$$e_r^T(P,Q) = f_{r,D_P}(D_Q). \tag{A.2.1}$$

Weil reciprocity shows that if D_Q is replaced by $D'_Q = D_Q + \operatorname{div} h \sim D_Q$, then (A.2.1) is multiplied by $h(D_P)^r$. So the value is only defined up to r-th powers. Replacing D_P by $D'_P = D_P + \operatorname{div} h$ changes f_{r,D_P} to $f_{r,D'_P} = f_{r,D_P} h^r$, and the value is well-defined modulo multiplication by r-th powers. If then Q is replaced by $Q + rR$, the value changes again by an r-th power. This leads to adapting the range and domain of e_r^T as follows.

Théorème A.2.1. *The Tate pairing is a map*

$$e_r^T : E(\mathbb{F}_q)[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}^\times/(\mathbb{F}_{q^k}^\times)^r$$

satisfying the following properties :

1. *Bilinearity,*
2. *Non-degeneracy,*
3. *Compatibility with isogenies.*

The reduced Tate pairing computes the unique rth root of unity belonging to the class of $f_{r,D_P}(D_Q)$ modulo $(\mathbb{F}_{q^k}^\times)^r$ as $f_{r,D_P}(D_Q)^{(q^k-1)/r}$. In practice, we take Q to lie in some subgroup $\mathbb{G}_2$ of order r of $E(\mathbb{F}_{q^k})$ that injects into $E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k})$ via the canonical map. The more popular Ate pairing [16] and its variants (see [137] for instance) are optimized versions of the Tate pairing when restricted to Frobenius eigenspaces. Besides its use in cryptographic protocols, the Tate pairing is also useful in other applications, such as walking on isogeny volcanoes [107], which can be used in the computation of endomorphism rings of elliptic curves.

However, in this article we concentrate on the computation of $f_{n,D_P}(D_Q)$ (which we write as $f_{n,P}(Q)$ in the sequel). This is done using Miller's algorithm described in the next subsection.

A.2.2 Computation of Pairings and Miller's Algorithm

In order to emphasize that our improvement can be applied in a very general context, we explain briefly in this subsection how pairings are computed. In what follows, $\mathbb{F}$ denotes a field (not necessarily finite), E an elliptic curve over $\mathbb{F}$ and r an integer not divisible by the characteristic of $\mathbb{F}$. We suppose that the group $E(\mathbb{F})$ of $\mathbb{F}$-rational points of E contains a point P of order r. Since r is prime to the characteristic of $\mathbb{F}$, the group $E[r]$ of points of order r of E is isomorphic to a direct sum of two cyclic groups of order r. In general, a point $Q \in E[r]$ that is not a multiple of P will be defined over some extension $\mathbb{F}'$ of $\mathbb{F}$ of finite degree.

If P, P' are two points in $E(\mathbb{F})$, we denote by $\ell_{P,P'}$ a function with divisor $[P] + [P'] + [-(P + P')] - 3[O_E]$ and by v_P a function with divisor $[P] + [-P] - 2[O_E]$. Clearly these functions are only defined up to a multiplicative constant ; we recall at the end of this section how to normalize functions so that they are uniquely determined by their divisor. Note that v_P is just the same as $\ell_{P,-P}$.

If s and t are two integers, we denote by $f_{s,P}$ (or simply f_s if there is no possibility of confusion) a function whose divisor is $s[P] - [sP] - (s-1)[O_E]$. We abbreviate $\ell_{sP,tP}$ to $\ell_{s,t}$ and v_{sP} to v_s. As understood in this paper, the purpose of Miller's algorithm is to calculate $f_{s,P}(Q)$ when $Q \in E[r]$ is not a multiple of P. All pairings can be expressed in terms of these functions for appropriate values of s.

Miller's algorithm is based on the following Lemma describing the so-called Miller's formula, which is proved by considering divisors.

Lemme A.2.2. *For s and t two integers, up to a multiplicative constant, we have $f_{s+t} = f_s f_t \frac{\ell_{s,t}}{v_{s+t}}$.*

The usual Miller algorithm makes use of Lemma B.2.1 with $t = s$ in a doubling step and $t = 1$ in an addition step. It is described by the pseudocode in Figure A.1, which presents the algorithm updating numerators and denominators separately, so that just one inversion is needed at the end. We write the functions ℓ and v as quotients $(N\ell)/(D\ell)$ and $(Nv)/(Dv)$, where each of the terms $(N\ell)$, $(D\ell)$, (Nv), (Dv) is computed using only additions and multiplications, and no inversions. Here the precise definitions of $(N\ell)$, $(D\ell)$, (Nv), (Dv) will depend on the representations that are used ; in Section A.3.2 we indicate one such choice when short Weierstrass coordinates and the associated Jacobian coordinates are used. In the algorithm, T is always a multiple of P, so that the hypothesis that Q is not a multiple of P implies that at the functions $\ell_{T,T}$, $\ell_{T,P}$, v_{2T} and v_{T+P} cannot vanish at Q. It follows that f and g never vanish at Q so that the final quotient f/g is well-defined and non-zero.

Obviously in any implementation it is essential for the functions appearing in the programs to be uniquely determined, and so we end this section by recalling briefly how this can be done in our context. If w is a uniformizer at O_E, we say that the non-zero rational function f on E is normalized (or monic) with respect to w if the Laurent expansion of f at O_E is of the form

$$f = w^n + c_{n+1}w^{n+1} + c_{n+2}w^{n+2} + \cdots, \quad c_i \in \mathbb{F},$$

(i. e. if the first non-zero coefficient is 1). Any non-zero rational function on E can be written in a unique way as a product of a constant and a normalized function,

Algorithme 4 : Miller(P, Q, s) usual

Data : $s = \sum_{i=0}^{l-1} s_i 2^i$ (radix 2), $s_i \in \{0,1\}$, $Q \in E(\mathbb{F}')$ not a multiple of P.
Result : $f_{s,P}(Q)$.

$T \leftarrow P$, $f \leftarrow 1$, $g \leftarrow 1$,;
for $i = l-2$ **to** 0 **do**
  $f \leftarrow f^2(\overline{N\ell})_{T,T}(Dv)_{2T}$,;
  $g \leftarrow g^2(D\ell)_{T,T}(Nv)_{2T}$,;
  $T \leftarrow 2T$;
  if $s_i = 1$ **then**
    $f \leftarrow f(N\ell)_{T,P}(Dv)_{T+P}$,;
    $g \leftarrow g(D\ell)_{T,P}(Nv)_{T+P}$,;
    $T \leftarrow T + P$;
return f/g

FIGURE A.1 – The usual Miller algorithm

and the normalized rational functions form a group under multiplication that is isomorphic to the group of principal divisors.

As a typical example, when E is in short Weierstrass form

$$y^2 = x^3 + ax + b,\ a, b \in \mathbb{F} \tag{A.2.2}$$

one can take $w = \frac{x}{y}$. Any rational function on E can be written as a quotient of two functions whose polar divisors are supported at the origin O_E of E. If f is a function whose only pole is at O_E, then there exist two polynomials $U(x)$ and $V(x)$ such that $f = U(x) + V(x)y$: here U and V are uniquely determined by f. Furthermore, if the order of the pole of f is n, then $n \geq 2$ and $U(x)$ is of degree $\frac{n}{2}$ and $V(x)$ of degree at most $\frac{n-4}{2}$ if n is even and $V(x)$ is of degree $\frac{n-3}{2}$ and $U(x)$ of degree at most $\frac{n-1}{2}$ when n is odd. Then f is normalized if and only if, when f is written in the form $U(x) + V(x)y$, $U(x)$ is monic when n is even and $V(x)$ is monic when n is odd. In general, a rational function on E is normalized if and only if it is a quotient of two normalized functions whose polar divisors are supported by the origin.

For example, when E is given by the equation (A.2.2), the functions

$$\ell_{T,P}(x,y) = \begin{cases} y - y_P - \frac{y_T - y_P}{x_T - x_P}(x - x_P), & T \neq O_E,\ \pm P, \\ y - y_P - \frac{3x_P^2 + a}{2y_P}(x - x_P), & T = P,\ 2P \neq O_E. \end{cases}$$

and

$$v_P(x,y) = x - x_P$$

are normalized, so that if they are used in implementations of the algorithms given in Figures A.1 and A.2, then these algorithms output $f_{s,P}(Q)$ with $f_{s,P}$ the normalized function with divisor $s[P] - [sP] - (s-1)[O_E]$.

A.3 Our Variant of Miller's Algorithm

In this section, we describe our variant of Miller's formula and algorithm and analyze the cost of the latter in terms of basic operations.

A.3.1 The Algorithm

The main improvement comes from the following Lemma.

Lemme A.3.1. *For s and t two integers, up to a multiplicative constant, we have*

$$f_{s+t} = \frac{1}{f_{-s} f_{-t} \ell_{-s,-t}}.$$

Démonstration. This lemma is again proved by considering divisors. Indeed,

$$\begin{aligned}
\operatorname{div}(f_{-s} f_{-t} \ell_{-s,-t}) &= (-s)[P] - [(-s)P] + (s+1)[O_E] \\
&\quad + (-t)[P] - [(-t)P] + (t+1)[O_E] \\
&\quad + [-sP] + [-tP] + [(s+t)P] - 3[O_E] \\
&= -(s+t)[P] + [(s+t)P] + (s+t-1)[O_E] \\
&= -\operatorname{div}(f_{s+t}),
\end{aligned}$$

which concludes the proof. □ □

We shall seek to exploit the fact that here the right hand member has only three terms whereas that of Lemma B.2.1 has four.

Our variant of Miller's algorithm is described by the pseudocode in Figure A.2.

It was inspired by the idea of applying Lemma A.3.1 with $t = s$ or $t \in \{\pm 1\}$. However, the scalar input is given in binary representation. It updates numerators and denominators separately, so that only one final inversion appears at the end. As in Figure A.1, the hypothesis that Q is not a multiple of P implies that at no stage do f and g vanish, so that the final quotient f/g makes sense. Note that T is always a positive multiple of P. We use the notation $\ell'_{-T,-P}$ for the function $f_{-1}\ell_{-T,-P}$, since in many situations it can be computed faster than simply by computing f_{-1} and $\ell_{-T,-P}$ and taking the product. For example, when E is given in short Weierstrass coordinates by the equation $y^2 = x^3 + ax + b$, $a, b \in \mathbb{F}$, we have

$$\ell'_{-T,-P} = f_{-1}\ell_{-T,-P} = \frac{1}{x_Q - x_P}\big(y_Q + y_P + \lambda(x_Q - x_P)\big) = \frac{y_Q + y_P}{x_Q - x_P} + \lambda, \qquad \text{(A.3.1)}$$

where $\frac{y_Q+y_P}{x_Q-x_P}$ can be precomputed (at the cost of one inversion and one multiplication in the big field) and λ denotes the slope of the line joining T to P.

The tables in Figures A.3 and A.4 show that our variant is more efficient than the classical Miller's algorithm as we save a product in the big field at each doubling and each addition step. We also save some multiplications and squarings in $\mathbb{F}$. The following subsection discusses all this in more detail. In Section A.4 we describe a version without denominators that works for elliptic curves with even embedding degree.

Algorithme 5 : Miller(P, Q, r) modified

Data : $s = \sum_{i=0}^{l-1} s_i 2^i$, $s_i \in \{0, 1\}$, $s_{l-1} = 1$, h Hamming weight of s, $Q \in E(\mathbb{F}')$ not a multiple of P

Result : $f_{s,P}(Q)$;

1 $f \leftarrow 1$, $T \leftarrow P$, ;
2 **if** $l + h$ is odd **then**
3 $\quad \delta \leftarrow 1$, $g \leftarrow f_{-1}$;
4 **else**
5 $\quad \delta \leftarrow 0$, $g \leftarrow 1$;
6 **for** $i = l - 2$ **to** 0 **do**
8 $\quad$ **if** $\delta = 0$ **then**
9 $\quad\quad f \leftarrow f^2 (N\ell)_{T,T}$,;
10 $\quad\quad g \leftarrow g^2 (D\ell)_{T,T}$,;
11 $\quad\quad T \leftarrow 2T$, $\delta \leftarrow 1$;
13 $\quad\quad$ **if** $s_i = 1$ **then**
14 $\quad\quad\quad g \leftarrow g(N\ell')_{-T,-P}$,;
15 $\quad\quad\quad f \leftarrow f(D\ell')_{-T,-P}$,;
16 $\quad\quad\quad T \leftarrow T + P$, $\delta \leftarrow 0$;
18 $\quad$ **else**
19 $\quad\quad g \leftarrow g^2 (N\ell)_{-T,-T}$,;
20 $\quad\quad f \leftarrow f^2 (D\ell)_{-T,-T}$,;
21 $\quad\quad T \leftarrow 2T$, $\delta \leftarrow 0$;
23 $\quad\quad$ **if** $s_i = 1$ **then**
24 $\quad\quad\quad f \leftarrow f(N\ell)_{T,P}$,;
25 $\quad\quad\quad g \leftarrow g(D\ell)_{T,P}$,;
26 $\quad\quad\quad T \leftarrow T + P$, $\delta \leftarrow 1$;
27 **return** f/g

FIGURE A.2 – Our modified Miller algorithm

A.3.2 Generic Analysis

In this subsection, we compare the number of operations needed to compute $f_{s,P}(Q)$ using the algorithms in Figures A.1 and A.2. In order to fix ideas, we make our counts using Jacobian coordinates (X, Y, Z) associated to a short Weierstrass model $y^2 = x^3 + ax + b$, $a, b \in \mathbb{F}$, so that $x = X/Z^2$ and $y = Y/Z^3$. We suppose that the Jacobian coordinates of P lie in $\mathbb{F}$ and that those of Q lie in some extension $\mathbb{F}'$ of $\mathbb{F}$ of whose degree is denoted by k. We denote by $\mathbf{m}_a$ the multiplication by the curve coefficient a and we denote respectively by $\mathbf{m}$ and $\mathbf{s}$ multiplications and squares in $\mathbb{F}$, while the same operations in $\mathbb{F}'$ are denoted respectively by $\mathbf{M}_k$ and $\mathbf{S}_k$ if k is the degree of the extension $\mathbb{F}'$. We assume that $\mathbb{F}'$ is given by a basis as a $\mathbb{F}$-vector space one of whose elements is 1, so that multiplication of an element of $\mathbb{F}'$ by an element of $\mathbb{F}$ counts as k multiplications in $\mathbb{F}$. We ignore additions and multiplications by small integers.

If S is any point of E, then X_S, Y_S and Z_S denote the Jacobian coordinates of S, so that when $S \neq O_E$, the Weierstrass coordinates of S are $x_S = X_S/Z_S^2$ and $y_S = Y_S/Z_S^3$. As before, T is a multiple of P, so that X_T, Y_T and Z_T all lie in $\mathbb{F}$. Since P and Q are part of the input, we assume they are given in Weierstrass coordinates and that $Z_P = Z_Q = 1$.

We need to define the numerators and the denominators of the quantities appearing in the algorithms of Figures A.1 and A.2. The cost of computing these quantities and the total cost of these algorithms are analyzed in Figures A.3 and A.4.

The doubling step.

We first deal with doubling. Suppose that $2T \neq O_E$, which will always be the case if r is odd. Then $y_T \neq 0$ and the slope of the tangent to E at T is

$$\mu_T = \frac{3x_T^2 + a}{2y_T} = \frac{(N\mu)_T}{(D\mu)_T},$$

where $(N\mu)_T = 3X_T^2 + aZ_T^4$ and $(D\mu)_T = 2Y_TZ_T = (Y_T + Z_T)^2 - Y_T^2 - Z_T^2$. Hence the value of $\ell_{T,T}$ at Q can be written

$$\ell_{T,T}(Q) = y_Q - y_T - \mu_T(x_Q - x_T) = \frac{(N\ell)_{T,T}}{(D\ell)_{T,T}},$$

where now

$$(N\ell)_{T,T} = (D\mu)_TZ_T^2y_Q - (N\mu)_TZ_T^2x_Q - 2Y_T^2 - (N\mu)_TX_T$$

and

$$(D\ell)_{T,T} = (D\mu)_TZ_T^2.$$

The coordinates X_{2T}, Y_{2T} and Z_{2T} of $2T$ are given by

$$\begin{cases} X_{2T} = (N\mu)_T^2 - 8X_TY_T^2, \\ Y_{2T} = (N\mu)_T(4X_TY_T^2 - X_{2T}) - 8Y_T^4, \\ Z_{2T} = (D\mu)_T. \end{cases}$$

Hence the value of v_{2T} at Q can be calculated as

$$v_{2T}(Q) = x_Q - x_{2T} = \frac{(Nv)_{2T}}{(Dv)_{2T}},$$

where $(Dv)_{2T} = Z_{2T}^2$ and $(Nv)_{2T} = Z_{2T}^2 x_Q - X_{2T}$.

In the modified algorithm, we also need to compute $\ell_{-T,-T}$ at Q. But the coordinates of $-T$ are $(x_T, -y_T)$, so that we can write

$$\ell_{-T,-T}(Q) = y_Q + y_T + \mu_T(x_Q - x_T) = \frac{(N\ell)_{-T,-T}}{(D\ell)_{-T,-T}},$$

with

$$(N\ell)_{-T,-T} = (D\mu)_T Z_T^2 y_Q + (N\mu)_T Z_T^2 x_Q + 2Y_T^2 - (N\mu)_T X_T$$

and

$$(D\ell)_{-T,-T} = (D\mu)_T Z_T^2.$$

All the operations needed in the doubling steps of the algorithms in Figures A.1 and A.2 are shown in detail in Figure A.3. The quantities are to be computed in the order shown in the table, a blank entry indicating that the corresponding quantity need not be computed in the case indicated at the top of the corresponding column. The costs of the computations are calculated assuming that the results of intermediate steps are kept in memory. An entry 0 indicated that the quantity has already been calculated at a previous stage.

Quantity	Formula	Classical Miller (Fig. A.1)	Modified Miller (Fig. A.2, loop **1**)	Modified Miller (Fig. A.2, loop **3**)
$(N\mu)_T$	$3X_T^2 + aZ_T^4$	$\mathbf{m}_a + 3\mathbf{s}$	$\mathbf{m}_a + 3\mathbf{s}$	$\mathbf{m}_a + 3\mathbf{s}$
$(D\mu)_T$	$2Y_T Z_T = (Y_T + Z_T)^2 - Y_T^2 - Z_T^2$	$2\mathbf{s}$	$2\mathbf{s}$	$2\mathbf{s}$
$(N\ell)_{T,T}$	$(D\mu)_T Z_T^2 y_Q - (N\mu)_T Z_T^2 x_Q$ $-2Y_T^2 - (N\mu)_T X_T$	$(3+2k)\mathbf{m}$	$(3+2k)\mathbf{m}$	
$(D\ell)_{T,T}$	$(D\mu)_T Z_T^2$	0	0	
X_{2T}	$(N\mu)_T^2 - 8X_T Y_T^2$	$\mathbf{s} + \mathbf{m}$	$\mathbf{s} + \mathbf{m}$	$\mathbf{s} + \mathbf{m}$
Y_{2T}	$(N\mu)_T(4X_T Y_T^2 - X_{2T}) - 8Y_T^4$	$\mathbf{s} + \mathbf{m}$	$\mathbf{s} + \mathbf{m}$	$\mathbf{s} + \mathbf{m}$
Z_{2T}	$(D\mu)_T$	0	0	0
$(Dv)_{2T}$	Z_{2T}^2	$\mathbf{s}$		
$(Nv)_{2T}$	$Z_{2T}^2 x_Q - X_{2T}$	$k\mathbf{m}$		
$(N\ell)_{-T,-T}$	$(D\mu)_T Z_T^2 y_Q + (N\mu)_T Z_T^2 x_Q$ $+2Y_T^2 - (N\mu)_T X_T$			$(3+2k)\mathbf{m}$
$(D\ell)_{-T,-T}$	$(D\mu)_T Z_T^2$			0
f	$\leftarrow f^2 (N\ell)_{T,T}(Dv)_{T,T}$	$k\mathbf{m} + \mathbf{S}_k + \mathbf{M}_k$		
g	$\leftarrow g^2 (D\ell)_{T,T}(Nv)_{T,T}$	$k\mathbf{m} + \mathbf{S}_k + \mathbf{M}_k$		
f	$\leftarrow f^2 (N\ell)_{T,T}$		$\mathbf{S}_k + \mathbf{M}_k$	
g	$\leftarrow g^2 (D\ell)_{T,T}$		$k\mathbf{m} + \mathbf{S}_k$	
f	$\leftarrow f^2 (D\ell)_{-T,-T}$			$k\mathbf{m} + \mathbf{S}_k$
g	$\leftarrow g^2 (N\ell)_{-T,-T}$			$\mathbf{S}_k + \mathbf{M}_k$
TOTAL		$\mathbf{m}_a + 8\mathbf{s}$ $+(5+5k)\mathbf{m}$ $+2\mathbf{S}_k + 2\mathbf{M}_k$	$\mathbf{m}_a + 7\mathbf{s}$ $+(5+3k)\mathbf{m}$ $+2\mathbf{S}_k + \mathbf{M}_k$	$\mathbf{m}_a + 7\mathbf{s}$ $+(5+3k)\mathbf{m}$ $+2\mathbf{S}_k + \mathbf{M}_k$

FIGURE A.3 – Analysis of the cost of generic doubling

The addition step.

Next we deal with addition. When $T \neq \pm P$, the slope of the line joining T and P is

$$\lambda_{T,P} = \frac{y_T - y_P}{x_T - x_P} = \frac{(N\lambda)_{T,P}}{(D\lambda)_{T,P}},$$

where $(N\lambda)_{T,P} = Y_T - y_P Z_T^3$ and $(D\lambda)_{T,P} = X_T Z_T - x_P Z_T^3$.

It follows that the value of $\ell_{T,P}$ at the point Q is given by

$$y_Q - y_P - \lambda(x_Q - x_P) = \frac{(N\ell)_{T,P}}{(D\ell)_{T,P}},$$

where we precompute $y_Q - y_P$ and $x_Q - x_P$, and the numerator and denominator are given by

$$(N\ell)_{T,P} = (D\lambda)_{T,P}(y_Q - y_P) - (N\lambda)_{T,P}(x_Q - x_P)$$

and

$$(D\ell)_{T,P} = (D\lambda)_{T,P}.$$

The coordinates X_{T+P}, Y_{T+P} and Z_{T+P} of $T + P$ are then given by

$$\begin{cases} X_{T+P} = (N\lambda)_{T,P}^2 - (X_T + x_P Z_T^2)(X_T - x_P Z_T^2)^2, \\ Y_{T+P} = -(D\lambda)_{T,P}^3 y_P + (N\lambda)_{T,P}(x_P (D\lambda)_{T,P}^2 - X_{T+P}), \\ Z_{T+P} = (D\lambda)_{T,P}. \end{cases}$$

It follows that we can write the value of v_{T+P} at Q as

$$v_{T+P}(Q) = x_Q - x_{T+P} = \frac{(Nv)_{T+P}}{(Dv)_{T+P}},$$

with $(Nv)_{T+P} = x_Q Z_{T+P}^2 - X_{T+P}$ and $(Dv)_{T+P} = Z_{T+P}^2$.

When the loop beginning with line **2** of Figure A.2 is used, we need to calculate $\ell'_{-T,-P}$ at Q. In fact, using equation (A.3.1), we can write

$$\ell'_{-T,-P}(Q) = \frac{y_Q + y_P}{x_Q - x_P} + \frac{(N\lambda)_{T,P}}{(D\lambda)_{T,P}}.$$

If $\frac{y_Q + y_P}{x_Q - x_P}$ has been precomputed and has value $\alpha_{Q,P}$, we can write

$$\ell'_{-T,-P}(Q) = \frac{(N\ell')_{-T,-P}}{(D\ell')_{-T,-P}},$$

with $(N\ell')_{-T,-P} = (D\lambda)_{T,P}\alpha_{Q,P} + (N\lambda)_{T,P}$ and $(D\ell')_{-T,-P} = (D\lambda)_{T,P}$.

All the operations needed in the addition steps of the algorithms in Figures A.1 and A.2 are shown in detail in Figure A.4. As with the doubling step, the quantities are to be computed in the order shown in the table, a blank entry indicating that the corresponding quantity need not be computed in the case indicated at the top of the corresponding column. The costs of the computations are calculated assuming that the results of intermediate steps are kept in memory. An entry 0 indicated that the quantity has already been calculated at a previous stage.

Quantity	Formula	Classical Miller (Fig. A.1)	Modified Miller (Fig. A.2, loop **2**)	Modified Miller (Fig. A.2, loop **4**)
$(D\lambda)_{T,P}$	$(X_T - x_P Z_T^2)Z_T$	$\mathbf{s} + 2\mathbf{m}$	$\mathbf{s} + 2\mathbf{m}$	$\mathbf{s} + 2\mathbf{m}$
$(N\lambda)_{T,P}$	$Y_T - y_P Z_T^3$	$2\mathbf{m}$	$2\mathbf{m}$	$2\mathbf{m}$
$(N\ell)_{T,P}$	$(D\lambda)_{T,P}(y_Q - y_P)$ $-(N\lambda)_{T,P}(x_Q - x_P)$	$2k\mathbf{m}$		$2k\mathbf{m}$
$(D\ell)_{T,P}$	$(D\lambda)_{T,P}$	0		0
X_{T+P}	$(N\lambda)_{T,P}^2 - X_T(X_T - x_P Z_T^2)^2$ $-x_P Z_T^2(X_T - x_P Z_T^2)^2$	$2\mathbf{s} + 2\mathbf{m}$	$2\mathbf{s} + 2\mathbf{m}$	$2\mathbf{s} + 2\mathbf{m}$
Y_{T+P}	$-y_P Z_T^3(X_T(X_T - x_P Z_T^2)^2$ $-x_P Z_T^2(X_T - x_P Z_T^2)^2)$ $+(N\lambda)_{T,P}(x_P Z_T^2(X_T - x_P Z_T^2)^2$ $-X_{T+P})$	$2\mathbf{m}$	$2\mathbf{m}$	$2\mathbf{m}$
Z_{T+P}	$(D\lambda)_{T,P}$	0	0	0
$(Nv)_{T+P}$	$x_Q Z_{T+P}^2 - X_{T+P}$	$\mathbf{s} + k\mathbf{m}$		
$(Dv)_{T+P}$	Z_{T+P}^2	0		
$(N\ell')_{-T,-P}$	$\alpha_{Q,P}(D\lambda)_{T,P} + (N\lambda)_{T,P}$		$k\mathbf{m}$	
$(D\ell')_{-T,-P}$	$(D\lambda)_{T,P}$		0	
f	$\leftarrow f(N\ell)_{T,P}(Dv)_{T+P}$	$k\mathbf{m} + \mathbf{M}_k$		
g	$\leftarrow g(D\ell)_{T,P}(Nv)_{T+P}$	$k\mathbf{m} + \mathbf{M}_k$		
f	$\leftarrow f(D\ell')_{-T,-P}$		$k\mathbf{m}$	
g	$\leftarrow g(N\ell')_{-T,-P}$		$\mathbf{M}_k$	
f	$\leftarrow f(N\ell)_{T,P}$			$\mathbf{M}_k$
g	$\leftarrow g(D\ell)_{T,P}$			$k\mathbf{m}$
TOTAL		$4\mathbf{s} + (8 + 5k)\mathbf{m}$ $+2\mathbf{M}_k$	$3\mathbf{s} + (8 + 2k)\mathbf{m}$ $+\mathbf{M}_k$	$3\mathbf{s} + (8 + 3k)\mathbf{m}$ $+\mathbf{M}_k$

FIGURE A.4 – Analysis of the cost of generic addition

A.3.3 The main result

The following theorem recapitulates the number of operations in our variant of Miller's algorithm.

Théorème A.3.2. *Suppose E is given in short Weierstrass form $y^2 = x^3 + ax + b$ with coefficients $a, b \in \mathbb{F}$. Let $P \in E(\mathbb{F})$ be a point of odd order $r \geq 2$ and let Q be a point of E of order r with coordinates in an extension field $\mathbb{F}'$ of $\mathbb{F}$ of degree k. We assume P and Q given in Weierstrass coordinates (x_P, y_P) and (x_Q, y_Q).*

1. *Using the associated Jacobian coordinates, the algorithms of Figures A.1 and A.2 can be implemented in such a way that all the denominators $(D\ell)_{T,T}$, $(D\ell)_{T,P}$, $(Dv)_{2T}$, $(Dv)_{T+P}$ and $(D\ell')_{-T,-P}$ belong to $\mathbb{F}$.*

2. *When this is the case :*

   (a) *Each doubling step of the generic usual Miller algorithm takes $\mathbf{m}_a + 8\mathbf{s} + (5 + 5k)\mathbf{m} + 2\mathbf{S}_k + 2\mathbf{M}_k$ operations while in the generic modified Miller algorithm it requires only $\mathbf{m}_a + 7\mathbf{s} + (5 + 3k)\mathbf{m} + 2\mathbf{S}_k + \mathbf{M}_k$ operations.*

   (b) *Each addition step of the generic usual Miller algorithm takes $4\mathbf{s} + (8 + 5k)\mathbf{m} + 2\mathbf{M}_k$ operations. On the other hand, the generic modified Miller algorithm requires only $3\mathbf{s} + (8 + 2k)\mathbf{m} + \mathbf{M}_k$ operations when line **2** is needed and $3\mathbf{s} + (8 + 3k)\mathbf{m} + \mathbf{M}_k$ operations when line **4** is needed.*

We have made no serious attempt to minimize the number of operations, for example by using formulas similar to those in [7].

Since the first part of Theorem A.3.2 implies that, when $\mathbb{F}$ is a finite field with q elements, we have

$$(D\ell)_{T,T}^{q-1} = (D\ell)_{T,P}^{q-1} = (Dv)_{2T}^{q-1} = (Dv)_{T+P}^{q-1} = (D\ell')_{-T,-P}^{q-1} = 1,$$

denominator elimination is possible when we only need to calculate $f_{s,P}(Q)$ to some power divisible by $q-1$. Such an algorithm saves at least $k\mathbf{m}$ operations both in the classical version and our variant.

Our new version of Miller's algorithm works perfectly well for arbitrary embedding degree. For example, using Theorem 2 of [100], it should be possible to find an elliptic curve with a prime embedding degree minimizing the number of iterations. Optimal pairings [183] involve in their computation a product $\prod_{i=0}^{\ell} f_{c_i,Q}^{q^i}(P)$ whose terms can be computed with our algorithm. Note that switching P and Q will lead to more computations in the extension field, but it is shown in [137] that optimized versions of the Ate and the twisted Ate pairing can be computed at least as fast as the Tate pairing. Note that Heß [100] §5, also mentions pairings of potential interest when k is odd and the elliptic curve has discriminant -4 and when k is not divisible by 3 and the elliptic curve has discriminant -3.

A.4 Curves with even embedding degree

Currently, most implementations (where $\mathbb{F}$ is a finite field) are adapted to curves with embedding degree 2^i3^j, since the usual version of Miller's algorithm can be implemented more efficiently. Indeed, such curves admit an even twist which allows denominator elimination [18, 19]. In the case of a cubic twist, denominator elimination is also possible [133]. Another advantage of embedding degrees of the form 2^i3^j is that the corresponding extensions of $\mathbb{F}$ can be written as composite extensions of degree 2 or 3, which allows faster basic arithmetic operations [122].

In what follows, we discuss a version with denominator elimination of our variant adapted to even embedding degrees. Similar ideas have been used before, for example in [133] or [46]. We suppose that $\mathbb{F}$ is a finite field of odd cardinality q which we denote by $\mathbb{F}_q$. We consider an elliptic curve E with short Weierstrass model $y^2 = x^3 + ax + b$ with $a, b \in \mathbb{F}_q$. We suppose that $E(\mathbb{F}_q)$ contains a point P with affine coordinates (x_P, y_P) of order an odd prime r and embedding degree k. If $n \geq 1$ is an integer, we denote by $\mathbb{F}_{q^n}$ the extension of degree n of $\mathbb{F}_q$ in a fixed algebraic closure of $\mathbb{F}_q$.

We suppose for the rest of this section that k is even. Let γ be a non-square element of $\mathbb{F}_{q^{k/2}}$ and fix a square root β in $\mathbb{F}_{q^k}$ of γ, so that every element of $\mathbb{F}_{q^k}$ can be written in a unique way in the form $x + y\beta$ with $x, y \in \mathbb{F}_{q^{k/2}}$. Then one knows that $E[r]$ contains non zero points $Q = (x_Q, y_Q)$ that satisfy $x_Q, y_Q/\beta \in \mathbb{F}_{q^{k/2}}$. In fact, if π denotes the Frobenius endomorphism of E over $\mathbb{F}_q$ given by $\pi(x, y) = (x^q, y^q)$, then π restricts to an endomorphism of $E[r]$ viewed as a vector space over the field with r elements, and a point $Q \in E[r] \setminus \{O_E\}$ has the desired property if and only if Q belongs to the eigenspace V of π with eigenvector q. We can easily construct such

points Q as follows : if $R \in E[r]$, then the point

$$Q := [k](R) - \sum_{i=0}^{k-1} \pi^i(R)$$

lies in V and, when r is large, the probability that it is non zero when R is selected at random is overwhelming.

As is well-known, this leads to speed-ups in the calculation of $f_{s,P}(Q)$ whenever $Q \in V$. For example, the calculation of $(N\ell)_{T,P}$ (see Figure A.4) takes only $(2\frac{k}{2})\mathbf{m} = k\mathbf{m}$ operations. On the other hand, in general neither $\alpha_{Q,P}$ nor $\alpha_{Q,P}/\beta$ belongs to $\mathbb{F}_{q^{k/2}}$, so that the calculation of $(N\ell')_{-T,-P}$ also requires $k\mathbf{m}$ operations. So, the improvement obtained in the generic case is lost.

When $P \in E[r](\mathbb{F}_q)$ and $Q \in V$, it is well-known that the Tate pairing $e_r^T(P,Q)$ is given by $e_r^T(P,Q) = f_{r,P}(Q)^{\frac{q^k-1}{r}}$. Denominator elimination is possible since $q^{k/2} - 1$ divides $\frac{q^k-1}{r}$, as follows from the fact that r is prime and the definition of the embedding degree k.

Let $v = x + y\beta$ with $x, y \in \mathbb{F}_{q^{k/2}}$ be an element of $\mathbb{F}_{q^k}$. The conjugate of v over $\mathbb{F}_{q^{k/2}}$ is then $\bar{v} = x - y\beta$. It follows that, if $v \neq 0$, then

$$\frac{1}{v} = \frac{\bar{v}}{x^2 - \gamma y^2} \tag{A.4.1}$$

where $x^2 - \gamma y^2 \in \mathbb{F}_{q^{k/2}}$. Thus, in a situation where elements of $\mathbb{F}_{q^{k/2}}$ can be ignored, $\frac{1}{v}$ can be replaced by $\bar{v}$, thereby saving an inversion in $\mathbb{F}_{q^k}$.

We exploit all this in the algorithm in Figure A.5, where we use the same notation as in Figure A.2. It outputs an element f of $\mathbb{F}_{q^k}$ such that $f^{q^{\frac{k}{2}}-1} = f_{s,P}(Q)^{q^{\frac{k}{2}}-1}$. Thus, when $s = r$, we find, since $q^{\frac{k}{2}} - 1$ divides $\frac{q^k-1}{r}$, that $e_r^T(P,Q) = f^{\frac{q^k-1}{r}}$.

We replace the denominators $\ell_{-T,-T}$ and $\ell_{-T,-P}$ (updated in the function g) by their conjugates $\overline{\ell_{-T,-T}}$ and $\overline{\ell_{-T,-P}}$. In Jacobian coordinates, one has

$$\overline{(N\ell')_{-T,-P}} = \overline{\alpha_{Q,P}}(D\lambda)_{T,P} + (N\lambda)_{T,P}, \qquad \text{and} \tag{A.4.2}$$

$$\overline{(N\ell)_{-T,-T}} = 2Y_T(-y_Q Z_T^3 + Y_T) + (N\mu)_T(x_Q Z_T^2 - X_T). \tag{A.4.3}$$

Cost analysis of the doubling and addition steps in this algorithm can be found in Figures A.6 and A.7. We summarize our conclusions in the following result.

Théorème A.4.1. *Let β be a non-square element of $\mathbb{F}_{q^k}$ whose square lies in $\mathbb{F}_{q^{k/2}}$. Suppose E is given in short Weierstrass form $y^2 = x^3 + ax + b$ with coefficients $a, b \in \mathbb{F}_q$. Let $P \in E(\mathbb{F}_q)$ be a point of odd prime order r with even embedding degree k. Let Q be a point of E of order r with coordinates in the extension field $\mathbb{F}_{q^k}$ of $\mathbb{F}_q$ of degree k. We assume P and Q given in Weierstrass coordinates (x_P, y_P) and (x_Q, y_Q) with x_Q, $y_Q/\beta \in \mathbb{F}_{q^{k/2}}$. Using the associated Jacobian coordinates, every doubling step of the algorithm of Figure A.5 requires $\mathbf{m}_a + 7\mathbf{s} + (5+k)\mathbf{m} + \mathbf{S}_k + \mathbf{M}_k$ operations and every addition step requires $3\mathbf{s} + (8+k)\mathbf{m} + \mathbf{M}_k$.*

As before, we have made no serious attempt to minimize the number of operations, for example by using formulas similar to those in [7]. Moreover, we believe that there is scope for further improvement in the case of curves in special families or curves with efficient arithmetic (see for example [76]).

Algorithme 6 : Miller(P, Q, s) modified with even embedding degree

Data : $s = \sum_{i=0}^{l-1} s_i 2^i$, $s_i \in \{0,1\}$, $s_{l-1} = 1$, h Hamming weight of s, $Q \in E[r]$ a non-zero element with x_Q, $y_Q/\beta \in \mathbb{F}_{q^{k/2}}$.

Result : An element f of $\mathbb{F}_{q^k}$ satisfying $f^{q^{k/2}-1} = f_{s,P}(Q)^{q^{k/2}-1}$

1 $f \leftarrow 1$, $T \leftarrow P$, ;
2 **if** $l + h$ is odd **then**
3 $\quad \delta \leftarrow 1$;
4 **else**
5 $\quad \delta \leftarrow 0$;
6 **for** $i = l - 2$ **to** 0 **do**
8 $\quad$ **if** $\delta = 0$ **then**
9 $\quad\quad f \leftarrow f^2(N\ell)_{T,T}$, $T \leftarrow 2T$, $\delta \leftarrow 1$;
11 $\quad\quad$ **if** $s_i = 1$ **then**
12 $\quad\quad\quad f \leftarrow f\overline{(N\ell')_{-T,-P}}$, $T \leftarrow T + P$, $\delta \leftarrow 0$;
14 $\quad$ **else**
15 $\quad\quad f \leftarrow f^2\overline{(N\ell)_{-T,-T}}$, $T \leftarrow 2T$, $\delta \leftarrow 0$;
17 $\quad\quad$ **if** $s_i = 1$ **then**
18 $\quad\quad\quad f \leftarrow f(N\ell)_{T,P}$, $T \leftarrow T + P$, $\delta \leftarrow 1$;
19 **return** f

FIGURE A.5 – The modified Miller algorithm for even embedding degree

Quantity	Formula	Modified Miller (Fig. A.5, loop **1**)	Modified Miller (Fig. A.5, loop **3**)
$(N\mu)_T$	$3X_T^2 + aZ_T^4$	$\mathbf{m}_a + 3\mathbf{s}$	$\mathbf{m}_a + 3\mathbf{s}$
$(D\mu)_T$	$2Y_T Z_T = (Y_T + Z_T)^2 - Y_T^2 - Z_T^2$	$2\mathbf{s}$	$2\mathbf{s}$
$(N\ell)_{T,T}$	$(D\mu)_T Z_T^2 y_Q - (N\mu)_T Z_T^2 x_Q - 2Y_T^2 - (N\mu)_T X_T$	$(3 + 2(k/2))\mathbf{m}$	
X_{2T}	$(N\mu)_T^2 - 8X_T Y_T^2$	$\mathbf{s} + \mathbf{m}$	$\mathbf{s} + \mathbf{m}$
Y_{2T}	$(N\mu)_T(4X_T Y_T^2 - X_{2T}) - 8Y_T^4$	$\mathbf{s} + \mathbf{m}$	$\mathbf{s} + \mathbf{m}$
Z_{2T}	$(D\mu)_T$	0	0
$(N\ell)_{-T,-T}$	$(D\mu)_T Z_T^2 y_Q + (N\mu)_T Z_T^2 x_Q + 2Y_T^2 - (N\mu)_T X_T$		$(3 + 2(k/2))\mathbf{m}$
f	$\leftarrow f^2(N\ell)_{T,T}$	$\mathbf{S}_k + \mathbf{M}_k$	
f	$\leftarrow f^2\overline{(N\ell)_{-T,-T}}$		$\mathbf{S}_k + \mathbf{M}_k$
TOTAL		$\mathbf{m}_a + 7\mathbf{s}$ $+(5+k)\mathbf{m}$ $+\mathbf{S}_k + \mathbf{M}_k$	$\mathbf{m}_a + 7\mathbf{s}$ $+(5+k)\mathbf{m}$ $+\mathbf{S}_k + \mathbf{M}_k$

FIGURE A.6 – Analysis of doubling for even embedding degree

Quantity	Formula	Modified Miller (Fig. A.5, loop **2**)	Modified Miller (Fig. A.5, loop **4**)
$(D\lambda)_{T,P}$	$(X_T - x_P Z_T^2)Z_T$	$\mathbf{s} + 2\mathbf{m}$	$\mathbf{s} + 2\mathbf{m}$
$(N\lambda)_{T,P}$	$Y_T - y_P Z_T^3$	$2\mathbf{m}$	$2\mathbf{m}$
$(N\ell)_{T,P}$	$(D\lambda)_{T,P}(y_Q - y_P) - (N\lambda)_{T,P}(x_Q - x_P)$		$2(k/2)\mathbf{m}$
X_{T+P}	$(N\lambda)_{T,P}^2 - X_T(X_T - x_P Z_T^2)^2$ $-x_P Z_T^2(X_T - x_P Z_T^2)^2$	$2\mathbf{s} + 2\mathbf{m}$	$2\mathbf{s} + 2\mathbf{m}$
Y_{T+P}	$-y_P Z_T^3(X_T(X_T - x_P Z_T^2)^2$ $-x_P Z_T^2(X_T - x_P Z_T^2)^2)$ $+(N\lambda)_{T,P}(x_P Z_T^2(X_T - x_P Z_T^2)^2$ $-X_{T+P})$	$2\mathbf{m}$	$2\mathbf{m}$
Z_{T+P}	$(D\lambda)_{T,P}$	0	0
$(N\ell')_{-T,-P}$	$\alpha_{Q,P}(D\lambda)_{T,P} + (N\lambda)_{T,P}$	$k\mathbf{m}$	
f	$\leftarrow f_{(N\ell')_{-T,-P}}$	$\mathbf{M}_k$	
f	$\leftarrow f_{(N\ell)_{T,P}}$		$\mathbf{M}_k$
TOTAL		$3\mathbf{s} + (8+k)\mathbf{m}$ $+\mathbf{M}_k$	$3\mathbf{s} + (8+k)\mathbf{m}$ $+\mathbf{M}_k$

FIGURE A.7 – Cost analysis of addition for even embedding degree

A.5 Experiments

We ran some experiments comparing usual Miller (Figure A.1) with the variant of Figure A.2 when $k = 17$ and $k = 19$. When $k = 18$, we compared the performance of the algorithms of Figures A.1 and A.5 and also the algorithm proposed in 2003 by [18]. In each case, the group order r has 192 bits and the rho-value $\rho = \frac{\log q}{\log r}$ is a little under 1.95, q being the cardinality of the base field. In the example with $k = 17$, the big field $\mathbb{F}_{q^k}$ was generated as $\mathbb{F}_q[x]/(x^{17} + x + 12)$ while when $k = 18$ and $k = 19$, $\mathbb{F}_{q^k}$ was generated as $\mathbb{F}_q[x]/(x^k + 2)$. Our curves were constructed using the Cocks-Pinch method (see [82]).

- For $k = 17$, the curve is $E : y^2 = x^3 + 6$ and

$$\begin{aligned} r &= 627710173538668076383578942320766641610235544446403993985 7 \\ q &= 220522060437535156222191257592633526736200793713321924733 \\ &\quad 078476278317592333015347957276809006053649122618843827 71 \end{aligned}$$

- For $k = 18$, the curve is $E : y^2 = x^3 + 3$ and

$$\begin{aligned} r &= 627710173538668076383578942320766641610235544446404691873 9 \\ q &= 352868453926302204292551351775152292482424484549774231757 \\ &\quad 096903373137251646468704115267717074091166525440296813 89 \end{aligned}$$

- For $k = 19$, the curve is $E : y^2 = x^3 + 2$ and

$$\begin{aligned} r &= 627710173538668076383578942320766641610235544446403823192 7 \\ q &= 328331730495825080256136908360300125975534969742697656797 5 \\ &\quad 265167703768467341628884414224762338965233382661234563 7 \end{aligned}$$

For the computations, we used the NTL library [174] and implemented the algorithms without any optimization on an Intel(R) Core(TM)2 Duo CPU E8500 @ 3.16Ghz using Ubuntu Operating System 9.04. The computations of the Miller function (without any final exponentiation) were executed on 100 random inputs. The experimental average results are summarized in Figure A.8.

k	Usual Miller (Fig. A.1)	Our variant (Fig. A.2)	Our variant with k even (Fig. A.5)	Miller without denominators [18]
17	$0.0664s$	$0.0499s$	-	-
18	$0.0709s$	-	$0.0392s$	$0.0393s$
19	$0.0769s$	$0.0683s$	-	-

FIGURE A.8 – Timings

A.6 Conclusion

In this paper we presented a variant of Miller's formula and algorithm. Generically, it is more efficient than the usual Miller algorithm as in Figure A.1, calculation suggest that it can lead to a real improvement in cases where denominator elimination is not available. Consequently, we believe it will have applications in pairing-based cryptography using elliptic curves with embedding degree not being on the form 2^i3^j, for example when the optimal Ate or Twisted Ate pairing is used. Further work is needed to clarify such questions.

Acknowledgments

This work was supported by Project ANR-07-TCOM-013-04 PACE financed by the Agence National de Recherche (France). We would like to thank Andreas Enge for several helpful discussions and the anonymous reviewers for their numerous suggestions and remarks which have enables us to substantially improve the paper.

Annexe B

Choosing and generating parameters for pairing implementation on BN curves

AAECC 2017 [54]
avec Sylvain Duquesne , Safia Haloui et Franck Rondepierre[1]

Abstract : Because pairings have many applications, many hardware and software pairing implementations can be found in the literature. However, the parameters generally used have been invalidated by the recent results on the discrete logarithm problem over pairing friendly elliptic curves [118]. New parameters must be generated to insure enough security in pairing based protocols. More generally it could be useful to generate nice pairing parameters in many real-world applications (specific security level, resistance to specific attacks on a protocol, database of curves). The main purpose of this paper is to describe explicitly and exhaustively what should be done to generate the best possible parameters and to make the best choices depending on the implementation context (in terms of pairing algorithm, ways to build the tower field, $\mathbb{F}_{p^{12}}$ arithmetic, groups involved and their generators, system of coordinates).
We focus on low level implementations, assuming that $\mathbb{F}_p$ additions have a significant cost compared to other $\mathbb{F}_p$ operations. However, our results are still valid if $\mathbb{F}_p$ additions can be neglected. We also explain why the best choice for the polynomials defining the tower field $\mathbb{F}_{p^{12}}$ is only dependent on the value of the BN parameter u mod small integers (like 12 for instance) as a nice application of old elementary arithmetic results. This should allow a faster generation of this parameter. Moreover, we use this opportunity to give some new slight improvements on $\mathbb{F}_{p^{12}}$ arithmetic (in a pairing context).

[1]This work was supported in part by French projects ANR-12-BS01-0010-01 "PEACE", ANR-12-INSE-0014 "SIMPATIC" and by the LIRIMA MACISA project.

B.1 Introduction

Pairing based cryptography has now many practical applications such as identity based cryptography [35] or broadcast encryption [37]. Because of recent attacks on the discrete logarithm problem in small characteristic finite fields [114, 13], it is now clear that prime base fields should be used to define pairing friendly elliptic curves. Many implementations, both hardware and software can be found in the literature and some pairing friendly parameters are given. For example, at the 128-bit security level, the parameters given in [155] are almost always used because they have many good properties, in particular in terms of efficiency. However, these parameters have been seriously weakened by the recent results on the middle prime case of the number field sieve [118]. This implies that new (and larger) parameters are necessary for real-world deployment of pairing-based cryptography. Pairing parameter generation is then a crucial matter today. Moreover, it is also a long-term concern. Indeed, depending on the situation, it could be useful to generate alternative nice parameters (e.g. resistance to subgroup attacks [132, 15], larger (or smaller) security levels, database of pairing friendly curves, sovereignty considerations). This paper deals only with the case of BN curves because they are the most popular today but also because they have many nice properties (e.g. maximal order twists). Of course, other pairing friendly families might have an interest for the next generation of curves for example for high security level. But many parts of this paper can be reused especially in case of degree 6 twists. For example BLS12 curves will use the same extension fields than BN curves and KSS18 curves will use almost the same (only the first level of the tower field will change).
The main purpose of this paper is to describe explicitly and exhaustively what should be done to generate the best possible parameters and to make the best choices depending on the implementation context. We focus on low level implementation (mainly hardware but also assembly language), assuming that $\mathbb{F}_p$ additions have a cost that remains significant compared to other $\mathbb{F}_p$ operations, whereas they are usually neglected in the literature. Indeed, in such contexts, the ratio between an addition and a multiplication can be as large as 0.2 to 0.5. However, the results obtained are still valid in the case where $\mathbb{F}_p$ additions can be neglected. Finally, one has to keep in mind that subtleties in the underlying architecture of the target device can blur the operation counts made in this work. We present a state of the art of possible optimizations of pairing computations. We recall some that already lie in the literature and propose new ones. For instance, we give some new ideas to minimize the number of $\mathbb{F}_p$ additions during the pairing computation. We also explain why the best choice for the polynomials defining the tower field $\mathbb{F}_{p^{12}}$ is only dependent on the value of the BN parameter u mod small integers (like 12).
The paper is organized as follows. In Section B.2 we recall how the optimal Ate pairing on BN curves is computed. In Section B.3 we present the different options to build $\mathbb{F}_{p^{12}}$ and how to implement the basic operations. This includes the choice of the defining polynomials in terms of u, the tower structure and the algorithms for basic arithmetic, depending on the relative cost of $\mathbb{F}_p$ operations. In Section B.4 we explain how to choose the BN parameter u. Sections B.5 and B.6 are devoted to the choices inherent to the curve (coefficients, generators, system of coordinates).

Finally, in Section B.7 we recall the other algorithms that should be used for efficient implementation and adapt them to our context and to the results obtained in the previous sections.
We will use the following notations for $\mathbb{F}_{p^i}$ arithmetic

- $\mathbf{A}_i$ denotes an addition,
- $\mathbf{A}'_i$ denotes a multiplication by 2,
- $\mathbf{M}_i$ denotes a multiplication ($\mathbf{M}_i^M$ if the method M is used),
- $\mathbf{sM}_i$ denotes a sparse multiplication,
- $\mathbf{m}_{i,c}$ denotes a multiplication by a constant c,
- $\mathbf{S}_i$ denotes a squaring ($\mathbf{S}_i^M$ if the method M is used),
- $\mathbf{I}_i$ denotes an inversion.

B.2 Background

B.2.1 Barreto-Naehrig (BN) curves

A BN curve [20] is an elliptic curve E defined over a finite field $\mathbb{F}_p$, $p \geq 5$, such that its order r and p are prime numbers given by

$$\begin{aligned} p &= 36u^4 + 36u^3 + 24u^2 + 6u + 1, \\ r &= 36u^4 + 36u^3 + 18u^2 + 6u + 1, \end{aligned}$$

for some u in $\mathbb{Z}$. It has an equation of the form $y^2 = x^3 + b$, where $b \in \mathbb{F}_p^*$ and its neutral element is denoted by O_E.
BN curves have an embedding degree equal to 12 which made them particularly appropriate for the 128-bit security level. Indeed, a 256-bit prime p leads to a 256-bit curve and to pairings taking values in $\mathbb{F}_{p^{12}}^*$, which is a 3072-bit multiplicative group. Both groups involved are then supposed to match the 128-bit security level according to the NIST recommendations [156] (which are however now invalidate by [118]). By the way, BN curves have been the object of numerous recent publications ([53, 3, 41, 87, 154, 92, 180]).
Finally, BN curves always have order 6 twists. If ξ is an element which is neither a square nor a cube in $\mathbb{F}_{p^2}$, the twisted curve E' of E is defined over $\mathbb{F}_{p^2}$ by the equation $y^2 = x^3 + b'$ with $b' = b/\xi$ or $b' = b\xi$. In order to simplify the computations, the element ξ should also be used to represent $\mathbb{F}_{p^{12}}$ as a degree 6 extension of $\mathbb{F}_{p^2}$ ($\mathbb{F}_{p^{12}} = \mathbb{F}_{p^2}[\gamma]$ with $\gamma^6 = \xi$) [53], [131]. In this paper, we deal only with the case $b' = b/\xi$ as usually done in the literature but $b' = b/\xi^5$ can be also used with a very small additional cost [92].

B.2.2 Optimal Ate pairing

Let a be an integer and $Q \neq O_E$ be a point on E. We denote by $f_{a,Q}$ the normalized function on the curve with divisor $(f_{a,Q}) = a[Q] - [aQ] - (a-1)[O_E]$. Such functions are the core of all known pairings. For example, the (reduced) Tate pairing is defined by

$$e_T(P,Q) = f_{r,P}(Q)^{\frac{p^{12}-1}{r}}.$$

There are many variants of the Tate pairing using a smaller value of a in order to shorten the length of the Miller loop ([16, 101, 100, 137, 130, 183]). It has been proven in [183] that the shortest possible loop has length $\log(r)/\varphi(12) = \log(r)/4$ which is reached by the so-called optimal Ate pairing.
Let $\pi(x,y) = (x^p, y^p)$ be the Frobenius map on the curve. If P is a rational point on E and Q is a point in $E\left(\mathbb{F}_{p^{12}}\right)$ which is in the p-eigenspace of π, the optimal Ate pairing of P and Q [154] can be defined by

$$\left(f_{v,Q}(P).\ell_{vQ,\pi(Q)}(P).\ell_{vQ+\pi(Q),-\pi^2(Q)}(P)\right)^{\frac{p^{12}-1}{r}},$$

where $v = 6u+2$ and $\ell_{A,B}$ is the normalized line function arising in the sum of the points A and B. Its computation is made of four steps :

1. A Miller loop to compute $f_{|v|,Q}(P)$. The algorithmic choices for this step are discussed in Section B.2.3.

2. If $v < 0$, the result f of the Miller loop must be inverted to recover $f_{v,Q}(P)$. Such an inversion is potentially expensive but thanks to the final exponentiation, f^{-1} can be replaced by f^{p^6} [3] which is nothing but the conjugation in $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$, thus it is almost for free.

3. Two line computations, namely $\ell_{vQ,\pi(Q)}(P)$ and $\ell_{vQ+\pi(Q),-\pi^2(Q)}(P)$ which are nothing but extra addition steps of the Miller loop.

4. A final exponentiation to the power of $\frac{p^{12}-1}{r}$. The algorithmic choices for this step are discussed in Section B.2.4.

Since BN curves have twists of order 6, the point Q can in fact be chosen of the form $(x_Q\gamma^2, y_Q\gamma^3) \in E\left(\mathbb{F}_{p^{12}}\right)$, where $x_Q, y_Q \in \mathbb{F}_{p^2}$. This means that elliptic curve operations lie in $\mathbb{F}_{p^2}$ instead of $\mathbb{F}_{p^{12}}$.

B.2.3 The Miller loop

The first step of the pairing computation evaluates $f_{|v|,Q}(P)$ using the Miller algorithm [140]. It is based on the double and add scheme used for the computation of $|v|Q$ by evaluating at P the lines occurring in the doubling and addition steps of this computation. More precisely, it is based on Miller's formula :

Lemme B.2.1. *For all $a, b \in \mathbb{Z}$, Q a point of E, we have*

$$f_{a+b,Q} = f_{a,Q} f_{b,Q} \frac{\ell_{aQ,bQ}}{v_{(a+b)Q}}$$

where $\ell_{aQ,bQ}$ is the line passing through the points aQ and bQ and $v_{(a+b)Q}$ is the vertical line passing through the point $(a+b)Q$.

Remark 1. *In the case of the optimal Ate pairing, the point Q comes from the twisted curve. Hence its x-coordinate lies in a proper subfield of $\mathbb{F}_{p^{12}}$ and $P \in E(\mathbb{F}_p)$ so that $v_{(a+b)Q}(P)$ also lies in a proper subfield of $\mathbb{F}_{p^{12}}$ and then is wiped out by the final exponentiation. This is known as the denominator elimination [17, 122]. Then we do not consider $v_{(a+b)Q}$ in the following.*

Miller's algorithm makes use of Lemma B.2.1 with $b = a$ (for doubling steps) or $b = 1$ (for addition steps). It is described in Algorithm 17 assuming Remark 1.

Algorithme 7 : Miller(P, Q, a)

Data : $a = (a_n \dots a_0)_2$,
$P \in E(\mathbb{F}_p)$,
$Q \in E\left(\mathbb{F}_{p^{12}}\right)$ having its x-coordinate in a proper subfield of $\mathbb{F}_{p^{12}}$;
Result : $\lambda f_{a,Q}(P) \in \mathbb{F}^*_{p^{12}}$ with λ in a proper subfield of $\mathbb{F}_{p^{12}}$;

1 $T \leftarrow Q$;
2 $f \leftarrow 1$;
3 **for** $i = n-1$ **to** 0 **do**
4 $\quad f \longleftarrow f^2 \times \ell_{T,T}(P)$;
5 $\quad T \leftarrow 2T$;
6 $\quad$ **if** $a_i = 1$ **then**
7 $\quad\quad f \longleftarrow f \times \ell_{T,Q}(P)$;
8 $\quad\quad T \leftarrow T + Q$;
9 **return** f

Since the Miller algorithm is based on the double and add algorithm, it is natural to try to improve it by using advanced exponentiation techniques like the sliding window method [44, Algo 9.10] or the NAF representation [44, Algo 9.14]. However, the interest is limited in practice for two reasons :

- In the context of pairing based cryptography, the exponent is not a secret. Then it is usually chosen sparse so these methods are useless.
- They involve curve operations like $T + 3Q$ and then field operations like $f \times f_{3,Q} \times \ell_{T,3Q}$. Even if $f_{3,Q}$ is precomputed, an additional $\mathbb{F}_{p^{12}}$ multiplication is required and it is the time consuming operation in Algorithm 17.

The only interesting case is a signed binary representation of the exponent (*i.e.* a 2-NAF) because it can help to find a sparse exponent.The subtraction step of Algorithm 17 should then involve an additional division by the vertical line passing trough Q and $-Q$ but it is wiped out by the final exponentiation.

B.2.4 The final exponentiation

As proposed in [122, 171] the cost of the final exponentiation can be reduced thanks to the integer factorization

$$\frac{p^{12}-1}{r} = \left(p^6-1\right)\left(p^2+1\right)\left(\frac{p^4-p^2+1}{r}\right).$$

Since p is the characteristic of $\mathbb{F}_{p^{12}}$, it is easy to compute the p^{th}–power of any element in $\mathbb{F}_{p^{12}}$ (see Section B.7.1 for details on Frobenius mapping). Powering to the $\left(p^6-1\right)\left(p^2+1\right)$ is then called the easy part of the final exponentiation, even if an expensive inversion in $\mathbb{F}_{p^{12}}$ is required.
Powering to the $\frac{p^4-p^2+1}{r}$ is called the hard part and the exponent is usually written in base p to take advantage of Frobenius mapping ([170, 53, 171, 40, 59]).
Moreover, as f has been raised to the power of $\left(p^6-1\right)\left(p^2+1\right)$, it has order dividing p^4-p^2+1. Then, as noticed in [91], it lies in the cyclotomic group $G_{\Phi_6}(\mathbb{F}_{p^2})$. As a consequence, squaring can be very efficiently computed [91, 116] (more details are given in Section B.7.2) and an inversion is nothing but the conjugation in $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$, thus it is almost free [170, 177].

The most popular way to perform this hard part uses the addition chain [171] :

$$f^{\frac{p^4-p^2+1}{r}} = y_0 y_1^2 y_2^6 y_3^{12} y_4^{18} y_5^{30} y_6^{36},$$

$$\text{where} \quad y_0 = f^p f^{p^2} f^{p^3}, \quad y_1 = \frac{1}{f}, \quad y_2 = \left(f^{u^2}\right)^{p^2}, \quad y_3 = \left(f^u\right)^p,$$
$$y_4 = \frac{\left(f^{u^2}\right)^p}{f^u}, \quad y_5 = \frac{1}{f^{u^2}}, \quad y_6 = \frac{\left(f^{u^3}\right)^p}{f^{u^3}}.$$

The cost of this method is $13\mathbf{M}_{12}, 4\mathbf{S}_{12}$ and 7 Frobenius maps in addition to the 3 exponentiations by u. The method given in [40] is slightly better but computes a power of the optimal Ate pairing. The main drawback of these methods is that they are memory consuming (up to 4Ko), which can be annoying in restricted environments. Some variants optimized in terms of memory consumption are given in [59].

B.3 Choosing the finite fields and their arithmetic

B.3.1 $\mathbb{F}_p$ arithmetic

In this paper, we assume that the basic operations in $\mathbb{F}_p$ (additions, subtractions, multiplications) are already implemented. There are usually performed by combining the schoolbook method or the Karatsuba method [120] and the Montgomery reduction [147] or the Barrett reduction [21].
Anyway, the method used for $\mathbb{F}_p$ arithmetic does not have direct consequences on the choices to be made for extension field or elliptic curve arithmetic. The only important criterion is the relative cost between the $\mathbb{F}_p$ operations ($\mathbf{A}_1$, $\mathbf{A}'_1$, $\mathbf{M}_1$, $\mathbf{S}_1$, $\mathbf{I}_1$).

One of the main purpose of this paper is to discuss the available choices depending on these ratios. We plan to cover most of the possible practical situations, so that we make the following assumptions :

- Addition can be relatively expensive ($\mathbf{A}_1 < 0.5\mathbf{M}_1$). They are often neglected in theoretical studies but this should not be the case for real life implementations, especially in low level implementation where the most common ratios at the 128-bit security level are between 0.2 and 0.3 [166, 89], but also in software implementation (for example, the ratio is 0.17 in the Microsoft ECC library [164] at the 128-bit security level). Therefore, the number of $\mathbb{F}_p$ additions involved in extension field arithmetic must be taken into account and may have an influence on the choices to be made.
- To stay in the most general case, we chose to use distinct symbols to denote an addition ($\mathbf{A}_1$) and a doubling ($\mathbf{A}'_1$). However, we usually have $\mathbf{A}'_1 = \mathbf{A}_1$.
- If a specific algorithm for squaring is implemented, then $\mathbf{S}_1$ is usually assumed to be $0.8\mathbf{M}_1$, otherwise, we have $\mathbf{S}_1 = \mathbf{M}_1$.

B.3.2 Arithmetic of $\mathbb{F}_{p^{2i}}/\mathbb{F}_{p^i}$

In theory, any irreducible quadratic polynomial can be used to build $\mathbb{F}_{p^{2i}}$ over $\mathbb{F}_{p^i}$, but sparse polynomials give better results. Then, $\mathbb{F}_{p^{2i}}$ is usually built as

$$\mathbb{F}_{p^{2i}} = \mathbb{F}_{p^i}[\alpha] \text{ with } \alpha^2 = \mu.$$

Whatever the choice for building $\mathbb{F}_{p^{2i}}$, an addition (resp. a multiplication by 2) requires 2 $\mathbb{F}_{p^i}$ additions (resp. multiplications by 2), so $\mathbf{A}_{2i} = 2\mathbf{A}_i$, $\mathbf{A}'_{2i} = 2\mathbf{A}'_i$.

$\mathbb{F}_{p^{2i}}$ multiplication

We will consider only two methods since the other ones do not reduce the overall complexity, independently of the relative cost of $\mathbb{F}_{p^i}$ operations. The first one is the schoolbook method which computes

$$(x_0 + x_1\alpha)(y_0 + y_1\alpha) = x_0y_0 + \mu x_1y_1 + (x_0y_1 + x_1y_0)\alpha$$

and requires $\mathbf{M}^{SB}_{2i} = 4\mathbf{M}_i + \mathbf{m}_{i,\mu} + 2\mathbf{A}_i$. The second one, the Karatsuba method, is a standard variant of the schoolbook one requiring $\mathbf{M}^{K}_{2i} = 3\mathbf{M}_i + \mathbf{m}_{i,\mu} + 5\mathbf{A}_i$.

$$(x_0 + x_1\alpha)(y_0 + y_1\alpha) = x_0y_0 + \mu x_1y_1 + ((x_0 + x_1)(y_0 + y_1) - x_0y_0 - x_1y_1)\alpha$$

Remark 2. *The schoolbook method should be preferred when* $3\mathbf{A}_i > \mathbf{M}_i$.

$\mathbb{F}_{p^{2i}}$ squaring

The schoolbook method computes a $\mathbb{F}_{p^{2i}}$ squaring via

$$(x_0 + x_1\alpha)^2 = x_0^2 + \mu x_1^2 + 2x_0x_1\alpha$$

while the Karatsuba method replaces $2x_0x_1$ by $(x_0+x_1)^2 - x_0^2 - x_1^2$. Then

$$\mathbf{S}_{2i}^{SB} = \mathbf{M}_i + 2\mathbf{S}_i + \mathbf{m}_{i,\mu} + \mathbf{A}_i + \mathbf{A}'_i \text{ and } \mathbf{S}_{2i}^{K} = 3\mathbf{S}_i + \mathbf{m}_{i,\mu} + 4\mathbf{A}_i.$$

For this squaring operation, we can alternatively use the complex method that is particularly efficient if $\mu = -1$, which explains its name. We have

$$(x_0 + x_1\alpha)^2 = (x_0 + \mu x_1)(x_0 + x_1) - (\mu + 1)x_0x_1 + 2x_0x_1\alpha$$

This method then requires $\mathbf{S}_{2i}^{c} = 2\mathbf{M}_i + \mathbf{m}_{i,\mu} + \mathbf{m}_{i,\mu+1} + 3\mathbf{A}_i + \mathbf{A}'_i$.

Remark 3. *Determining which method is the best is not as easy as for multiplication because it depends on both the relative cost of* $\mathbf{M}_i$ *and* $\mathbf{A}_i$ *and of* $\mathbf{M}_i$ *and* $\mathbf{S}_i$*. We postpone this question to next sections.*

$\mathbb{F}_{p^{2i}}$ inversion

The $\mathbb{F}_{p^{2i}}$ inversion is classically done thanks to the norm of an element

$$N = (x_0 + x_1\alpha)(x_0 - x_1\alpha) = x_0^2 - \mu x_1^2 \in \mathbb{F}_p.$$

We then easily get the inverse of $x_0 + x_1\alpha$ as $\frac{x_0}{N} - \frac{x_1}{N}\alpha$. This way of computing an inverse in $\mathbb{F}_{p^{2i}}$ requires $\mathbf{I}_{2i} = \mathbf{I}_i + 2\mathbf{M}_i + 2\mathbf{S}_i + \mathbf{A}_i + \mathbf{m}_{i,\mu}$.

B.3.3 Arithmetic of $\mathbb{F}_{p^{3i}}/\mathbb{F}_{p^i}$

As in Section B.3.2, a sparse polynomial is usually chosen to minimize the cost of $\mathbb{F}_{p^{3i}}$ arithmetic. Thus, $\mathbb{F}_{p^{3i}}$ is built as $\mathbb{F}_{p^i}[\alpha]$, where $\alpha^3 = \xi \in \mathbb{F}_{p^i}$. Of course, $\mathbb{F}_{p^{3i}}$ arithmetic will involve multiplications by ξ, so ξ must be chosen carefully.
Of course, we trivially have $\mathbf{A}_{3i} = 3\mathbf{A}_i$ and $\mathbf{A}'_{3i} = 3\mathbf{A}'_i$.

$\mathbb{F}_{p^{3i}}$ multiplication

Again, we give only the schoolbook and Karatsuba methods because other ones (like Toom-Cook) require many additions (and divisions by 2 or 3), which is contradictory with our assumptions (additions are not so negligible).

The schoolbook method computes $(x_0 + x_1\alpha + x_2\alpha^2)(y_0 + y_1\alpha + y_0\alpha^2)$ as

$$x_0y_0 + \xi(\mathbf{x_1y_2} + \mathbf{x_2y_1}) + [\mathbf{x_0y_1} + \mathbf{x_1y_0} + \xi x_2y_2]\alpha + [\mathbf{x_0y_2} + \mathbf{y_2x_0} + x_1y_1]\alpha^2 \quad \text{(B.3.1)}$$

which requires $\mathbf{M}_{3i}^{SB} = 9\mathbf{M}_i + 2\mathbf{m}_{i,\xi} + 6\mathbf{A}_i$.

As in the case of quadratic extensions, the Karatsuba method allows to compute the 3 bolded terms of (B.3.1) with only one multiplication (and 3 extra additions). This means that $2\mathbf{M}_i + \mathbf{A}_i$ is replaced by $\mathbf{M}_i + 4\mathbf{A}_i$ three times. As in the case of quadratic extensions, the Karatsuba method then becomes interesting when $\mathbf{M}_i \geq 3\mathbf{A}_i$ and requires $\mathbf{M}_{3i}^{K} = 6\mathbf{M}_i + 15\mathbf{A}_i + 2\mathbf{m}_{i,\xi}$.

We will also use in the following the Karatsuba method when one of the operands is sparse : if one coefficient, say y_2, is zero, computing $(x_0 + x_1\alpha + x_2\alpha^2)(y_0 + y_1\alpha)$ requires $\mathbf{sM}_{3i}^K = 5\mathbf{M}_i + 6\mathbf{A}_i + \mathbf{m}_{i,\xi}$ using the formula

$$x_0y_0 + \xi x_2y_1 + \left[(x_0 + x_1)(y_0 + y_1) - x_0y_0 - x_1y_1\right]\alpha + \left[x_2y_0 + x_1y_1\right]\alpha^2$$

Squaring

The Schoolbook squaring is deduced from the multiplication.

$$(x_0 + x_1\alpha + x_2\alpha^2)^2 \quad = \quad x_0^2 + \mathbf{2x_1x_2}\xi + \left[\mathbf{2x_0x_1} + \xi x_2^2\right]\alpha + \left[x_1^2 + \mathbf{2x_0x_2}\right]\alpha^2 \quad \text{(B.3.2)}$$

Note that if we first compute $2x_1$, only 2 multiplications by 2 are necessary to evaluate this formula, so the cost is $\mathbf{S}_{3i}^{SB} = 3\mathbf{M}_i + 3\mathbf{S}_i + 3\mathbf{A}_i + 2\mathbf{A}'_i + 2\mathbf{m}_{i,\xi}$.

As usual, the Karatsuba method computes the bolded terms of (B.3.2) with only $\mathbf{S}_i + 3\mathbf{A}_i$ so that $\mathbf{S}_{3i}^K = 6\mathbf{S}_i + 12\mathbf{A}_i + 2\mathbf{m}_{i,\xi}$.

We can also use the Chung-Hasan method [42] for squaring in degree 3 extensions. There are several variants but the most interesting in our context is to compute the term in α^2 in (B.3.2) using the formula

$$x_1^2 + 2x_0x_2 = (x_0 + x_1 + x_2)^2 - (2x_0x_1 + 2x_1x_2 + x_0^2 + x_2^2).$$

Computing this term requires $\mathbf{S}_i + 6\mathbf{A}_i$ instead of $\mathbf{M}_i + \mathbf{S}_i + \mathbf{A}_i + \mathbf{A}'_i$, so the overall complexity is $\mathbf{S}_{3i}^{CH} = 2\mathbf{M}_i + 3\mathbf{S}_i + 8\mathbf{A}_i + \mathbf{A}'_i + 2\mathbf{m}_{i,\xi}$.

B.3.4 Building $\mathbb{F}_{p^{12}}$

Let us now discuss the ways to build the extension tower $\mathbb{F}_{p^{12}}$ in the pairing context that implies some constraints :

- As explained in Section B.2.2, $\mathbb{F}_{p^{12}}$ must be built as an extension of $\mathbb{F}_{p^2}$ because of the use of a sextic twist.
- $\mathbb{F}_{p^{12}}$ must be built over $\mathbb{F}_{p^2}$ with a polynomial $X^6 - \xi$ where ξ is the element used to define the twisted curve. This allows the line involved in the Miller algorithm to be a sparse element of $\mathbb{F}_{p^{12}}$ (see Section B.6 for more details).

Then, $\mathbb{F}_{p^{12}}$ should be built as a cubic extension of a quadratic one of $\mathbb{F}_{p^2}$ (case $2, 2, 3$), as a quadratic extension of a cubic one (case $2, 3, 2$) or as a sextic extension. The latter case is proved to be less efficient [52], so we will only consider the first two ones (in Sections B.3.6 and B.3.7). In any case, we have

$$\mathbb{F}_{p^{12}} = \mathbb{F}_{p^2}[\gamma] \text{ with } \gamma^6 = \xi \in \mathbb{F}_{p^2}.$$

Of course, ξ must be carefully chosen, in order to optimize $\mathbb{F}_{p^{12}}$ arithmetic (that is the purpose of Section B.3.8). But let us first focus on building $\mathbb{F}_{p^2}$.

B.3.5 Choice of $\mathbb{F}_{p^2}$ and its arithmetic

As explained in Section B.3.2, $\mathbb{F}_{p^2}$ is built thanks to a non-square element μ

$$\mathbb{F}_{p^2} = \mathbb{F}_p[\alpha] \text{ with } \alpha^2 = \mu.$$

According to Remark 2, the Karatsuba method for $\mathbb{F}_{p^2}$ multiplication is better when $\mathbf{A}_1 < \frac{1}{3}\mathbf{M}_1$. But according to Section B.3.2 the schoolbook method is better than the Karatsuba one, unless the addition in $\mathbb{F}_p$ is really very efficient. The schoolbook method is also better than the complex one if μ is randomly chosen ($\mathbf{m}_{1,\mu+1} = \mathbf{M}_1$). If μ is chosen small, the best method is dependent on the cost of the multiplication by $\mu + 1$ and then on the choice of μ.

Choice of μ and consequences on $\mathbb{F}_{p^2}$ arithmetic

We focus on multiplications and squarings because other operations are rare or independent of the choice of μ. μ has to be a non-square in $\mathbb{F}_p$ and must be chosen such that $\mathbf{m}_{1,\mu}$ and $\mathbf{m}_{1,\mu+1}$ are as cheap as possible. The best choice is then $\mu = -1$. If this choice is not possible, $\mu = \pm 2$ is also a good choice.

If $\mu = -1$, α is usually denoted by $\mathbf{i}$ and $\mathbf{m}_{1,\mu}$ and $\mathbf{m}_{1,\mu+1}$ are both for free. In fact, the complex method computes $(a_0 + a_1\mathbf{i})^2 = (a_0 - a_1)(a_0 + a_1) + 2a_0a_1\mathbf{i}$ with only $2\mathbf{M}_1 + 2\mathbf{A}_1 + \mathbf{A}'_1$ which is the fastest method in any case.

If $\mu = -2$, $\mathbf{m}_{1,\mu} = \mathbf{A}'_1$ and $\mathbf{m}_{1,\mu+1} = 0$. The complex method for squaring in $\mathbb{F}_{p^2}$ then requires $2\mathbf{M}_1 + 3\mathbf{A}_1 + 2\mathbf{A}'_1$ so that it is better than the schoolbook one if $\mathbf{S}_1 = \mathbf{M}_1$. If $\mathbf{S}_1 = 0.8\mathbf{M}_1$, it is better only if $\mathbf{A}_1 \leq 0.3\mathbf{M}_1$.

If $\mu = -5$, the complex method can be rewritten in a more efficient way

$$(a_0 + a_1\alpha)^2 = (a_0 + a_1 + 2a'_1)(a_0 + a_1) + 2a_0a'_1 + a_0a'_1\alpha \text{ with } a'_1 = 2a_1$$

that requires $2\mathbf{M}_1 + 4\mathbf{A}_1 + 2\mathbf{A}'_1$. This is almost as good as the case $\mu = -2$ and better than the schoolbook method (except if $\mathbf{A}'_1 << \mathbf{A}_1$ and $\mathbf{S}_1 = 0.8\mathbf{M}_1$).

Other small values for μ have no interest. Indeed choosing $\mu = 2$ instead of -2 is less efficient for the complex method ($\mu + 1 = 3$ instead of -1). Moreover it is a square if -1 and -2 are, so it cannot be an alternative. The same remark holds for $\mu = 5$. Finally, since $p = 1 \bmod 3$ for BN primes, -3 is always a square in $\mathbb{F}_p$, so it cannot be chosen to build $\mathbb{F}_{p^2}$ and 3 is a square at the same time as -1 (and $\mu = -1$ is better).

Consequences on u of the choice of μ

The choice of μ can be expressed in terms of u thanks to the following result

Proposition B.3.1. *Let $p = 36u^4 + 36u^3 + 24u^2 + 6u + 1$ be a BN prime, we have*

i) -1 (and then 3) is a square in $\mathbb{F}_p$ if and only if u is even.

ii) 2 *is a square in* $\mathbb{F}_p$ *if and only if* $u = 0$ *or* 1 *mod* 4.

iii) 5 *is a square in* $\mathbb{F}_p$ *if and only if* $u = 0$ *or* 4 *mod* 5.

Démonstration.

i) Since $p = 1+2u \bmod 4$, this is a direct consequence of the classical result stating that -1 is a square in $\mathbb{F}_p$ if and only if $p = 1 \bmod 4$.

ii) It is well known that 2 is a square in $\mathbb{F}_p$ if and only if $p = \pm 1 \bmod 8$. Since $p = 1 + 6u \bmod 8$, 2 is a square in $\mathbb{F}_p$ if and only if $u = 0$ or $1 \bmod 4$.

iii) According to the quadratic reciprocity law, 5 is a square in $\mathbb{F}_p$ if and only if $p = \pm 1 \bmod 5$. It is then easy to check that this holds if $u = 0$ or $4 \bmod 5$.

□

As a consequence, one should choose $\mu = -1$ when u is odd. If u is even and $u = 2 \bmod 4$, $\mu = -2$ should be used. If $u = 0 \bmod 4$, then $\mu = -5$ can and should be chosen when $u = 8, 12$ or $16 \bmod 20$. In the remaining cases, namely $u = 0$ or $4 \bmod 20$, small values for μ have no or little interest.

Summary of choices and complexities for $\mathbb{F}_{p^2}$ arithmetic

We trivially have $\mathbf{A}_2 = 2\mathbf{A}_1$ and $\mathbf{A}'_2 = 2\mathbf{A}'_1$ whatever the choice of μ and the complexities for $\mathbf{M}_2$ and $\mathbf{S}_2$ depending on μ are summarized in Tables B.1 and B.2.

μ	$\mathbf{A}_1 \leq 0.33\mathbf{M}_1$	Algo	$\mathbf{A}_1 > 0.33\mathbf{M}_1$	Algo	Condition on u
-1	$3\mathbf{M}_1 + 5\mathbf{A}_1$	K	$4\mathbf{M}_1 + 2\mathbf{A}_1$	SB	1 mod 2
-2	$3\mathbf{M}_1 + 5\mathbf{A}_1 + \mathbf{A}'_1$	K	$4\mathbf{M}_1 + 2\mathbf{A}_1 + \mathbf{A}'_1$	SB	2 mod 4
-5	$3\mathbf{M}_1 + 6\mathbf{A}_1 + 2\mathbf{A}'_1$	K	$4\mathbf{M}_1 + 3\mathbf{A}_1 + 2\mathbf{A}'_1$	SB	$8, 12, 16 \bmod 20$
any	$4\mathbf{M}_1 + 5\mathbf{A}_1$	K	$5\mathbf{M}_1 + 2\mathbf{A}_1$	SB	$0, 4 \bmod 20$

TABLE B.1 – Complexities of $\mathbf{M}_2$ depending on the way to build $\mathbb{F}_{p^2}$

μ	$\mathbf{S}_1 = \mathbf{M}_1$ or $\mathbf{S}_1 = 0.8\mathbf{M}_1, \mathbf{A}_1 \leq 0.3\mathbf{M}_1$	$\mathbf{S}_1 = 0.8\mathbf{M}_1, \mathbf{A}_1 > 0.3\mathbf{M}_1$	Algo	Condition on u
-1	$2\mathbf{M}_1 + 2\mathbf{A}_1 + \mathbf{A}'_1$		$\mathbb{C}$	1 mod 2
-2	$2\mathbf{M}_1 + 3\mathbf{A}_1 + 2\mathbf{A}'_1$	$\mathbf{M}_1 + 2\mathbf{S}_1 + \mathbf{A}_1 + 2\mathbf{A}'_1$	$\mathbb{C}$/SB	2 mod 4
-5	$2\mathbf{M}_1 + 4\mathbf{A}_1 + 2\mathbf{A}'_1$		$\mathbb{C}$	8,12,16 mod 20
any	$2\mathbf{M}_1 + 2\mathbf{S}_1 + \mathbf{A}_1 + \mathbf{A}'_1$		SB	0,4 mod 20

TABLE B.2 – Complexities of $\mathbf{S}_2$ depending on the way to build $\mathbb{F}_{p^2}$

Remark 4. *The relative cost of* $\mathbf{A}'_1$ *compared to* $\mathbf{A}_1$ *has essentially no influence on the choices to be made, even if* -2 *is chosen to define* $\mathbb{F}_{p^2}$

Thanks to Tables B.1 and B.2, it is easy to choose the best algorithm for $\mathbb{F}_{p^2}$ multiplication and squaring depending on the context.

B.3.6 Choice of $\mathbb{F}_{p^{12}}$ arithmetic in the case $2, 3, 2$

In this section, $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^2}$ is built via $\mathbb{F}_{p^6}$ and $\xi \in \mathbb{F}_{p^2}$ neither a square nor a cube

$$\mathbb{F}_{p^6} = \mathbb{F}_{p^2}[\beta] \text{ with } \beta^3 = \xi \quad \text{and} \quad \mathbb{F}_{p^{12}} = \mathbb{F}_{p^6}[\gamma] \text{ with } \gamma^2 = \beta.$$

$\mathbb{F}_{p^6}$ arithmetic

We saw in Section B.3.3 that the Karatsuba method becomes interesting when $\mathbf{M}_2 \geq 3\mathbf{A}_2$. Regarding Table B.1, this condition is clearly always satisfied. Therefore, the Karatsuba method should always be used for $\mathbb{F}_{p^6}/\mathbb{F}_{p^2}$ multiplications so that $\mathbf{M}_6 = 6\mathbf{M}_2 + 15\mathbf{A}_2 + 2\mathbf{m}_{2,\xi}$.

It is also easy to check that the Chung-Hasan method is the best in each case of Tables B.1 and B.2. Therefore, it should always be used for $\mathbb{F}_{p^6}/\mathbb{F}_{p^2}$ squaring and $\mathbf{S}_6 = 2\mathbf{M}_2 + 3\mathbf{S}_2 + 8\mathbf{A}_2 + \mathbf{A}'_2 + 2\mathbf{m}_{2,\xi}$.

$\mathbb{F}_{p^{12}}$ arithmetic

We obviously have $\mathbf{A}_{12} = 12\mathbf{A}_1$ and $\mathbf{A}'_{12} = 12\mathbf{A}'_1$. According to Section B.3.2, the Karatsuba method for $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$ multiplications is better than the schoolbook one when $3\mathbf{A}_6 < \mathbf{M}_6$, which is obviously always the case according to B.3.6. The $\mathbb{F}_{p^6}$ multiplication by β involved in Karatsuba formulas is given by

$$\left(b_0 + b_1\beta + b_2\beta^2\right)\beta = \xi b_2 + b_0\beta + b_1\beta^2,$$

thus $\mathbf{m}_{6,\beta} = \mathbf{m}_{2,\xi}$. We then finally have $\mathbf{M}_{12} = 18\mathbf{M}_2 + 60\mathbf{A}_2 + 7\mathbf{m}_{2,\xi}$.

During the Miller loop of the Ate pairing, one of the operands (the line ℓ) is sparse ($\ell = b_0 + (b_1 + b_3\beta)\gamma$ as explained in Section B.6) but the Karatsuba method remains the best way to multiply ℓ by any $c_0 + c_1\gamma \in \mathbb{F}_{p^{12}}$ as

$$b_0c_0 + (b_1 + b_3\beta)c_1\beta + \left(\left(b_0 + b_1 + b_3\beta\right)\left(c_0 + c_1\right) - b_0c_0 - \left(b_1 + b_3\beta\right)c_1\right) \qquad \text{(B.3.3)}$$

Evaluating this formula requires

- one multiplication of $c_0 \in \mathbb{F}_{p^6}$ by $b_0 \in \mathbb{F}_{p^2}$ which trivially costs $3\mathbf{M}_2$,
- one multiplication of $c_1 \in \mathbb{F}_{p^6}$ by $b_1 + b_3\beta$. It is done thanks to the sparse Karatsuba multiplication $\mathbf{sM}_6^K$ given in Section B.3.3 for $5\mathbf{M}_2 + 6\mathbf{A}_2 + \mathbf{m}_{2,\xi}$,
- $4\mathbf{A}_2$ to compute $b_0 + b_1$ and $c_0 + c_1$ and another $\mathbf{sM}_6^K$:$(c_0 + c_1)(b_0 + b_1 + b_3\beta)$,
- $\mathbf{m}_{6,\beta} = \mathbf{m}_{2,\xi}$ and $3\mathbf{A}_6$ to compute the final result.

Hence a $\mathbb{F}_{p^{12}}$ sparse multiplication requires $\mathbf{sM}_{12} = 13\mathbf{M}_2 + 25\mathbf{A}_2 + 3\mathbf{m}_{2,\xi}$.

Using the complexities obtained for $\mathbf{M}_6$ and $\mathbf{S}_6$ in B.3.6 and Tables B.1 and B.2, we can easily verify that using the Karatsuba method for squaring in $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$ is always better than the schoolbook one. On the other hand, the difference with the complex method is $\mathbf{S}_{12}^K - \mathbf{S}_{12}^c = 9\mathbf{S}_2 - 6\mathbf{M}_2 - 6\mathbf{A}_2 + \mathbf{m}_{2,\xi}$ whose sign is depending on μ. We do not give the details but we can easily determine that

- The Karatsuba method should be used if $\mu = -1$ or -5 but also if $\mu = -2$ assuming that $\mathbf{A}_1 \leq 0.33\mathbf{M}_1$ or $\mathbf{S}_1 = 0.8\mathbf{M}_1$. In these cases, the squaring complexity is given by $\mathbf{S}_{12} = 6\mathbf{M}_2 + 9\mathbf{S}_2 + 36\mathbf{A}_2 + 3\mathbf{A}'_2 + 7\mathbf{m}_{2,\xi}$.

- The complex method should be used if μ is not small and if $\mu = -2$ with $\mathbf{A}_1 > 0.33\mathbf{M}_1$ and $\mathbf{S}_1 = \mathbf{M}_1$. In this case $\mathbf{S}_{12} = 12\mathbf{M}_2 + 42\mathbf{A}_2 + 3\mathbf{A}'_2 + 6\mathbf{m}_{2,\xi}$.

B.3.7 Choice of $\mathbb{F}_{p^{12}}$ arithmetic in the case $2, 2, 3$

In this section, $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^2}$ is built via $\mathbb{F}_{p^4}$ and $\xi \in \mathbb{F}_{p^2}$ which is neither a square nor a cube.

$$\mathbb{F}_{p^4} = \mathbb{F}_{p^2}[\beta] \text{ where } \beta^2 = \xi \quad \text{and} \quad \mathbb{F}_{p^{12}} = \mathbb{F}_{p^4}[\gamma] \text{ with } \gamma^3 = \beta.$$

$\mathbb{F}_{p^4}$ arithmetic

As in the case $2, 3, 2$, the Karatsuba method should always be used for $\mathbb{F}_{p^4}/\mathbb{F}_{p^2}$ multiplications and $\mathbf{M}_4 = 3\mathbf{M}_2 + 5\mathbf{A}_2 + \mathbf{m}_{2,\xi}$.

All the methods to square in $\mathbb{F}_{p^4}/\mathbb{F}_{p^2}$ are close. We do not give details here, but thanks to Tables B.1 and B.2, we can decide and summarize in Table B.3 which method is preferable depending on the context.

Assumption	μ	Complexity	Algo
	-1 or -5	$3\mathbf{S}_2 + \mathbf{m}_{2,\xi} + 4\mathbf{A}_2$	K
$\mathbf{A}_1 \leq 0.33\mathbf{M}_1$	-2	$3\mathbf{S}_2 + \mathbf{m}_{2,\xi} + 4\mathbf{A}_2$	K
	any	$2\mathbf{M}_2 + \mathbf{m}_{2,\xi} + \mathbf{m}_{2,\xi+1} + 3\mathbf{A}_2 + \mathbf{A}'_2$	$\mathbb{C}$
$\mathbf{A}_1 > 0.33\mathbf{M}_1$	-2	$3\mathbf{S}_2 + \mathbf{m}_{2,\xi} + 4\mathbf{A}_2$	K
$\mathbf{S}_1 = 0.8\mathbf{M}_1$	any	$\mathbf{M}_2 + 2\mathbf{S}_2 + \mathbf{m}_{2,\xi} + \mathbf{A}_2 + \mathbf{A}'_2$	SB
$\mathbf{A}_1 > 0.33\mathbf{M}_1$	-2	$2\mathbf{M}_2 + \mathbf{m}_{2,\xi} + \mathbf{m}_{2,\xi+1} + 3\mathbf{A}_2 + \mathbf{A}'_2$	$\mathbb{C}$
$\mathbf{S}_1 = \mathbf{M}_1$	any		

TABLE B.3 – Complexities of $\mathbf{S}_4$ depending on the context

$\mathbb{F}_{p^{12}}$ arithmetic

In this case, we also have $\mathbf{A}_{12} = 12\mathbf{A}_1$, $\mathbf{A}'_{12} = 12\mathbf{A}'_1$ and $3\mathbf{A}_4 < \mathbf{M}_4$ so that the Karatsuba method should be used for multiplications in $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^4}$. The $\mathbb{F}_{p^4}$ multiplication by β involved in Karatsuba formulas is given by

$$(b_0 + b_1\beta)\,\beta = \xi b_1 + b_0\beta,$$

thus $\mathbf{m}_{4,\beta} = \mathbf{m}_{2,\xi}$. We then finally have

$$\mathbf{M}_{12} = 18\mathbf{M}_2 + 60\mathbf{A}_2 + 8\mathbf{m}_{2,\xi}.$$

As in the case $2,3,2$, the sparse multiplication of any $c \in \mathbb{F}_{p^{12}}$ by the sparse line $b_0 + b_3\beta + b_1\gamma$ (with $b_i \in \mathbb{F}_{p^2}$) is done using Karatsuba as in Section B.3.3

$$\begin{aligned}(c_0 + c_1\gamma + c_2\gamma^2)(b_0 + b_3\beta + b_1\gamma) = c_0(b_0 + b_3\beta) + \beta c_2 b_1 \\ + [(c_0 + c_1)(b_0 + b_3\beta + b_1) - c_0(b_0 + b_3\beta) - c_1 b_1]\gamma \\ + [c_2(b_0 + b_3\beta) + c_1 b_1]\gamma^2 \end{aligned} \quad \text{(B.3.4)}$$

Evaluating this formula requires

- 2 multiplications of c_1 (or c_2) $\in \mathbb{F}_{p^4}$ by $b_1 \in \mathbb{F}_{p^2}$ which cost $2\mathbf{M}_2$ each.
- 3 $\mathbb{F}_{p^4}$ Karatsuba multiplications which cost $3\mathbf{M}_2 + 5\mathbf{A}_2 + \mathbf{m}_{2,\xi}$ each.
- $3\mathbf{A}_2$ (for $b_0 + b_1$ and $c_0 + c_1$), $1\mathbf{m}_{4,\beta}$ and finally $4\mathbf{A}_4$ to add all terms.

Hence a $\mathbb{F}_{p^{12}}$ sparse multiplication requires $\mathbf{sM}_{12} = 13\mathbf{M}_2 + 26\mathbf{A}_2 + 4\mathbf{m}_{2,\xi}$.

We do not explain why the Chung-Hasan method should always be used for $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^4}$ squaring because it would be repetitive and tedious after previous sections but it is easy to get that $\mathbf{S}_{12} = 3\mathbf{S}_4 + 6\mathbf{M}_2 + 26\mathbf{A}_2 + 2\mathbf{A}'_2 + 4\mathbf{m}_{2,\xi}$.

B.3.8 Choice of ξ and consequences on u

Contrary to μ, ξ has no influence on arithmetic choices on $\mathbb{F}_{p^{12}}$, but multiplications by ξ still should be optimized to maximize performances. Note that an efficient μ should be chosen in priority because it has much more influence on $\mathbb{F}_{p^{12}}$ efficiency.

Once μ is chosen, ξ must be chosen neither a square nor a cube in $\mathbb{F}_{p^2}$. Let us first give some results characterizing cubes and squares.

Proposition B.3.2. *Let p be an odd prime number. If $p \equiv 1[3]$ (always true for BN primes), then an element is a square (resp. a cube) in $\mathbb{F}_{p^2}$ if and only if its norm is a square (resp. a cube) in $\mathbb{F}_p$.*

The proof is classical in number theory. It is involving the linear maps $x \mapsto x^2$ (resp. x^3) and $x \mapsto x^{\frac{p^2-1}{2}}$ (resp. $x^{\frac{p^2-1}{3}}$).

Corollary 1. *Let p as in Proposition B.3.2, any element of $\mathbb{F}_p$ is a square in $\mathbb{F}_{p^2}$.*

Démonstration. As $N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(n) = n^2$ is a square in $\mathbb{F}_p$, Proposition B.3.2 implies that n is a square in $\mathbb{F}_{p^2}$. □

Proposition B.3.3. *Let $p = 36u^4 + 36u^3 + 24u^2 + 6u + 1$ be a BN prime, we have*

- *2 is a cube in $\mathbb{F}_p$ if and only if $u = 0$ mod 3.*
- *3 is a cube in $\mathbb{F}_p$ if and only if $u = 0, 1$ or 5 mod 9.*
- *5 is a cube in $\mathbb{F}_p$ if and only if $u = 0, 2, 4, 6$ or 8 mod 15.*

Démonstration. This result is based on two old statements. The first one is a theorem of Fermat stating that every prime $p = 1 \bmod 3$ can be uniquely (up to signs) written in the form $a^2 + 3b^2$. The second one is known as Euler's conjectures and is proven by Cox in [48]. It states that if $p = a^2 + 3b^2$ as above, we have

- 2 is a cube in $\mathbb{F}_p$ if and only if $3|b$.
- 3 is a cube in $\mathbb{F}_p$ if and only if $9|b$ or $9|a \pm b$.
- 5 is a cube in $\mathbb{F}_p$ if and only if $15|b$ or $3|b, 5|a$ or $15|a \pm b$ or $15|2a \pm b$.

The Fermat representation of BN primes can be explicitly computed. We have $p = (6u^2 + 3u + 1)^2 + 3u^2$ and Euler's conjectures ends the proof. □

Remark 5. *Since -1 is always a square and a cube in $\mathbb{F}_{p^2}$, there is no interest to consider the sign of ξ. In the same way, there is no interest to consider the conjugates of ξ because they are squares and cubes at the same times as ξ.*

If $\alpha^2 = \mu$ defines $\mathbb{F}_{p^2}$, the most natural choice for ξ is α, so that $\mathbf{m}_{2,\xi} = \mathbf{m}_{1,\mu}$

$$(a_0 + a_1\alpha)\xi = \mu a_1 + a_0\alpha.$$

However, this choice is not always possible. For example, the best choice for $\mathbb{F}_{p^2}$ is $\mu = -1$ but it is obviously always a cube. Then ξ must be chosen as an element of $\mathbb{F}_{p^2}$ with small coefficients. According to Corollary 1, it is not possible to choose small integers and according to Remark 5 we can consider only positive coefficients. Then, after α, the best candidates are $1+\alpha$ and $2+\alpha$ (the latter is of course less interesting in terms of efficiency). Let us now precise the possible choices for ξ depending on μ and their consequences on the parameter u.

The case $\mu = -1$

The smallest complexity for $\mathbf{m}_{2,\xi}$ is reached by $\xi = 1 + \mathbf{i}$ and equals $2\mathbf{A}_1$ since

$$(1 + \mathbf{i})(a_0 + a_1\mathbf{i}) = a_0 - a_1 + (a_0 + a_1)\mathbf{i}.$$

Théorème B.3.4. *Let p be a BN prime with u odd (thus $\mathbb{F}_{p^2}$ can be defined by $\mathbf{i} = \sqrt{-1}$). Then $1 + \mathbf{i}$ is neither a square nor a cube in $\mathbb{F}_{p^2}$ if and only if $u = 7$ or 11 mod 12. In this case, $\mathbb{F}_{p^{12}}$ can be defined over $\mathbb{F}_{p^2}$ by a sixth root of $1 + \mathbf{i}$.*

Démonstration. By Proposition B.3.2, $1 + \mathbf{i}$ is a square (resp. a cube) in $\mathbb{F}_{p^2}$ if and only if $2 = N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(1 + \mathbf{i})$ is a square (resp. a cube) in $\mathbb{F}_p$.

- According to Proposition B.3.1, 2 is a square in $\mathbb{F}_p$ if and only if $u = 0$ or $1 \bmod 4$. As u is odd, $1 + \mathbf{i}$ is not a square in $\mathbb{F}_{p^2}$ if and only if $u = 3 \bmod 4$.
- In the same way, $1 + \mathbf{i}$ is not a cube if and only if 2 is not a cube which means, according to Proposition B.3.3, that $u \neq 0 \bmod 3$. Thus $1 + \mathbf{i}$ is not a cube if and only if $u = 1$ or $5 \bmod 6$.

Combining these two congruences ends the proof. □

If $1+\mathbf{i}$ cannot be chosen, the next candidate in terms of efficiency is $\xi = 2+\mathbf{i}$ for which $\mathbf{m}_{2,\xi} = 2\mathbf{A}_1 + 2\mathbf{A}'_1$ and whose norm is 5. Combining the conditions ensuring that 5 is neither a square nor a cube (Proposition B.3.1 and B.3.3) and the fact that $u \neq 7$ or 11 mod 12 (otherwise $\xi = 1+\mathbf{i}$ is chosen), we get that $2+\mathbf{i}$ should be chosen to define $\mathbb{F}_{p^{12}}$ if $u = 1, 3, 13, 27, 33, 37, 41$ or 57 mod 60. In 3 of the remaining cases ($u = 17, 39$ and 53 mod 60), one can choose $3+\mathbf{i}$, whose norm is 10. Of course, this study can be easily continued with "larger" values of ξ ($4+\mathbf{i}, 3+2\mathbf{i}, ...$) to deal with the 9 remaining cases. But the interest is limited and the reader can easily do it by himself if necessary.

The case $\mu = -2$

This choice should be made only if $\mu = -1$ cannot so we assume that $u = 2$ mod 4 in this section. The best choice for ξ is of course $\alpha = \sqrt{-2}$.

Théorème B.3.5. *Let p be a BN prime with $u = 2$ mod 4. Then $\alpha = \sqrt{-2}$ is neither a square nor a cube in $\mathbb{F}_{p^2}$ if and only if $u = 2$ or 10 mod 12. In this case, $\mathbb{F}_{p^{12}}$ can be defined over $\mathbb{F}_{p^2}$ by a sixth root of α.*

Démonstration.

- Since -1 is a square and $\mu = -2$ is not, $N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(\alpha) = 2$ is never a square in $\mathbb{F}_p$ and thus, by proposition B.3.2, α is never a square in $\mathbb{F}_{p^2}$.
- α is a cube in $\mathbb{F}_{p^2}$ if and only if $2 = -N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(\alpha)$ is a cube in $\mathbb{F}_p$. According to Proposition B.3.3, 2, and then α, is not a cube if and only if $u = 1$ or 2 mod 3. Taking into account that -1 is a square and -2 is not a square, this condition can be rewritten $u = 2$ or 10 mod 12.

□

Unfortunately, $1+\alpha$ cannot be chosen since its norm equals 3 which is always a square in the remaining case ($u = 6$ mod 12). The situation is better with $2+\alpha$ whose norm is 6 which is never a square in $\mathbb{F}_p$ and is a cube at the same time as 3. According to Proposition B.3.3, $\xi = 2+\alpha$ can then be chosen if $u = 6$ or 30 mod 36. In this case $\mathbf{m}_{2,\xi} = 2\mathbf{A}_1 + 2\mathbf{A}'_1$. Again, this can be easily continued to deal with the last remaining case ($u = 18$ mod 36).

The case $\mu = -5$

This choice of μ has to be done only if -1 and -2 are squares. So we assume that $u = 8, 12$ or 16 mod 20 in this section. Again, the best choice for ξ is α.

Théorème B.3.6. *Let p be a BN prime with $u = 8, 12$ or 16 mod 20. Then $\alpha = \sqrt{-5}$ is neither a square nor a cube in $\mathbb{F}_{p^2}$ if and only if $u = 12, 16, 28, 48, 52, 56$ mod 60. In this case, $\mathbb{F}_{p^{12}}$ can be defined over $\mathbb{F}_{p^2}$ by a sixth root of α.*

Démonstration. As for Theorem B.3.5, we prove that α is never a square in $\mathbb{F}_{p^2}$. Moreover, the norm of α is -5 and Proposition B.3.3 gives a condition ensuring that 5, and then α, is not a cube and allows to conclude. □

Again $1+\alpha$ cannot be chosen since its norm equals 6 which is always a square in this case. The situation is the same for $2+\alpha$ since its norm is 9. The "smallest" usable candidates are $1+2\alpha$ and $3+\alpha$ whose norms are respectively 21 and 14. This will give conditions on u mod 7 but there are very few concerned cases ($u = 8, 32$ or 36 mod 60) and the cost of $\mathbf{m}_{2,\xi}$ becomes high.

Let us finally summarize the possible choices of ξ in term of u in Table B.4.

μ	Value of u mod					ξ	$\mathbf{m}_{2,\xi}$
	4	12	20	36	60		
-1	$1,3$	$7,11$				$1+\mathbf{i}$	$2\mathbf{A}_1$
		$1,3,5,9$			$1,3,13,27,33,37,41,57$	$2+\mathbf{i}$	$2\mathbf{A}_1+2\mathbf{A}'_1$
					$17,39,53$	$3+\mathbf{i}$	$4\mathbf{A}_1+2\mathbf{A}'_1$
					$5,9,15,21,25,29,45,49,51$	–	–
-2	2	$2,10$				$\sqrt{-2}$	$\mathbf{A}'_1$
		6		$6,30$		$2+\sqrt{-2}$	$2\mathbf{A}_1+2\mathbf{A}'_1$
				18		–	–
-5	0	$0,4,8$	$8,12,16$		$12,16,28,48,52,56$	$\sqrt{-5}$	$\mathbf{A}_1+2\mathbf{A}'_1$
					$8,32,36$	–	–
any			$0,4$			$\sqrt{\mu}$	$\mathbf{M}_1$

TABLE B.4 – Best choices of μ and ξ in terms of u

B.3.9 New improvements of $\mathbb{F}_{p^{12}}$ arithmetic

In our context (additions are not neglected), we get two new improvements saving some additions. The first one is a computational trick probably already used in good implementations but is usually not explicitly written. The second one consists in precomputing the traces of some elements of the intermediate levels of the extension tower because there are involved in several operations.
These results could have been included in our previous discussion on $\mathbb{F}_{p^{12}}$ construction but first, they require some precomputations storage which is not always desirable and second, they have no influence on the choices to be made. For readability, the details are postponed to Appendix B.9.

Multiplications by $\xi - 1$ in Karatsuba operations

If $\mathbf{m}_{i,\xi-1} \leq \mathbf{m}_{i,\xi}$, the Karatsuba multiplication of $x_0 + x_1\beta$ by $y_0 + y_1\beta$ in $\mathbb{F}_{p^{2i}}$ can be evaluated as

$$x_0y_0 + x_1y_1 + (\xi-1)x_1y_1 + ((x_0+x_1)(y_0+y_1) - x_0y_0 - x_1y_1)\beta$$

Then $\mathbf{m}_{i,\xi}$ is replaced by $\mathbf{m}_{i,\xi-1}$. The same can be done in $\mathbb{F}_{p^{3i}}$ and allows to save $2\mathbf{A}_1$ in the intermediate field Karatsuba multiplications or squarings when ξ is not $\sqrt{\mu}$. This trick has probably been used already by implementors (especially when $\xi = 1+\mathbf{i}$) but we did not find it written in the literature.

Precomputed traces

If an element is used in several operations, we can precompute and store some quantities. For example, if $x_0 + x_1 = tr_{\mathbb{F}_{p^{2i}}/\mathbb{F}_{p^i}}(x_0 + x_1\alpha)$ is precomputed, computing $(x_0 + x_1\alpha)^2$ in $\mathbb{F}_{p^{2i}}$ using Karatsuba requires $3\mathbf{A}_i$ instead of 4. The same remark holds for all the squaring or multiplying methods in $\mathbb{F}_{p^{2i}}$ or $\mathbb{F}_{p^{3i}}$ involving traces (more precisely sums of coordinates). It is summarized in Table B.5.

Operation	Method	Precomputations	Saving
$(x_0 + x_1\alpha)(y_0 + y_1\alpha)$	$\mathbf{M}_{2i}^K$	$x_0 + x_1, y_0 + y_1$	$2\mathbf{A}_i$
$(x_0 + x_1\alpha)^2$	$\mathbf{S}_{2i}^K$ or $\mathbf{S}_{2i}^{\mathbb{C}}$	$x_0 + x_1$	$\mathbf{A}_i$
$(x_0 + x_1\xi + x_2\xi^2)(y_0 + y_1\xi + y_2\xi^2)$	$\mathbf{M}_{3i}^K$	$x_0+x_1, x_0+x_2, x_1+x_2$ $y_0+y_1, y_0+y_2, y_1+y_2$	$6\mathbf{A}_i$
$(x_0 + x_1\xi + x_2\xi^2)^2$	$\mathbf{S}_{3i}^K$	$x_0+x_1, x_0+x_2, x_1+x_2$	$3\mathbf{A}_i$
	$\mathbf{S}_{3i}^{CH}$	$x_0 + x_1 + x_2$	$2\mathbf{A}_i$

TABLE B.5 – Precomputing traces

Of course, we get savings only if the precomputations are used in several operations. For example, a schoolbook multiplication in $\mathbb{F}_{p^{2i}}$ will use x_0 both for x_0y_0 and for x_0y_1. On the contrary no element is used twice in Karatsuba operations or in complex squarings. As Karatsuba is always used at higher levels of the extension tower (according to Sections B.3.6 and B.3.7), precomputing traces seems not interesting for $\mathbb{F}_{p^{12}}$ multiplications. However, it can be applied in three situations in the pairing context

- The final exponentiation involves several $\mathbb{F}_{p^{12}}$ multiplications by a constant term, so the traces involved in this constant term should be precomputed.
- Sparse multiplications involve schoolbook steps which take advantage of trace precomputation.
- 3 elements are used twice in Chung-Hasan method for $\mathbb{F}_{p^{3i}}/\mathbb{F}_{p^i}$ squarings.

These situations are detailed in Appendix B.9 and the savings obtained are given in table B.6 when Karatsuba is used for $\mathbb{F}_{p^2}$ arithmetic. The saving are less interesting if schoolbook is used for $\mathbb{F}_{p^2}$ arithmetic, so that Karatsuba should be always use for $\mathbb{F}_{p^2}$ arithmetic in a pairing context despite Remark 2.

Operation	Case	Arithmetic used				Saving
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$	
$\mathbf{M}_{12}$	2,3,2	K		K	K	$42\mathbf{A}_1$
	2,2,3		K			$42\mathbf{A}_1$
$\mathbf{sM}_{12}$	2,3,2	K		K	K	$9\mathbf{A}_1$
	2,2,3		K			$11\mathbf{A}_1$
$\mathbf{S}_{12}$	2,3,2	K/$\mathbb{C}$		CH	K	$11\mathbf{A}_1$
	2,2,3		K		CH	$15\mathbf{A}_1$

TABLE B.6 – Savings provided by trace precomputation

In all considered cases, the saving obtained is around 10% of the total number of additions in $\mathbb{F}_{p^{12}}$ operations, which is not negligible.

B.3.10 Summary of $\mathbb{F}_{p^{12}}$ arithmetic

Let us now recapitulate the choices to be made for $\mathbb{F}_{p^{12}}$ in the pairing context. We first assume that $\mu = -1$ has been chosen. This is possible if u is odd and ξ can be chosen "small" in most cases :

- $\xi = 1 + \mathbf{i}$ if $u = 7$ or $11 \bmod 12$ (and then $\mathbf{m}_{2,\xi} = 2\mathbf{A}_1$)
- $\xi = 2 + \mathbf{i}$ if $u = 1, 3, 13, 27, 33, 37, 41, 57 \bmod 60$ (and $\mathbf{m}_{2,\xi} = 2\mathbf{A}_1 + 2\mathbf{A}'_1$)
- $\xi = 3 + \mathbf{i}$ if $u = 17, 39$ or $53 \bmod 60$ (and $\mathbf{m}_{2,\xi} = 4\mathbf{A}_1 + 2\mathbf{A}'_1$)

If $\mathbf{A}_1 \leq 0.33\mathbf{M}_1$ or if traces are precomputed, $\mathbb{F}_{p^{12}}$ multiplication use Karatsuba arithmetic at all levels of the tower. For squaring in $\mathbb{F}_{p^{12}}$, the Chung Hasan method is used at the degree 3 levels ($\mathbb{F}_{p^6}/\mathbb{F}_{p^2}$ or $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^4}$) but other operations use the Karatsuba or the complex method. The overall complexities are summarized in Table B.7 where K, CH, SB and $\mathbb{C}$ denote the methods used for arithmetic (K/$\mathbb{C}$ means that Karatsuba is used for multiplications and the complex method is used for squarings).

Operation	Case	Arithmetic used				Number of operations without/with improvement			
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$	$\mathbf{M}_1$	$\mathbf{A}_1$	$\mathbf{A}'_1$	$\mathbf{m}_{2,\xi}$
$\mathbf{M}_{12}$	2,3,2	K		K	K	54	210/162	0	7
	2,2,3		K				210/156		8
$\mathbf{sM}_{12}$	2,3,2	K		K	K	39	115/106	0	3
	2,2,3		K				117/100		4
$\mathbf{S}_{12}$	2,3,2	K/$\mathbb{C}$		CH	K	36	120/109	15	7
	2,2,3		K		CH		124/99	13	

TABLE B.7 – $\mathbb{F}_{p^{12}}$ complexities if $\mu = -1$ (assuming $\mathbf{A}_1 \leq 0.33\mathbf{M}_1$ if our new improvements are not used)

If $\mathbf{A}_1 > 0.33\mathbf{M}_1$ and our improvement are not used, the schoolbook method should be used on $\mathbb{F}_{p^2}$ and we get the Table B.8.

Operation	Case	Arithmetic used				Number of operations			
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$	$\mathbf{M}_1$	$\mathbf{A}_1$	$\mathbf{A'}_1$	$\mathbf{m}_{2,\xi}$
$\mathbf{M}_{12}$	$2,3,2$	SB		K	K	72	156	0	7
	$2,2,3$		K				156		8
$\mathbf{sM}_{12}$	$2,3,2$	SB		K	K	52	76	0	3
	$2,2,3$		K				78		4
$\mathbf{S}_{12}$	$2,3,2$	SB/$\mathbb{C}$		CH	K	42	102	15	7
	$2,2,3$		K		CH		106	13	

TABLE B.8 – $\mathbb{F}_{p^{12}}$ complexities if $\mu = -1$ assuming $\mathbf{A}_1 > 0.33\mathbf{M}_1$

The same work has been done when $\mu = -2, -5$ or is large. The corresponding tables are given in Appendix B.10 for readability. Looking at these tables, we can make some interesting remarks :

- It is slightly better to choose a $2,3,2$ extension if our new improvements are not used. This is in accordance with the state of the art [52, 41, 4, 92]. However, our improvements take more advantage of a $2,2,3$ extension. Anyway, the difference between the two ways to build $\mathbb{F}_{p^{12}}$ is negligible.
- As already mentioned, the choice of μ has a great influence on the number of additions involved in $\mathbb{F}_{p^{12}}$ arithmetic, so it should be chosen in priority, even if the choice of ξ is not optimal in this case. For example, it is better to choose $\mu = -2, \xi = 2 + \sqrt{-2}$ than $\mu = -5, \xi = \sqrt{-5}$.
- The $\mathbf{S}_1/\mathbf{M}_1$ and $\mathbf{A'}_1/\mathbf{A}_1$ ratios have very little impact on the choices to be made for $\mathbb{F}_{p^{12}}$ arithmetic.
- The main assumption of this paper (additions should not be neglected) is highly justified because there are many additions.
- Under this assumption, our new improvements have a significant impact on $\mathbb{F}_{p^{12}}$ arithmetic. For example if $\mathbf{A}_1 = 0.25\mathbf{M}_1$ (which is probably not far from the average cost of additions in hardware implementations), we get a theoretical gain of 4.7% for sparse multiplication, 7.8% for squarings and 11.8% for multiplications in $\mathbb{F}_{p^{12}}$. These are the most consuming operations of a pairing computation but there are not the only ones so we cannot conclude about the global gain. Moreover, these theoretical gains should be validated by an implementation but it was not the goal of this work.

B.4 Choosing u

The parameter u is involved at several levels of the pairing computation, so that the best choice is not trivial to do. Let us summarize the constraints on u.

- The parameter u is defining the security level. Indeed, it is parametrizing both the size of the elliptic curve (which is $36u^4 + 36u^3 + 18u^2 + 6u + 1$) and the size of the target finite field (which is $(36u^4 + 36u^3 + 24u^2 + 6u + 1)^{12}$).

- The exponent of the Miller loop is $6u+2$ so that $6u+2$ should be sparse.
- If the addition chain given in Section B.2.4 is used for the final exponentiation, u is used (3 times) as an exponent, so it should be chosen sparse. Other final exponentiation methods may involve exponents $6u+5$ and $6u^2+1$ [53, 59] or $6u+4$ [59] but these quantities are usually sparse at the same time as u.
- The sign of u has no consequence in terms of complexity. Indeed, even if changing u in $-u$ requires an $\mathbb{F}_{p^{12}}$ inversion, this inversion can be replaced by a $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$ conjugation thanks to the final exponentiation.
- Choosing u with a signed binary representation (to facilitate the research of a sparse u) is possible if the exponentiation algorithms are adapted.
- The choice of u has a great impact on $\mathbb{F}_{p^{12}}$ arithmetic. As shown in Section B.3, the choice of the extension tower is a direct consequence of the value of u mod $12, 20, 36$ or 60. It is summarized in Table B.4. The best choice is $u = 7$ or 11 mod 12 so that we can use $\mu = -1$ and $\xi = 1 + \mathbf{i}$. Depending on the situation, it could be better to choose a sparser $u \neq 7, 11 \bmod 12$ or reciprocally a u of higher Hamming weight but equal to 7 or 11 mod 12.

Algorithm 8 recapitulates how the parameter u is generated ($w(u)$ is the Hamming weight of the signed binary representation of u). It can be used to generate a database of pairing friendly parameters, new parameters for any security levels or twist-secure parameters (with an additional primality test).

Algorithme 8 : Generating u

Data : A security level n
Result : A BN parameter u ensuring the security level n
1 **Prerequisite :** Determine the bitsize b of u such that $p=36u^4+36u^3+24u^2+6u+1$ has at least $2n$ bits and computing a discrete logarithm in $\mathbb{F}_{p^{12}}$ requires at least 2^n operations.
2 $h \leftarrow 2$, `cond` $\leftarrow$ "equals 7 or 11 mod 12"
3 **while** nothing is returned **do**
4 **for** u in $[2^{b-1}, 2^b[$ such that $w(u) = h$ and u verifies `cond` **do**
5 **if** p and $r = 36u^4+36u^3+18u^2+6u+1$ are prime **then**
6 **return** u
7 $h \leftarrow h+1$ or change `cond` according to Table B.4 (depending on the context)

The prerequisite step is crucial in this algorithm. If we follow NIST recommendations [156] for the 128-bit security level, u should be a 63-bit integer. That is the reason why $-2^{62}-2^{55}-1$ [155] (which equals 11 mod 12) was widely used in the literature. But according to [118] the security level provided by this value is in fact only around 100 bits. Of course, Algorithm 8 can be used to generate new parameters but the prerequisite step requires to precisely estimate the complexity of the algorithm given in [118] in the particular case of BN curves. We do not do it here because it is a highly non-trivial task but according to existing draft works [93, 139], u should probably be chosen with around 100 bits.

B.5 Choosing the groups involved

B.5.1 Choosing the elliptic curves

Once u is chosen, p and r are fixed by the BN parametrization. Then, we only have to find $b \in \mathbb{F}_p$ such that the curve E defined by the equation $y^2 = x^3 + b$ has cardinality r. Because of the sextic twist, there are 6 isomorphism classes over $\mathbb{F}_p$, so there is about one chance in 6 that $\#E(\mathbb{F}_p) = r$. Checking the cardinality is easily done by verifying that $rP = O_E$ for some $P \in E(\mathbb{F}_p)$.

Remark 6. *We will see in Section B.6 that the coefficient b is involved in the elliptic curve arithmetic, so that it is better to choose it small.*

If ξ is the $\mathbb{F}_{p^2}$ element defining $\mathbb{F}_{p^{12}}$, the twist E' of E is defined over $\mathbb{F}_{p^2}$ by

$$y^2 = x^3 + b' \text{ where } b' = b/\xi \text{ or } b' = b\xi.$$

The correct choice of b' is such that $r|\#E'(\mathbb{F}_{p^2})$ (which holds for exactly one of the two choices [3]). For example, $b = N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(\xi)$ and $b' = \overline{\xi}$ can be used [87].

Remark 7. *Since $E(\mathbb{F}_p)$ has prime order, it is naturally protected against subgroup attacks (exploiting small prime divisors of the cofactor [132]). However, $\#E'(\mathbb{F}_{p^2}) = r(2p - r)$ may have small factors. For example, if $u = -2^{62} - 2^{55} - 1$ [155], $E'(\mathbb{F}_{p^2})$ is sensitive to subgroup attacks. As a protection, one can use (possibly expensive) membership tests [15] or choose u such that $2p - r$ is also prime.*

B.5.2 Choosing the generators

Let us construct generators P and Q of the input groups of the Ate pairing. Since $E(\mathbb{F}_p)$ has prime order, any point $P \neq O_E$ is suitable. The point Q must come from the twisted curve E', have order r in $E\left(\mathbb{F}_{p^{12}}\right)$ and, as a consequence, it satisfies $\pi(Q) = pQ$ [101]. It is provided by Algorithm 9.

Algorithme 9 : Generating Q

Data : p, r primes. The curve E such that $E(\mathbb{F}_p) = r$ and its $\mathbb{F}_{p^2}$ twist E'.
Result : A generator of the second input group of the Ate pairing

```
Q' ← O_E'
while Q' = O_E' do
    Pick a random point Q' ≠ O_E' in E'(F_p^2).
    if rQ' ≠ O_E' then
        Q' ← (2p − r) Q'   // Q' has now order r since #E'(F_p^2) = (2p − r)r
// Finally map Q' = (x, y) to E(F_p^12) thanks to the twist isomorphism
return Q = (x γ^2, y γ^3)   // with γ^6 = ξ
```

If one uses a well chosen family of curves, Q can be simpler to construct [87].

B.6 Choosing the system of coordinates

The Miller loop is a sequence of 2 operations involving $P = (x_P, y_P) \in E(\mathbb{F}_p)$ and $T = (x_T\gamma^2, y_T\gamma^3)$ with $x_T, y_T \in \mathbb{F}_{p^2}$, a multiple of Q in $E\left(\mathbb{F}_{p^{12}}\right)$:

- Doubling steps where we have to double T, compute the tangent line to E at T and evaluate this line at P.
- Addition steps where we have to add T and Q, compute the line passing through T and Q, and evaluate it at P.

These steps are performed with affine or projective coordinates (Jacobian ones are less efficient for pairing computations [46]). Let us give the formulas and their complexities to be able to choose the best ones depending on the context.

B.6.1 Affine coordinates

The slope of the line passing through T and Q is $\lambda\gamma$ where

$$\lambda = \frac{y_T - y_Q}{x_T - x_Q} \quad \left(\text{or } \lambda = \frac{3x_T^2}{2y_T} \text{ if } T = Q\right).$$

Then $T + Q$ (or $2T$) can be written in the form $(x_{T+Q}\gamma^2, y_{T+Q}\gamma^3)$ with

$$x_{T+Q} = \lambda^2 - x_T - x_Q \text{ and } y_{T+Q} = \lambda(x_T - x_{T+Q}) - y_T.$$

The equation of the line involved in the operation is $y = \lambda\gamma(x - x_T\gamma^2) - y_T\gamma^3$, thus the $\mathbb{F}_{p^{12}}$ element involved in the update of f in Algorithm 17 is

$$\ell = y_P - \lambda x_P\gamma + (\lambda x_T - y_T)\gamma^3.$$

If $-x_P$ is precomputed [92], the cost of the addition step is then $\mathbf{I}_2 + 3\mathbf{M}_2 + \mathbf{S}_2 + 2\mathbf{M}_1 + 7\mathbf{A}_2$ and the doubling step requires $\mathbf{I}_2 + 3\mathbf{M}_2 + 2\mathbf{S}_2 + 2\mathbf{M}_1 + 5\mathbf{A}_2 + 2\mathbf{A}'_2$. Note that, using some precomputations, $4\mathbf{M}_1$ can be saved in both cases [4].

Remark 8. *As λ is used 3 times in $\mathbb{F}_{p^2}$ operations ($\lambda^2, \lambda(x_T - x_{T+Q})$ and λx_T), $2\mathbf{A}_1$ can be saved by precomputing its trace if the Karatsuba/complex methods are used for $\mathbb{F}_{p^2}$ arithmetic. In the same way, x_T is used twice in the doubling step so that an additional $\mathbf{A}_1$ can be saved.*

B.6.2 Projective coordinates

In this case, the point T can be written $(X_T\gamma^2, Y_T\gamma^3, Z_T)$ with X_T, Y_T and $Z_T \in \mathbb{F}_{p^2}$. However, the point Q is kept in affine coordinates (mixed addition method). According to [46], $2T = (X_{2T}\gamma^2, Y_{2T}\gamma^3, Z_{2T})$ with

$$\begin{aligned} X_{2T} &= 2X_TY_T(Y_T^2 - 9b'Z_T^2) \\ Y_{2T} &= \left(Y_T^2 + 9b'Z_T^2\right)^2 - 12(3b'Z_T^2)^2 \\ Z_{2T} &= 8Y_T^3Z_T \end{aligned}$$

and the tangent to the curve at T evaluated in P is (up to some subfield factor)

$$\ell = 2y_P Y_T Z_T - 3x_P X_T^2 \gamma + \left(Y_T^2 - 3b' Z_T^2\right)\gamma^3.$$

If $-3x_P$ is precomputed [4], the doubling step then requires $2\mathbf{M}_2 + 7\mathbf{S}_2 + 4\mathbf{M}_1 + 13\mathbf{A}_2 + 5\mathbf{A}'_2 + \mathbf{m}_{2,b'}$ (using that $2X_TY_T = (X_T + Y_T)^2 - X_T^2 - Y_T^2$, which is always interesting over $\mathbb{F}_{p^2}$ because $\mathbf{S}_2$ is clearly cheaper than $\mathbf{M}_2$ according to Section B.3.5). In this case, precomputing traces has no interest.

In the same way, we compute the addition step with

$$\begin{aligned} N &= Y_T - y_Q Z_T,\ D = X_T - x_Q Z_T \quad (\text{so that } \lambda = \tfrac{N}{D}) \\ X &= D^3 + Z_T N^2 - 2X_T D^2 \\ X_{T+Q} &= DX, \quad Y_{T+Q} = N(X_T D^2 - X) - Y_T D^3, \quad Z_{T+Q} = D^3 Z_T \\ \ell &= y_P D - N x_P \gamma + (N x_Q - D y_Q)\gamma^3. \end{aligned}$$

Assuming that $-x_P$ is precomputed, it requires $11\mathbf{M}_2 + 2\mathbf{S}_2 + 4\mathbf{M}_1 + 7\mathbf{A}_2 + \mathbf{A}'_2$.

Remark 9. *Many $\mathbb{F}_{p^2}$ operands are used several times, so that precomputing their traces saves additions in $\mathbb{F}_p$. We do not give the details here because the addition step is rarely used in the Miller loop but it is not difficult to see that $16\mathbf{A}_1$ can be saved if Karatsuba/complex arithmetic is used for $\mathbb{F}_{p^2}$ arithmetic.*

B.6.3 Consequences of formulas

Several remarks can be made looking at these formulas. The first one is that b' is involved in only one $\mathbb{F}_{p^2}$ multiplication, so its influence is small. Hence the choice of u (following Section B.4) is a priority, even if a small b is not available.

The second one is that, as mentioned in Sections B.3.6 and B.3.7, the line ℓ is of the form $b_0 + b_1\gamma + b_3\gamma^3$ with $b_i \in \mathbb{F}_{p^2}$, thus it is sparse in $\mathbb{F}_{p^{12}}$.

The third one is that, as already mentioned in [1, 92, 127, 129], affine coordinates should be preferred in some contexts. Indeed, it is easy to verify that the affine coordinates are better than the projective ones for the doubling step (and then for the full Miller loop) as soon as

$$\mathbf{I}_1 < 5\mathbf{S}_2 - \mathbf{M}_2 - 2\mathbf{S}_1 + 15\mathbf{A}_1 + 6\mathbf{A}'_1 + \mathbf{m}_{2,b'} - \mathbf{m}_{1,\mu}.$$

Depending on the target device, this inequality may hold in practice, especially if $\mathbb{F}_p$ addition are not negligible. In Table B.9, we give the maximal cost of $\mathbf{I}_1$ for which affine coordinates should be preferred. To make the results more readable, we assumed that $\mathbf{S}_1 = \mathbf{M}_1, \mathbf{A}_1 = \mathbf{A}'_1$ and $\mathbf{m}_{2,b'}$ is minimal.

μ	Use affine coordinates if
-1	$\mathbf{I}_1 < 5\mathbf{M}_1 + 33\mathbf{A}_1$
-2	$\mathbf{I}_1 < 5\mathbf{M}_1 + 41\mathbf{A}_1$
-5	$\mathbf{I}_1 < 5\mathbf{M}_1 + 42\mathbf{A}_1$
any	$\mathbf{I}_1 < 13\mathbf{M}_1 + 28\mathbf{A}_1$

TABLE B.9 – Affine versus projective coordinates

B.7 Other algorithms for efficient implementation

As this paper may be used as a survey for pairing implementors, we describe here some algorithms which should not influence the choice of the parameters but which are important for the efficiency of pairing computation.

B.7.1 Frobenius computation

Let $a = b_0 + b_1\gamma + b_2\gamma^2 + b_3\gamma^3 + b_4\gamma^4 + b_5\gamma^5 \in \mathbb{F}_{p^{12}}$, where $b_j \in \mathbb{F}_{p^2}$ and $\gamma^6 = \xi \in \mathbb{F}_{p^2}$. In this section, we explain how to efficiently compute the Frobenius map, which consists in computing a^p, as well as its composites.

Computation of a^p

Since the Frobenius map is linear, we have

$$a^{p^i} = b_0^{p^i} + b_1^{p^i}\gamma^{p^i} + b_2^{p^i}\left(\gamma^2\right)^{p^i} + b_3^{p^i}\left(\gamma^3\right)^{p^i} + b_4^{p^i}\left(\gamma^4\right)^{p^i} + b_5^{p^i}\left(\gamma^5\right)^{p^i}$$

As $b_j \in \mathbb{F}_{p^2}$, b_j^p is the conjugate $\overline{b_j}$ of b_j. To compute the $\left(\gamma^j\right)^{p^i}$, we define

$$\begin{aligned}\delta &= \xi^{\frac{p-1}{6}} \in \mathbb{F}_{p^2} \text{ so that } \gamma^p = \left(\gamma^6\right)^{\frac{p-1}{6}}\gamma = \delta\gamma,\\ \omega &= \xi^{\frac{p^2-1}{6}} = N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(\delta) \in \mathbb{F}_p \text{ so that } \gamma^{p^2} = \omega\gamma,\end{aligned}$$

which can be precomputed and included in domain parameter. Then, for $1 \le i \le 11$, a^{p^i} can be simplified as

$$\begin{cases}\overline{b_0} + \overline{b_1}c_{i,1}\delta\gamma + \overline{b_2}c_{i,2}\delta^2\gamma^2 + \overline{b_3}c_{i,3}\delta^3\gamma^3 + \overline{b_4}c_{i,4}\delta^4\gamma^4 + \overline{b_5}c_{i,5}\delta^5\gamma^5 & \text{if } i \text{ is odd,}\\ b_0 + b_1c_{i,1}\gamma + b_2c_{i,2}\gamma^2 + b_3c_{i,3}\gamma^3 + b_4c_{i,4}\gamma^4 + b_5c_{i,5}\gamma^5 & \text{if } i \text{ even,}\end{cases}$$

where $c_{i,j} = \omega^{j\lfloor i/2\rfloor}$. Notice that ω is a primitive 6th root of unity (because ξ is neither a square nor a cube in $\mathbb{F}_{p^2}$). In particular, $\omega^2 - \omega + 1 = 0$ so that $c_{i,j} = 1$ (resp. ω, $\omega - 1$, -1, $-\omega$, $-\omega + 1$) if $j\lfloor i/2\rfloor = 0 \mod 6$ (resp. 1, 2, 3, 4, 5).

As a consequence, computing a^{p^i} when i is even requires one addition in $\mathbb{F}_p$ to compute $\omega - 1$ (it can be precomputed but the interest is really limited) and 4 multiplications of an element of $\mathbb{F}_{p^2}$ (b_j) by an element of $\mathbb{F}_p$ ($c_{i,j}$), thus the total cost is $8\mathbf{M}_1 + \mathbf{A}_1$ unless $i = 6$ which if for free (conjugation in $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$).

If i is odd, the $c_{i,j}\delta^j$ can be precomputed and then computing a^{p^i} requires $5\mathbf{M}_2$. In the extreme case where non-volatile memory would be a too scarce resource, the $c_{i,j}\delta^j$ can all be recovered from δ using less than $2\mathbf{M}_2 + 2\mathbf{S}_2 + 8\mathbf{M}_1 + \mathbf{A}_1$. In fact, this can be improved for particular choices of μ and ξ. Let us for example give the complexities to compute a^p in the two most interesting cases.

If $\mu = -1, \xi = 1 + \mathbf{i}$, we have $u = 7$ or $11 \bmod 12$, so $p = 7 \bmod 12$. Then

$$\delta = (1+\mathbf{i})^{\frac{p-1}{6}} = (1+\mathbf{i})^{2\frac{p-7}{12}}(1+\mathbf{i}) = (2\mathbf{i})^{\frac{p-7}{12}}(1+\mathbf{i}) = \mathbf{i}^{\frac{p-7}{12}}(1+\mathbf{i})2^{\frac{p-7}{12}},$$

thus we only need to precompute $d_1 = 2^{\frac{p-7}{12}} \in \mathbb{F}_p$ instead of $\delta \in \mathbb{F}_{p^2}$ and, if $d_2 = -2d_1^2$ and $d_3 = d_1 d_2$, it is easy to find that $(\delta, \delta^2, \delta^3, \delta^4, \delta^5)$ equals

$$\begin{cases} \left(\overline{\xi} d_1, \mathbf{i} d_2, \xi d_3, d_2 + 1, \overline{\xi}(d_1 + d_3)\right) & \text{if } u = 11, 19 \bmod 24, \\ \left(-\overline{\xi} d_1, \mathbf{i} d_2, m\xi d_3, d_2 + 1, -\overline{\xi}(d_1 + d_3)\right) & \text{if } u = 7, 23 \bmod 24. \end{cases}$$

It is not necessary to precompute d_2 and d_3 because their computation requires only $\mathbf{M}_1 + \mathbf{S}_1 + \mathbf{A}'_1$. Once they are computed, computing a^p requires 5 multiplications of an element of $\mathbb{F}_p$ by an element of $\mathbb{F}_{p^2}$, 2 additions ($1 + d_2$ and $d_1 + d_3$) and 3 multiplications by ξ or $\overline{\xi}$. Finally, assuming that d_1 is precomputed, computing a^p requires only $11\mathbf{M}_1 + \mathbf{S}_1 + 8\mathbf{A}_1 + \mathbf{A}'_1$.

If $\xi = \sqrt{\mu}$, u is even (otherwise $\mu = -1$ is chosen), so $p = 1 \bmod 12$. We have

$$\delta = \sqrt{\mu}^{\frac{p-1}{6}} = \mu^{\frac{p-1}{12}} \in \mathbb{F}_p.$$

Therefore, multiplications by δ and its powers by $\mathbb{F}_{p^2}$ elements cost only $2\mathbf{M}_1$ instead of $\mathbf{M}_2$. Finally, if δ is precomputed, computing a^p requires $11\mathbf{M}_1 + \mathbf{S}_1 + 2\mathbf{A}_1$ ($\mathbf{M}_1 + \mathbf{S}_1 + 2\mathbf{A}_1$ to compute the $\delta^2, \delta^3, \delta^4 = \delta^2 - 1$ and $\delta^5 = \delta^3 - \delta$ and $10\mathbf{M}_1$ to compute the $\overline{b_j}\delta^j$).

B.7.2 Cyclotomic squaring

The hard part of the final exponentiation takes place in the cyclotomic subgroup $G_{\Phi_6(p^2)}$ of $\mathbb{F}^*_{p^{12}}$ (Section B.2.4). Given $a \in \mathbb{F}_{p^{12}}$, the condition $a^{\Phi_6(p^2)} = 1$ combined with a description of the Frobenius action gives algebraic relations on the coordinates of a. Granger and Scott [91] first proposed to exploit these relations to compute a^2 (the main operation during the final exponentiation).
Furthermore, these relations can be used to compress the representation of a in such a way that a^2 can be directly computed in its compressed form. In particular, Karabina's method for compressed squaring [116, 3] costs $3\mathbf{S}_2 + 10\mathbf{A}_2 + \mathbf{m}_{2,\xi}$ less than Granger and Scott's method (in most cases). However, multiplications have to be performed in the non-compressed form and the decompression step is relatively expensive and requires an inversion in $\mathbb{F}_{p^2}$. Fortunately, there are few multiplication steps because the exponents have low Hamming weight (see Sections B.2.4 and B.4), and Montgomery's simultaneous inversion trick [145] can be used (we refer to [3] for more details). Therefore, if $\mathbb{F}_{p^2}$ inversions are available (at a reasonable cost), Karabina's method should be used for final exponentiation, otherwise, Granger and Scott's method should be chosen.

Granger and Scott's method

This method is given in [91] in the case of a $2,2,3$ tower but it can be easily adapted to the $2,3,2$ case.

Case $2,2,3$. With the notations of Section B.3.7, let $a = c_0 + c_1\gamma + c_2\gamma^2$ in $G_{\Phi_6(p^2)} \subset \mathbb{F}_{p^{12}} = \mathbb{F}_{p^4}[\gamma]$, then $a^2 = C_0 + C_1\gamma + C_2\gamma^2$ with [91]

$$C_0 = 3c_0^2 - 2\overline{c_0}, \quad C_1 = 3\beta c_2^2 + 2\overline{c_1}, \quad C_2 = 3c_1^2 - 2\overline{c_2},$$

where $\overline{u + v\beta} = u - v\beta$, for $u, v \in \mathbb{F}_{p^2}$. Therefore, the cost of squaring in $G_{\Phi_6(p^2)}$ is $3\mathbf{S}_4 + 3\mathbf{A}'_4 + 6\mathbf{A}_4 + \mathbf{m}_{4,\beta}$ (using that an expression of the form $3u + 2v = 2(u + v) + u$ can be computed with one doubling and 2 additions).

Case $2,3,2$. With the notations of Section B.3.6, let $a = (b_0 + b_2\beta + b_4\beta^2) + (b_1 + b_3\beta + b_5\beta^2)\gamma$ in $G_{\Phi_6(p^2)} \subset \mathbb{F}_{p^2}[\gamma]$, then $a^2 = (B_0 + B_2\beta + B_4\beta^2) + (B_1 + B_3\beta + B_5\beta^2)\gamma$ with

$$\begin{aligned} B_0 &= 3(b_3^2\xi + b_0^2) - 2b_0, & B_1 &= 3\xi((b_2 + b_5)^2 - b_2^2 - b_5^2)) + 2b_1, \\ B_2 &= 3(b_4^2\xi + b_1^2) - 2b_2, & B_3 &= 3((b_0 + b_3)^2 - b_0^2 - b_3^2)) + 2b_3, \\ B_4 &= 3(b_5^2\xi + b_2^2) - 2b_4, & B_5 &= 3((b_4 + b_1)^2 - b_4^2 - b_1^2)) + 2b_5. \end{aligned}$$

Then the cost of squaring in $G_{\Phi_6(p^2)}$ is $9\mathbf{S}_2 + 6\mathbf{A}'_2 + 24\mathbf{A}_2 + 4\mathbf{m}_{2,\xi}$.

Remark 10. *According to Table B.3,* $\mathbf{S}_4 = 3\mathbf{S}_2 + \mathbf{m}_{2,\xi} + 4\mathbf{A}_2$ *in the most common case and then it is easy to verify that the cost of squaring in the case* $2,2,3$ *is exactly the same as in the case* $2,3,2$ *so this has no consequence on the way to build* $\mathbb{F}_{p^{12}}$*.*

Karabina's method

This method is due to Karabina [116] and was slightly improved in [3]. Let $a = b_0 + b_1\gamma + b_2\gamma^2 + b_3\gamma^3 + b_4\gamma^4 + b_5\gamma^5 \in G_{\Phi_6(p^2)} \setminus \{1\}$, with $b_j \in \mathbb{F}_{p^2}$. The compressed representation of a is $[b_1, b_2, b_4, b_5]$. The full representation (decompression) of a is obtained via the formulas

$$\begin{aligned} b_3 &= \frac{b_5^2\xi + 3b_2^2 - 2b_4}{4b_1}, & b_0 &= (2b_3^2 + b_1b_5 - 3b_4b_2)\xi + 1, & &\text{if } b_1 \neq 0, \\ b_3 &= \frac{2b_2b_5}{b_4}, & b_0 &= (2b_3^2 - 3b_4b_2)\xi + 1, & &\text{if } b_1 = 0, \end{aligned}$$

and requires $\mathbf{I}_2 + 3\mathbf{M}_2 + 3\mathbf{S}_2 + 4\mathbf{A}'_2 + 6\mathbf{A}_2 + 2\mathbf{m}_{2,\xi} + \mathbf{A}_1$ (or $\mathbf{I}_2 + 3\mathbf{M}_2 + \mathbf{S}_2 + 2\mathbf{A}'_2 + 2\mathbf{A}_2 + \mathbf{m}_{2,\xi} + \mathbf{A}_1$ if $b_1 = 0$). The compressed representation of a^2 is given by

$$\begin{aligned} B_1 &= 2b_1 + 3(S_{2,5} - S_2 - S_5)\xi & B_4 &= 3(S_2 + S_5\xi) - 2b_4 \\ B_2 &= 3(S_1 + S_4\xi) - 2b_2 & B_5 &= 2b_5 + 3(S_{1,4} - S_1 - S_4), \end{aligned}$$

where $S_{i,j} = (b_i + b_j)^2$ and $S_i = b_i^2$, for a cost of $6\mathbf{S}_2 + 4\mathbf{A}'_2 + 16\mathbf{A}_2 + 3\mathbf{m}_{2,\xi}$.

B.8 Conclusion

We explained in details how to choose and generate parameters for the optimal Ate pairing on BN curves. It can be summarized by the following sequence :

1. Choose the parameter u using Algorithm 8 given in Section B.4.
2. Choose the way to build $\mathbb{F}_{p^{12}}$ depending on the value of u following Table B.4.
3. Choose the arithmetic of $\mathbb{F}_{p^{12}}$ depending on the relative cost of $\mathbb{F}_p$ operations on the targeted device as summarized in Section B.3.10 and Appendix B.10.
4. Generate the elliptic curve (and its twist) having the smallest coefficient using an incremental search as in Section B.5.1.
5. Find generators of the input groups of the Ate pairing using Algorithm 9.
6. Choose the system of coordinates to be used depending on the $\mathbf{I}_1/\mathbf{M}_1/\mathbf{A}_1$ ratios following Table B.9.

Because of [118], it will be necessary to generate new parameters for the 128 bits security level in the near future and there are many other situations where generating parameters is necessary (pairing parameters database, other security levels, resistance to subgroups attacks, trusted parameters, ...). We hope that this paper would help anyone interested in generating pairing parameters depending on the target device (security level, relative cost of $\mathbb{F}_p$ operations, memory resources, ...). In particular, our result on the way to choose the parameter u (which is a nice application of old elementary arithmetic results), should allow a faster generation. Note that all the method we described for BN curves could be adapted to other families of pairing friendly curves.
Moreover, we used this opportunity to give some new improvements on $\mathbb{F}_{p^{12}}$ arithmetic (in a pairing context) in terms of $\mathbb{F}_p$-addition because they are usually not so negligible for small devices. The theoretical gain obtained on $\mathbb{F}_{p^{12}}$ arithmetic is dependent on the context and on the $\mathbb{F}_{p^{12}}$ operation considered but we can say that the number of $\mathbb{F}_p$ additions is reduced by around 10% in most cases. This should be validated by an implementation but it was not the goal of this work.

B.9 Appendix 1 : Details for the new improvements of $\mathbb{F}_{p^{12}}$ arithmetic given in section B.3.9.

B.9.1 Multiplications by $\xi - 1$ in Karatsuba operations

The Karatsuba multiplication of $x_0 + x_1\beta + x_2\beta^2$ by $y_0 + y_1\beta + y_0\beta^2$ in $\mathbb{F}_{p^{3i}}$ can be evaluated as

$$\begin{aligned}
&x_0y_0 + \xi((x_1+x_2)(y_1+y_2) - x_1y_1 - x_2y_2)\\
&+[(x_0+x_1)(y_0+y_1) - x_0y_0 - x_1y_1 + x_2y_2 + (\xi-1)x_2y_2]\beta\\
&+[(x_0+x_2)(y_0+y_2) - x_0y_0 - (-x_1y_1 + x_2y_2)]\beta^2.
\end{aligned}$$

As in $\mathbb{F}_{p^{2i}}$, one of the multiplications by ξ of the formula given in B.3.3 is replaced by a multiplication by $\xi - 1$. Of course, this trick also applies to $\mathbb{F}_{p^{2i}}$ and $\mathbb{F}_{p^{3i}}$ Karatsuba squarings.
In the cases considered in Section B.3.8, this improvement is only interesting in the intermediate fields ($\mathbb{F}_{p^4}/\mathbb{F}_{p^2}$ or $\mathbb{F}_{p^6}/\mathbb{F}_{p^2}$) and allows to save some $\mathbb{F}_p$ additions for each Karatsuba multiplication or squaring in $\mathbb{F}_{p^4}$ or $\mathbb{F}_{p^6}$ in the cases given in Table B.10.

ξ	$m_{2,\xi-1}$	$m_{2,\xi}$	Saving
$1+\mathbf{i}$	0	$2\mathbf{A}_1$	$2\mathbf{A}_1$
$2+\mathbf{i}$	$2\mathbf{A}_1$	$2\mathbf{A}_1 + 2\mathbf{A}'_1$	$2\mathbf{A}'_1$
$3+\mathbf{i}$	$2\mathbf{A}_1 + 2\mathbf{A}'_1$	$4\mathbf{A}_1 + 2\mathbf{A}'_1$	$2\mathbf{A}_1$
$2+\sqrt{-2}$	$2\mathbf{A}_1 + \mathbf{A}'_1$	$2\mathbf{A}_1 + 2\mathbf{A}'_1$	$\mathbf{A}'_1$

TABLE B.10 – Savings provided by the $\xi - 1$ trick

Remark 11. *This trick is more interesting in the case* $2,2,3$ *that in the case* $2,3,2$. *Indeed, a Karatsuba multiplication in* $\mathbb{F}_{p^{12}}$ *requires* 6 *multiplications at the middle level in the case* $2,2,3$ *(and then* $6m_{2,\xi}$ *are replaced by* $6m_{2,\xi-1}$*) but only* 3 *in the case* $2,3,2$. *Of course, this remark also applies to sparse* $\mathbb{F}_{p^{12}}$ *multiplications and to* $\mathbb{F}_{p^{12}}$ *squarings.*

Let us now give the details of the saving obtained if traces are precomputed in the three situations described in section B.3.9.

B.9.2 Use of precomputed traces in $\mathbb{F}_{p^{12}}$ squarings

As explained in Section B.3.3, the Chung-Hasan method for $\mathbb{F}_{p^{3i}}$ squarings computes

$$x_1^2 + 2x_0x_2 = (x_0 + x_1 + x_2)^2 - (2x_0x_1 + 2x_1x_2 + x_0^2 + x_2^2).$$

Then $x_0, 2x_1$ and x_2 are each used in two $\mathbb{F}_{p^i}$ operations. If Karatsuba or complex arithmetic is used for these $\mathbb{F}_{p^i}$ operations, $3\mathbf{A}_{i/2}$ can then be saved by precomputing $x_0, 2x_1$ and x_2 traces. In fact, one can do even better, depending on the way to build $\mathbb{F}_{p^{12}}$.

Case 2,3,2. We saw in Section B.3.6 that a $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$ squaring is usually performed using the Karatsuba method :

$$(c_0 + c_1\gamma)^2 = c_0^2 + \left[(c_0 + c_1)^2 - c_0^2 - c_1^2\right]\gamma + c_1^2\beta.$$

Then the Chung-Hasan squaring in $\mathbb{F}_{p^6}$ is used 3 times. Moreover the $\mathbb{F}_{p^6}/\mathbb{F}_{p^2}$ traces of c_0, c_1 and $c_0 + c_1$ (which costs $2\mathbf{A}_2$ each) are necessary but the latter is the sum of the 2 others so that one additional $\mathbf{A}_2$ can be saved. Hence, if the Karatsuba method is used in $\mathbb{F}_{p^2}$, $11\mathbf{A}_1$ can be saved in $\mathbf{S}_{12}$ thanks to trace precomputations (9 from $\mathbb{F}_{p^6}/\mathbb{F}_{p^2}$ Chung-Hasan over Karatsuba squaring and 2 from $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$ Karatsuba over Chung-Hasan squaring). If the Karatsuba method is not used in $\mathbb{F}_{p^2}$, only the 2 last ones can be saved.

Case 2,2,3. We saw that computing $(c_0 + c_1\gamma + c_2\gamma^2)^2$ with the Chung-Hasan method, c_0, c_1 and $c_2 \in \mathbb{F}_{p^4}$ are used twice, so that precomputing their $\mathbb{F}_{p^4}/\mathbb{F}_{p^2}$ traces t_i saves $3\mathbf{A}_2$. But their $\mathbb{F}_{p^2}$ components are of course also used twice (in the same operations) and precomputing their $\mathbb{F}_{p^2}/\mathbb{F}_p$ traces is then interesting if the Karatsuba/complex method is used in $\mathbb{F}_{p^2}$. Moreover, in $\mathbb{F}_{p^4}$ operations, the t_i play the same role as the $\mathbb{F}_{p^2}$ components so that precomputing their traces is also interesting. Finally, if Karatsuba is used in $\mathbb{F}_{p^2}$, $15\mathbf{A}_1$ can be saved in $\mathbf{S}_{12}$ thanks to trace precomputations (6 from the $\mathbb{F}_{p^4}/\mathbb{F}_{p^2}$ traces of the c_i, 6 from the $\mathbb{F}_{p^2}/\mathbb{F}_p$ traces of the $\mathbb{F}_{p^2}$ components of the c_i and 3 from the $\mathbb{F}_{p^2}/\mathbb{F}_p$ traces of the $\mathbb{F}_{p^4}/\mathbb{F}_{p^2}$ traces of the c_i). If Karatsuba is not used in $\mathbb{F}_{p^2}$, only the first $6\mathbf{A}_1$ can be saved.

B.9.3 Use of precomputed traces in $\mathbb{F}_{p^{12}}$ sparse multiplications

The sparse multiplication involved in the Miller loop for the optimal Ate pairing involves schoolbook steps which will take advantage of precomputed traces. Again, the savings are dependent on the way to build $\mathbb{F}_{p^{12}}$.

Case 2,3,2. Looking at Formula (B.3.3) given in Section B.3.6, we can see that

- b_0 is used in 3 $\mathbb{F}_{p^2}$ multiplications. Precomputing its trace then saves $2\mathbf{A}_1$,
- b_1 and the third component of c_1 are used in 2 $\mathbb{F}_{p^2}$ multiplications during the sparse product $(b_1 + b_3\beta)c_1$, so $2\mathbf{A}_1$ can be saved,
- The same holds for the sparse product $(b_0 + b_1 + b_3\beta)(c_0 + c_1)$,
- b_3 is used twice in each of these sparse products, so $3\mathbf{A}_1$ can be saved by precomputing $tr_{\mathbb{F}_{p^2}/\mathbb{F}_p}(b_3)$.

Finally, $9\mathbf{A}_1$ can be saved in the sparse multiplication if traces are precomputed (assuming that the Karatsuba method is used for $\mathbb{F}_{p^2}$ multiplications)

Case 2,2,3. Looking at formula (B.3.4) given in Section B.3.7, we can see that

- $b_0 + b_3\beta$ is used in 2 $\mathbb{F}_{p^4}$ multiplications. Precomputing its trace $(b_0 + b_3)$ saves $\mathbf{A}_2$,
- As a consequence of the previous point, b_0, b_3 and $b_0 + b_3$ are used in 2 $\mathbb{F}_{p^2}$ multiplications, so $3\mathbf{A}_1$ can be saved,
- b_3 is also used in the $\mathbb{F}_{p^4}$ product $(c_0+c_1)(b_0+b_1+b_3\beta)$ which saves one additional $\mathbf{A}_1$,
- b_1 is used in 4 $\mathbb{F}_{p^2}$ multiplications (c_2b_1 and c_1b_1) so $3\mathbf{A}_1$ can be saved,
- c_2 is used twice (in c_2b_1 and in $c_2(b_0+b_3\beta)$) so its $\mathbb{F}_{p^2}$ coefficients are used twice each which saves $2\mathbf{A}_1$.

Finally, $11\mathbf{A}_1$ can be saved in the sparse multiplication if traces are precomputed (assuming that the Karatsuba method is used for $\mathbb{F}_{p^2}$ multiplications, otherwise only $2\mathbf{A}_1$ are saved).

In all considered cases, the saving obtained is around 10% of the total number of additions in $\mathbb{F}_{p^{12}}$ operations which is not negligible if the relative cost of an addition compared to a multiplication in $\mathbb{F}_p$ is not small.

B.9.4 Use of precomputed traces in the final exponentiation

Full multiplications in $\mathbb{F}_{p^{12}}$ are only used in the final exponentiation. If the implemented exponentiation parses the exponent from left to right (which is usually the case), then the multiplication steps are performed with one constant term c. Hence, we can precompute and store all the traces depending only on c. Since the Karatsuba method is used at all levels of the extension tower (except in $\mathbb{F}_{p^2}$ if $\mathbf{A}_1 > 0.33\mathbf{M}_1$), we will significantly reduce the number of required additions, whatever the way to build $\mathbb{F}_{p^{12}}$.

Case 2,3,2.

- $\mathbf{M}_{12}^K$ requires the $\mathbb{F}_{p^{12}}/\mathbb{F}_{p^6}$ trace of c, thus $\mathbf{A}_6$ can be saved if this trace is precomputed. It also requires $3\mathbf{M}_6^K$ by a constant term (the 2 coordinates of c and its trace).
- Each $\mathbf{M}_6^K$ involving a constant term b requires 3 sums of 2 coordinates of b, thus $3\mathbf{A}_2$ can be saved if these sums are precomputed. Hence $9\mathbf{A}_2$ are saved at this level. Each $\mathbf{M}_6^K$ also requires $6\mathbf{M}_2$ by a constant term (the 3 coordinates of b and the 3 sums of 2 coordinates).
- Each $\mathbf{M}_2^K$ involving a constant term a requires the $\mathbb{F}_{p^2}/\mathbb{F}_p$ trace of a, thus one $\mathbf{A}_1$ can be saved if this trace is precomputed. Hence $18\mathbf{A}_1$ can be saved at this level when the Karatsuba method is used for $\mathbf{M}_2$.

Case 2,2,3.

- $\mathbf{M}_{12}^K$ requires 3 sums of 2 coordinates of c, thus $3\mathbf{A}_4$ can be saved if these sums are precomputed. It also requires $6\mathbf{M}_4^K$ by a constant term (the 3 coordinates of c and the 3 sums of 2 coordinates).
- Each $\mathbf{M}_4^K$ involving a constant term b requires the $\mathbb{F}_{p^4}/\mathbb{F}_{p^2}$ trace of b, thus one $\mathbf{A}_2$ can be saved if this trace is precomputed. Hence $6\mathbf{A}_2$ are saved at this level. Each $\mathbf{M}_4^K$ also requires $3\mathbf{M}_2$ by a constant term (the 2 coordinates of b and its trace).
- Again, one $\mathbf{A}_1$ can be saved for each $\mathbf{M}_2^K$ involving a constant term if its trace is precomputed. Hence $18\mathbf{A}_1$ can be saved at this level when Karatsuba is used for $\mathbf{M}_2$.

In both cases, $42\mathbf{A}_1$ can be saved for each multiplication in $\mathbb{F}_{p^{12}}$ involving a constant term if its traces are precomputed. This is about 20% of the total number of additions in $\mathbf{M}_{12}$ which is significant if the relative cost of an addition compared to a multiplication in $\mathbb{F}_p$ is not small. If $\mathbf{A}_1 > 0.33\mathbf{M}_1$, the schoolbook method is used for $\mathbb{F}_{p^2}$ multiplication and only $24\mathbf{A}_1$ can be saved.

B.10 Appendix 2

In this appendix, we give tables summarizing complexities for $\mathbb{F}_{p^{12}}$ arithmetic if $\mu \neq -1$ (see Section B.3.10 for more details).

Operation	Case	Arithmetic used				Number of operations without/with improvement				
						$\mathbf{M}_1$	if $\xi = \sqrt{-2}$		if $\xi = 2+\sqrt{-2}$	
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$		$\mathbf{A}_1$	$\mathbf{A}'_1$	$\mathbf{A}_1$	$\mathbf{A}'_1$
$\mathbf{M}_{12}$	2,3,2	K		K	K	54	210/168	25	224/182	32/29
	2,2,3		K				210/168	26	226/184	34/28
$\mathbf{sM}_{12}$	2,3,2	K		K	K	39	115/106	16	121/112	19
	2,2,3		K				117/106	17	125/114	21/18
$\mathbf{S}_{12}$	2,3,2	K/$\mathbb{C}$		CH	K	36	129/118	37	143/132	44
	2,2,3		K		CH		133/118	35	147/132	42/37

TABLE B.11 – $\mathbb{F}_{p^{12}}$ complexities if $\mu = -2$ (assuming $\mathbf{A}_1 \leq 0.33\mathbf{M}_1$ if our improvements are not used).

Operation	Case	Arithmetic used				Number of operations				
						$\mathbf{M}_1$	if $\xi = \sqrt{-2}$		if $\xi = 2+\sqrt{-2}$	
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$		$\mathbf{A}_1$	$\mathbf{A}'_1$	$\mathbf{A}_1$	$\mathbf{A}'_1$
$\mathbf{M}_{12}$	2,3,2	SB		K	K	72	156	25	170	32
	2,2,3		K				156	26	172	34
$\mathbf{sM}_{12}$	2,3,2	SB		K	K	52	76	16	82	19
	2,2,3		K				78	17	86	21
$\mathbf{S}_{12}$	2,3,2	SB		K	$\mathbb{C}$	48	108	24	120	30
$\mathbf{s}_1=\mathbf{M}_1$	2,2,3		K/$\mathbb{C}$		CH		106	32	126	42
$\mathbf{S}_{12}$	2,3,2	SB		CH	K	47.4	93	37	107	44
$\mathbf{s}_1=0.8\mathbf{M}_1$	2,2,3		K		CH		97	35	111	40

TABLE B.12 – $\mathbb{F}_{p^{12}}$ complexities if $\mu = -2$ if our improvements are not used and $\mathbf{A}_1 > 0.33\mathbf{M}_1$.

Operation	Case	Arithmetic used				Number of operations without/with improvement		
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$	$\mathbf{M}_1$	$\mathbf{A}_1$	$\mathbf{A}'_1$
$\mathbf{M}_{12}$	$2,3,2$	K		K	K	54	225/183	50
	$2,2,3$		K				226/184	52
$\mathbf{sM}_{12}$	$2,3,2$	K		K	K	39	131/122	32
	$2,2,3$		K				134/123	34
$\mathbf{S}_{12}$	$2,3,2$	K/$\mathbb{C}$		CH	K	36	151/140	50
	$2,2,3$		K		CH		155/140	48

TABLE B.13 – $\mathbb{F}_{p^{12}}$ complexities if $\mu = -5$ (assuming $\mathbf{A}_1 \leq 0.33\mathbf{M}_1$ if our improvements are not used).

Operation	Case	Arithmetic used				Number of operations		
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$	$\mathbf{M}_1$	$\mathbf{A}_1$	$\mathbf{A}'_1$
$\mathbf{M}_{12}$	$2,3,2$	SB		K	K	72	181	50
	$2,2,3$		K				182	52
$\mathbf{sM}_{12}$	$2,3,2$	SB		K	K	52	92	32
	$2,2,3$		K				95	34
$\mathbf{S}_{12}$	$2,3,2$	SB/$\mathbb{C}$		CH	K	42	133	50
	$2,2,3$		K		CH		137	48

TABLE B.14 – $\mathbb{F}_{p^{12}}$ complexities if $\mu = -5$ if our improvements are not used and $\mathbf{A}_1 > 0.33\mathbf{M}_1$.

Operation	Case	Arithmetic used				Number of operations without/with improvement		
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$	$\mathbf{M}_1$	$\mathbf{A}_1$	$\mathbf{A}'_1$
$\mathbf{M}_{12}$	$2,3,2$	K		K	K	79	210/168	0
	$2,2,3$		K			80	210/168	
$\mathbf{sM}_{12}$	$2,3,2$	K		K	K	55	115/106	0
	$2,2,3$		K			56	117/106	
$\mathbf{S}_{12}$	$2,3,2$	K		K	$\mathbb{C}$	54	144/133	6
	$2,2,3$		K/$\mathbb{C}$		CH	58	142/127	10

TABLE B.15 – $\mathbb{F}_{p^{12}}$ complexities if μ is large (assuming $\mathbf{A}_1 \leq 0.33\mathbf{M}_1$ if our improvements are not used).

Operation	Case	Arithmetic used				Number of operations		
		$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^{12}}$	$\mathbf{M}_1$	$\mathbf{A}_1$	$\mathbf{A}'_1$
$\mathbf{M}_{12}$	$2,3,2$	SB		K	K	97	156	0
	$2,2,3$		K			98		
$\mathbf{sM}_{12}$	$2,3,2$	SB		K	K	68	76	0
	$2,2,3$		K			69	78	
$\mathbf{S}_{12}$	$2,3,2$	SB		K	$\mathbb{C}$	66	108	6
$\mathbf{s}_{1=\mathbf{M}_1}$	$2,2,3$		K/$\mathbb{C}$		CH	70	106	10
$\mathbf{S}_{12}$	$2,3,2$	SB		K	$\mathbb{C}$	66	108	6
$\mathbf{s}_{1=\mathbf{M}_1}$	$2,2,3$		K/SB		CH	73.6	82	16

TABLE B.16 – $\mathbb{F}_{p^{12}}$ complexities if μ is large if our improvements are not used and $\mathbf{A}_1 > 0.33\mathbf{M}_1$.

Annexe C

A scalable and systolic architectures of montgomery modular multiplication for public key cryptosystems based on dsps

HASS 2017 [150]
avec A. Mrabet, R. Lashermes, J-B. Rigaud, B. Bouallegue, S. Mesnager and M. Machhout[1]

Abstract The arithmetic in a finite field constitutes the core of Public Key Cryptography like RSA, ECC or pairing-based cryptography. This paper discusses an efficient hardware implementation of the Coarsely Integrated Operand Scanning method (CIOS) of Montgomery modular multiplication combined with an effective systolic architecture designed with a Two-dimensional array of Processing Elements. The systolic architecture increases the speed of calculation by combining the concepts of pipelining and the parallel processing into a single concept. We propose the CIOS method for the Montgomery multiplication using a systolic architecture. As far as we know this is the first implementation of such design. The proposed architectures are designed for Field Programmable Gate Array platforms. They targeted to reduce the number of clock cycles of the modular multiplication. The presented implementation results of the CIOS algorithms focuses on different security levels useful in cryptography. This architecture have been designed in order to use the flexible DSP48 on Xilinx FPGAs. Our architecture is scalable and depends only on the number and size of words. For instance, we provide results of implementation for 8, 16, 32 and 64 bit long words in 33, 66, 132 and 264 clock cycles. We highlight the fact that for a given number of word, the number of clock cycles is constant.

[1]This work was supported in part by French project ANR-12-INSE-0014 "SIMPATIC".

C.1 Introduction

Since 1976, many Public Key Cryptosystems (PKC) have been proposed and all these cryptosystems based their security on the difficulty of some mathematical problem. The hardness of this underlying mathematical problem is essential for security. Elliptic Curve Cryptosystems which were proposed by Koblitz [121] and Miller [142], RSA [165] and the Pairing-Based Cryptography[113] are examples of PKCs. All these systems rely on an efficient finite field multiplication. As a consequence, the development of efficient architecture for modular multiplication has been a very popular subject of research. In 1985, Montgomery has presented a new method for modular multiplication [147]. It's one of the most suitable algorithm for performing modular multiplications in hardware and software implementations. The efficient implementation of the Montgomery modular multiplication in hardware was considered by many authors [157, 109, 98, 104, 161, 178]. There are a variety of ways to perform the Montgomery multiplication, considering if multiplication and reduction are separated or integrated. The separated approach consists in first performing the product and then the Montgomery reduction. It was presented in 1996 by Koç and Tolga in [123]. This method is called the Separated Operand Scanning method (SOS). On the contrary, the integrated approach is characterized by an alternation between multiplication and reduction. Several integrated approaches are presented in [123] : the Coarsely Integrated Operand Scanning Method (CIOS), the Finely Integrated Operand Scanning Method (FIOS), the Finely Integrated Product Scanning Method (FIPS) and the Coarsely Integrated Hybrid Scanning Method (CIHS). According to Koç and Tolga in [123] the CIOS method is a scalable word-based method for Montgomery multiplication, and it is the most efficient algorithm that integrates the multiplication with reduction steps. A systolic array architecture [124, 184] is one possibility for the implementation of the Montgomery algorithm in hardware [178, 161, 157, 98]. These architectures offer Processing Elements (PE) array where each Processing Element performs arithmetic computation additions and multiplications. In accordance with the number of words used, the architecture can employ a variable number of PEs. The systolic architecture uses very simples Processing Elements. As a consequence, the systolic architecture decreases the needs for logic elements in hardware implementations. Our contribution in this work is to combine a systolic architecture, which is assumed to be the best choice for FPGA implementation, with the CIOS method of Montgomery modular multiplication. We optimize the number of clock cycles required to compute a n-bit Montgomery multiplication and we reduce the utilization of FPGA resources. We have implemented the modular multiplication in a fixed number of clock cycles. To the best of our knowledge, this is the first time that a hardware or a software multiplier of modular Montgomery multiplication, suitable for various security level, is performed in just 33 clock cycles. Furthermore, as far as we know, our work is the first one dealing with systolic architecture and CIOS method over large prime characteristic finite fields. This paper is organized as follows : Section C.2 discusses related state-of-the-art works. Section C.3 presents the Montgomery modular multiplication algorithm. The proposed architectures and results are presented in Section C.4 and Section C.5. Finally, the conclusion is presented in Section C.6.

C.2 Brief state of the art

In hardware design, the systolic architecture [124] is a pipelined network arrangement of Processing Elements (or cells). It is a specialized form of parallel design. Each cell compute the data which is coming as input and calculate data independently. In [184] the authors proposed a systolic design for FPGA implementation. Several works are devoted to the implementation of the Montgomery multiplication [79, 123, 157, 147, 161, 98, 104, 108, 109, 178]. The first ones to our knowledge who proposed a systolic array are Iwamura, Matsumoto and Imai [108, 109]. They presented a systolic architecture that can execute a modular exponentiation using Montgomery multiplications. In [178] Tenca and Koç introduced a pipelined Montgomery modular multiplication, which has the ability to work in any given operand precision and which is adjustable to any chip area. Harris et al. in [99] improve the result of [178] using a systolic architecture for the Montgomery multiplication. Siddika Berna Örs, Lejla Batina, Bart Preneel and Joos Vandewalle presented in [157] a modular exponentiation based on the modular Montgomery. In [161] Guilherme Perin, Daniel Gomes Mesquita and Jõao Baptista Martins proposed a comparison between two modular multiplication architectures : a systolic and a very high-radix multiplexed implementation. Their approach uses a radix-16 and radix-32 decomposition. Both implementations targeted a Virtex-4 and a Virtex-5 FPGA. (A radix-n word is a word of size n.) Their work is the latest and the most efficient describing the use of a systolic approach for the Montgomery multiplication. We briefly recall the definition of a systolic architecture before a summary of their work. A systolic architecture is a pipelined network arrangement of PEs called cells. It is a specialized form of parallel computing, where cells compute the data which is coming as input and store them independently. A systolic architecture is an array composed of matrix-like rows of cells. Each PE shares the information with its neighbours immediately after processing. Cell at each step takes input data from one or more neighbours. The systolic architecture proposed in the work [161] is composed of s Processing Elements distributed in a one-dimensional array. The number s is the number of words. At each iteration of the Montgomery Algorithm, the words are read from an external memory (BRAM) and passed to their architecture. To evaluate the number of clock cycles for a Montgomery multiplication in the systolic architecture, they have to consider the first s cycles to read the input operands from RAM memories. Furthermore the first iteration of algorithm also needs s clock cycles. Finally the remaining iterations of algorithm are performed in $4 \times s$ clock cycles. As a consequence, this architecture requires a $6 \times s (= s + s + 4 \times s)$ clock cycles. For the multiplexed architecture, the first steps are identical to thus of the systolic architecture $(2 \times s)$. The number of clock cycles required to remaining iterations of Montgomery Algorithm is $6 \times s$ clock cycles. In order to perform the multiplexed architecture the algorithm requires $8 \times s (= 2 \times s + 6 \times s)$ clock cycles.

C.3 Montgomery Multiplication

The Montgomery Multiplication Algorithm for large prime characteristic finite fields [147] is a method for performing modular multiplication without needing to divide by

Algorithme 10 : Montgomery Modular Multiplication

Input : p an odd prime, $n = \lceil log_2(p) \rceil$, $R = 2^n$, $p' = -p^{-1} mod\ R$, $M(a)$, $M(b) \in \mathbb{F}_p$

Output : $M(ab) \mod p$

1 $\gamma \leftarrow M(a) \times M(b)$
2 $\delta \leftarrow \gamma \times p' \mod R$
3 $T \leftarrow \frac{\gamma + \delta \times p}{R}$
4 **If** $T \geq p$ **then** $T \leftarrow T - p$
5 **return** $\underline{T}$

the modulus. In cryptography, the Montgomery Algorithm is the most used modular multiplication to perform the operation $a \times b \mod p$. The Montgomery multiplication transforms the division by p into several divisions by a power of 2, which consists only in shifts in hardware and software implementation. Furthermore, the Montgomery multiplication among large numbers can be constructed using a radix representation of the numbers. Let p be an odd prime number. Let $n = \lceil log_2(p) \rceil$ be the length of the binary decomposition of p. We choose the base of numeration to be $R = 2^n$, such that $p < R$. As p and R are coprime, we can define $p' = -p^{-1} \mod R$. The choice of R is motivated by the facts that $\gcd(R, p) = 1$ and reductions and divisions by R must be efficient. As R is a power of 2, divisions are right shifts and the modulo operation is a simple assignment of the first n-bit. Montgomery multiplication is performed with numbers represented in the Montgomery representation. The conversion from ordinary domain to Montgomery domaine detailed in Table C.1 The map $M : a \in$

Ordinary Domain	$\longleftrightarrow$	Montgomery Domain
a	$\longleftrightarrow$	M(a)=a·R mod p
b	$\longleftrightarrow$	M(b)=b·R mod p
a·b	$\longleftrightarrow$	M(a·b)=a·b·R mod p

TABLE C.1 – Conversion between Montgomery and Ordinary Domains

$\mathbb{F}_p \to aR \in \mathbb{F}_p$ is a bijection and a field isomorphism of $\mathbb{F}_p$. For any element a of $\mathbb{F}_p$, the product $aR \in \mathbb{F}_p$ is called the Montgomery representation of a in basis R and it is denoted $M(a)$. We describe the Montgomery multiplication in Algorithm 10. The Montgomery multiplication computes $M(a) \times M(b)$ and gives as result $M(ab)$.

C.3.1 CIOS Method

The Coarsely Integrated Operand Scanning (CIOS) method presented in Algorithm 11, improves the Montgomery Algorithm by integrating the multiplication

and reduction. More specifically, instead of computing the product $a \cdot b$, then reducing the result, this method allows an alternation between iterations of the outer loops for multiplication and reduction. The integers (p, a and b) are seen as lists of s words of size w. In order to perform this algorithm we need an array T of size only $s+2$. The intermediate results are stored in T. The final result of the CIOS algorithm is composed by the $s+1$ least significant words of this array. The alternation

Algorithme 11 : CIOS algorithm for Montgomery multiplication [123]

Input : $p < 2^K$, $p' = -p^{-1} mod\ 2^w$, w, s , $K = s \cdot w$:bit length, $R = 2^K$, $a, b < p$
Output : $a \cdot b \cdot R^{-1}$ mod p

```
1  T ← Null;
2  for i ← 0 to s − 1 do
3      C ← 0;
4      for j ← 0 to s − 1 do
5          (C, S) ← T[j] + a[i] · b[j] + C
6          T[j] ← S
7      (C, S) ← T[s] + C
8      T[s] ← S
9      T[s + 1] ← C
10     C ← 0;
11     m ← T[0] · p' mod 2^w
12     (C, S) ← T[0] + m · p[0]
13     for j ← 1 to s − 1 do
14         (C, S) ← T[j] + m · p[j] + C
15         T[j] ← S
16     (C, S) ← T[s] + C
17     T[s − 1] ← S
18     T[s] ← T[s + 1] + C
19 return T;
```

between multiplication and reduction is possible since the value of m (in line 11 of the Algorithm 11) in the i^{th} iteration of the outer loop for reduction depends only on the value $T[j]$, which is computed by the i^{th} iteration of the outer loop for the multiplication. In order to perform the multiplication, we have modified the CIOS algorithm of [123] and designed this method with a systolic architecture. Indeed, instead of using an array to store the intermediate result, we replace T by Input and Output signals for each Processing Element. As a consequence, our design uses fewer of multiplexers and then we have better results considering the number of slices.

C.4 Hardware Implementation

C.4.1 Block DSP in Xilinx FPGAs

Modern FPGA devices like Xilinx Virtex-4, Virtex-5 and Artix-7 as well as Altera Stratix FPGAs have been equipped with arithmetic hardcore extensions to accelerate

digital signal processing applications. These function DSP blocks can be used to build a more efficient implementation interms of performance and reduce at the same time the demand for areas. DSP blocks can be programmed to perform basic arithmetic functions, multiplication, addition and subtraction of unsigned integers. Figure C.1 shows the generic DSP structure in advanced FPGAs. DSP can operate on external Input A,B and C as well as on feedback values from P or result PCIN.

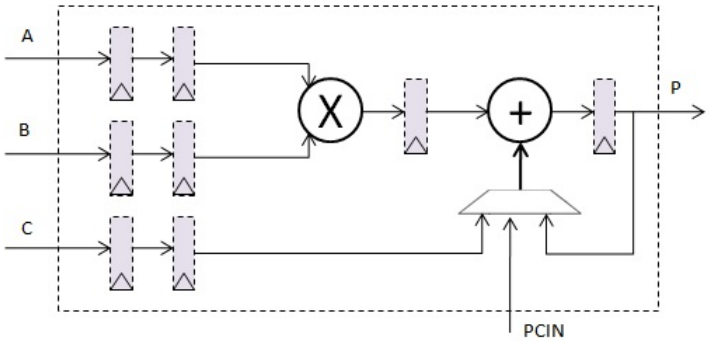

FIGURE C.1 – Structure of DSP block in modern FPGA device.

C.4.2 Proposed Architecture

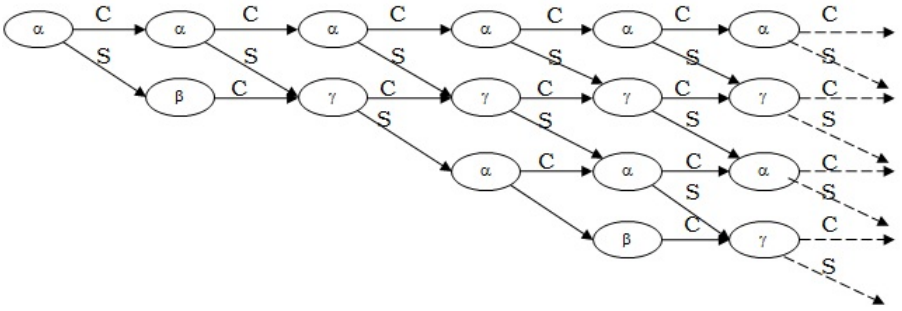

FIGURE C.2 – data dependency in general systolic architecture.

The idea of our design is to combine the CIOS method of Montgomery Modular multiplier presented in [123] with a two-dimensional systolic architecture in the model of [105, 184]. As seen in section C.3.1, the CIOS method is an alternation between iterations of the loops for multiplication and reduction. The concept of the two-dimensional systolic architecture presented in Section C.2 combines an identical Processing Elements with local connections, which take external inputs and handle them with a predetermined manner in a pipelined fashion. This new architecture is directly based on the arithmetic operations of the CIOS method of Montgomery Algorithm. The arithmetic is performed in a radix-w base (2^w). The input operands are processed in s words of w bits. We present many versions of this method. We illustrate our design for $s = 8$, $s = 16$, $s = 32$ and a $s = 64$ architectures, respectively denoted NW-8 (for Number of Words), NW-16, NW-32 and NW-64. Before the descriptions of the architectures NW-8 and NW-16, we begin with a generic description

of our systolic architecture. Our proposed architectures for the implementation of the Montgomery modular multiplication is detailed in this section. We describe it in detail as well as the different Processing Element behaviours. In order to have less of states in our Final State Machine (FSM), we divided our Algorithm 11 of Montgomery on five kinds of PE noted :

- cells alpha denoted α ;
- cells beta denoted β ;
- cells gamma denoted γ ;
- cells alpha final denoted α_f ;
- cells gamma final denoted γ_f.

Figure C.2 presents the dependency of the different cells. Below we describe precisely each cells. The letters MSB stand for the Most Significant Bits and LSB for the Least Significant Bits. In our notation the letter C denote the MSB of the results and the letter S the LSB.

1. alpha : Presented by the lines 4 and 5 in the Algorithm 11 and detailed in Algorithm 12 . The PEs alpha are scalable according to the NW in the design. We use this cell to perform the multiplication step. The input of the cell alpha are : S_In provided by the previous step, C_In provided by the previous step, $a[i]$: The words of the operand a, and $b[j]$: The words of the operand b. The output of the cell alpha are : S provided to the next step and C provided to the next step.

2. beta : Presented by the lines 9, 10 and 11 in the Algorithm 11 and detailed in Algorithm 13. The input of the cell beta are : S_In provided by the previous step, $p[0]$: The first word of the modulo p and p' : predefined. The output of the cell beta are : m provided to the next step and C provided to the next step.

3. gamma : Presented by the lines 13 and 14 in the Algorithm 11 and detailed in Algorithm 14. The PEs gamma are scalable according to the NW in the design. We use this cell to perform the reduction step. The input of the cell gamma are : S_In provided by the previous step, C_In provided by the previous step, $p[j]$: The words of the modulo p and m provide by the cell beta. The output of the cell gamma are : S provided to the next step and C provided to the next step.

4. alpha_final : Presented by the lines lines 6, 7 and 8 in the Algorithm 11 and detailed in Algorithm 15. The input of the cell alpha_final are : S_In provided by the previous step and C_In provided by the previous step. The output of the cell alpha_final are : $S1$ provided to the next step and $S2$ provide to the next step.

5. gamma_final : Presented by the lines 15, 16 and 17 in the Algorithm 11 and detailed in Algorithm 16. The input of the cell gamma_final are : $S1\_In$ provided by the previous step, $S2\_In$ provided by the previous step and C_In provided by the previous step. The output of the cell gamma_final are : $S1$ provided to the next step and $S2$ provided to the next step.

Algorithme 12 : Cell alpha

Input : $a[i]$, $b[j]$, C_In, S_In
Output : C, S
1 $tmp1 \leftarrow S\_In + C\_In$
2 $tmp2 \leftarrow a[i] \cdot b[j]$
3 $tmp2 \leftarrow tmp2 + tmp1$
4 $C \leftarrow$MSB($tmp2$)
5 $S \leftarrow$LSB($tmp2$)
6 **return** $\underline{C, S}$;

Algorithme 13 : Cell beta

Input : S_in, $p[0]$, $p' = -p^{-1} mod\ 2^w$
Output : C, m
1 $tmp1 \leftarrow S\_in \cdot p'$
2 $m \leftarrow LSB(tmp1)$
3 $tmp1 \leftarrow p[0] \cdot m$
4 $tmp1 \leftarrow S\_in + tmp1$
5 $C \leftarrow$MSB($tmp1$)
6 **return** $\underline{C, m}$;

Algorithme 14 : Cell gamma

Input : $p[i]$, m, C_in, S_in
Output : C, S
1 $tmp1 \leftarrow S\_in + C\_in$
2 $tmp2 \leftarrow p[i] \cdot m$
3 $tmp2 \leftarrow tmp2 + tmp1$
4 $C \leftarrow$MSB($tmp2$)
5 $S \leftarrow$LSB($tmp2$)
6 **return** $\underline{C, S}$;

This organization allows us to optimize the number of clock cycles. Each Processing Element in Figure C.11 is responsible for performing arithmetic operations. The different Processing Elements establish communication with the control block (FSM) as shown in Figure C.10 by receiving starts signals at each state of Montgomery Algorithm iteration. Each PE sends a done signal to the FSM at each end of the calculation. The final result is a concatenation of the last output of gamma and gamma_final PEs. The structure of all PEs have a combinational behaviour.

Algorithme 15 : Cell alpha_final

Input : C_in, S_in
Output : $S1$, $S2$
1 $tmp1 \leftarrow S\_in + C\_in$
2 $S1 \leftarrow$LSB($tmp1$)
3 $S2 \leftarrow$MSB($tmp1$)
4 **return** $\underline{C, S}$;

Algorithme 16 : Cell gamma_final

Input : C_in, $S1\_in$, $S2\_in$
Output : $S1$, $S2$
1 $tmp1 \leftarrow S1\_in + C\_in$
2 $S1 \leftarrow$LSB($tmp1$)
3 $S2 \leftarrow$MSB($tmp1$)
4 $S2 \leftarrow S2\_in + S2$
5 **return** $\underline{S1, S2}$;

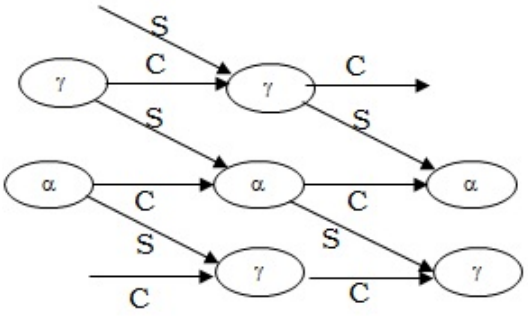

FIGURE C.3 – PEs of Systolic Architecture in two-dimensional array.

C.4.3 Internals architectures of cells

In this section we will describe the internals architectures of PEs used in these designs. Our five cells are designed in order to use DSP(s) blocks.

Description of the cell α

As illustrated in Figure C.4, the multiplication between $a[i]$ and $b[j]$ words returns a $2w$ bits result. This result is added thereafter to S_α_In. This latter is the least significant bits of the result of Processing Element gamma, which is provided through the output multiplexer. The last add is also added to C_α_In. The C_α_In is the most significant bits of the result of the previous Processing Element alpha, which is provided also through an output of a second multiplexer. The different inputs outputs of the PE alpha are presented in Figure C.10. The most significant bits of the result of alpha is propagated to the multiplexer to fix the next PE of alpha. Whereas the least significant bits are propagated to an other multiplexer to fix the next PE of gamma. After each computation of the alpha PE a shift in the input b

is triggered.

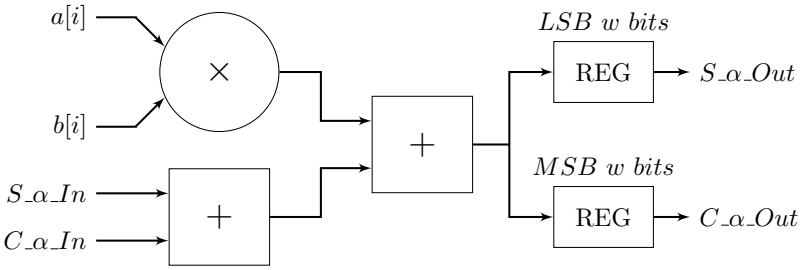

FIGURE C.4 – Alpha Processing Element internal architecture.

Description of the cell β

According to our algorithm 13 and as illustrated in Figure C.5, the zero index word of p ($p[0]$) and p' are provided to this beta Processing Element. The number p' corresponds the modular inverse of p modulo 2^w. The multiplication between p' and S_β_In returns a $2w$ bits result, where only the least significant bits of this multiplication is multiplied by the first word of p and returns a $2w$ bits result. Finally, this result is added to a w bits word S_β_In. Only the most significant bit part of this result is used in the next gamma PE. The different inputs/outputs of PE beta are presented in Figure C.10.

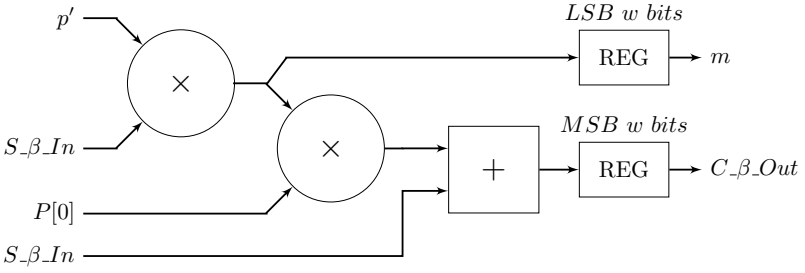

FIGURE C.5 – Beta Processing Element internal architecture.

Description of the cell γ

As illustrated in Figure C.6, the multiplication between m and $p[j]$ words returns a $2w$ bits result. This latter is added thereafter to S_γ_In. The number S_γ_In corresponds to the least significant bits of the result of Processing Element alpha, which is provided through an output multiplexer. This add is also added to C_γ_In, where C_γ_In is the most significant bits of the result of the previous Processing Element gamma. This PE gamma is provided also through an output of a second multiplexer. The different inputs/outputs of the gamma PE are shown in Figure C.10. The most significant bits of result are propagated to the multiplexer to fix the next PE of gamma. Whereas the least significant bits are propagated to an other multiplexer to fix the next PE of alpha.

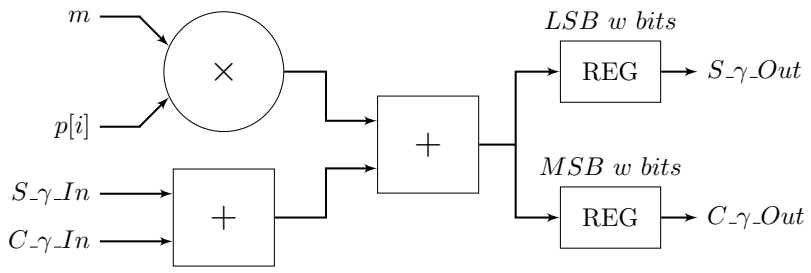

FIGURE C.6 – Gamma Processing Element internal architecture.

Description of the cell α_f

The cell α_f corresponds to the final α computed at the end of the line correspond to the multiplication step. In the PE alpha_final, the $S\_\alpha\_f\_In$ added to C_α_f returns a $2w$ bits result as presented in Figure C.7.

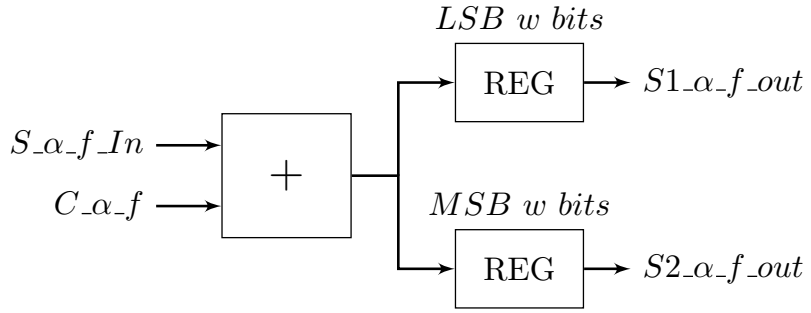

FIGURE C.7 – Alpha_f Processing Element internal architecture.

Description of the cell γ_f

The cell γ_f corresponds to the final γ computed at the end of the line correspond to the reduction step. For Processing Element gamma_final, $S1\_\gamma\_f\_In$ is added to C_γ_f, the result is a $2w$ bits. The least significant bits of the last result is added to $S2\_\gamma\_f\_In$. The internal architecture of the gamma_final type PE is presented in Figure C.8.

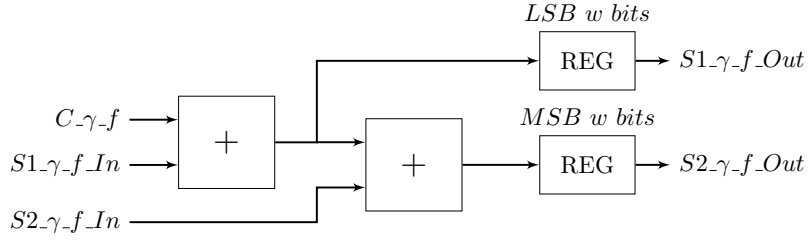

FIGURE C.8 – Gamma_f Processing Element internal architecture.

In the remainder of this section we detail our design for a s = 8 and a s = 16 architectures, respectively denoted NW-8 and NW-16.

C.4.4 Our architectures

Firstly, we will start with the NW-8 architecture which contains 3 PEs of type alpha and 3 of type gamma. With this design we can compute a modular multiplication in 33 clock cycles. Secondly we will present the NW-16 architecture that is composed by 6 PEs of type alpha and 6 PEs of type gamma. And we can perform a modular multiplication with this architecture in 66 clock cycles. Similarly, in order to implement the NW-32 architecture and the NW-64 architecture we need every time to double the number of cells. We provide a comparison of our architectures at the end of this section.

NW-8 Architecture

In this architecture, the operands and the modulo are divided in 8 words as illustrated in Figure C.11. As illustrated in Figure C.9 the NW-8 architecture is composed of 9 Processing Elements distributed in a two-dimensional array. Every Processing Elements are responsible for the calculus involving w bits words of the input operands. For example, for a 256 bits modular multiplication with NW-8, the operands are split in 8 words of 32 bits which results in a two-dimensional array of 9 Processing Elements. The 9 Processing Elements are divided in the following manner : 3 cells alpha, 1 cell alpha_final, 1 cell beta, 3 cells gamma, et 1 cell gamma_final. Those choices were made in order to optimize the number of states in our FSM.

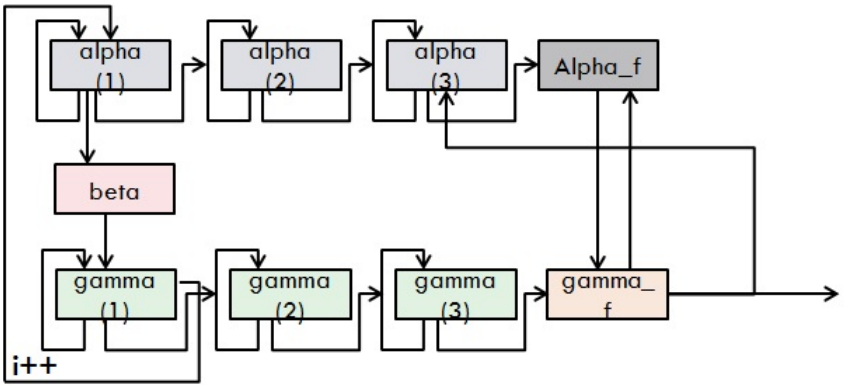

FIGURE C.9 – CIOS NW-8 Architecture.

As seen in section C.2 each PE in the N-dimensional array is connected to 2N data In/Out paths for communicating with 2N PEs in the N-dimensional array. Since we are working with two-dimensional elements, each PE in our design is connected to 4 data paths, 2 Input and 2 Output as presented in Figure C.3. In this architecture, the Processing Elements are designed with finite state machines (FSM). The control block communicates with the PEs and shift registers through starts signals. The Figure C.10 presents an overview of our architecture. For more technical details the Figure C.25 presents the differents PEs with input/output. The shift register is designed to provide the required words for a modular multiplication to the PEs. The Processing Element alpha requires words $a[i]$ and $b[j]$ of the operands a and b, on the other side the Processing Element gamma required a words of the operand p. Thus, these operands are defined in the package body. At the end of the

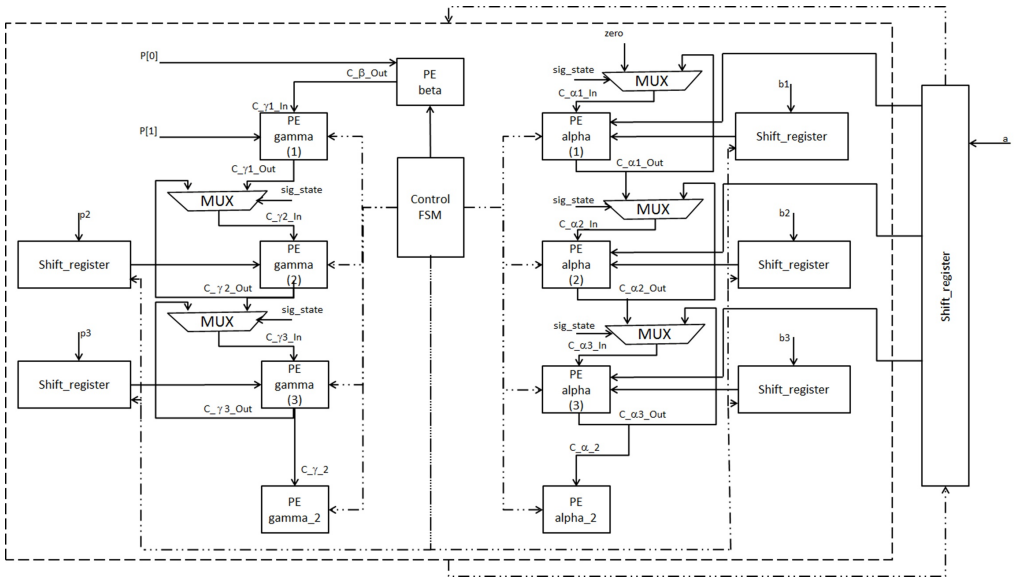

FIGURE C.10 – Proposed Montgomery modular multiplication architecture.

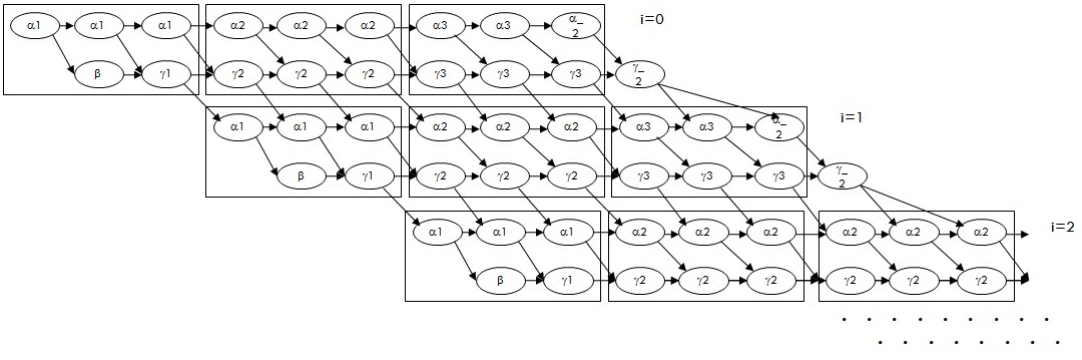

FIGURE C.11 – The data dependency graph of the proposed new Systolic Architecture with a Tow-dimensional array of Processing Elements (NW-8).

Montgomery modular multiplication, the control block provides the multiplication result $a \cdot b \cdot R^{-1} \mod p$ through the outputs of the last gamma and gamma_final Processing Elements. To evaluate the number of clock cycles for a CIOS method of modular multiplication in NW-8, the first parameter is max{number of alpha, number of gamma}=3, it means that our design can handle three iterations of i at the same time as illustrated in Figure C.11. Implying that our algorithm require to loop $s + 3$ times. we can performing our design in 33 clock cycles since our design requires three states ($33 = 3 \times (s + 3)$). The different results of this architecture in bit-length 256 are given in Table C.2. And we illustrate a part of the execution of this architecture in the appendix C.8.1. To ensure the use of the word adapted te each steps of execution, we used the rotations and shift. The operand B was divided

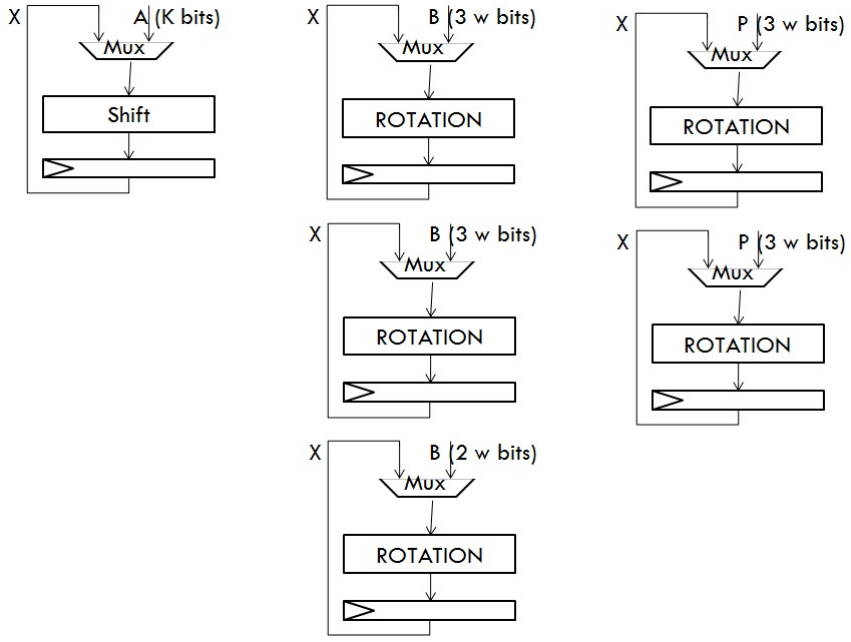

FIGURE C.12 – Internal architectures - Rotations.

into 3 vectors, the first vectoris of size 3 words, the second vector is of size 3 words and the last vector is of size 2 words. Then we need 3 rotations for this 3 vectors. Similarly, the modulo is divided into 4 vectors. And we used 4 rotations. The Figure C.12 illustrate the different rotations used in our architecture of NW-8.

NW-16 Architecture

In this architecture, the operands and the modulo are divided in 16 words as presented in Figure C.15. As illustrated in Figure C.13 the NW-16 architecture is designed in the same way as the NW-8. This example illustrates the scalability of our design. The NW-16 architecture is composed of 15 Processing Elements distributed in a two-dimensional array, where every Processing Elements are responsible for the calculus involving w bits words of the input operands. The 15 Processing Elements are divided like this : 6 cells alpha, 1 cell alpha_final, 1 cell beta, 6 cells gamma et 1 cell gamma_final. We can remark that the number of PEs of type alpha and

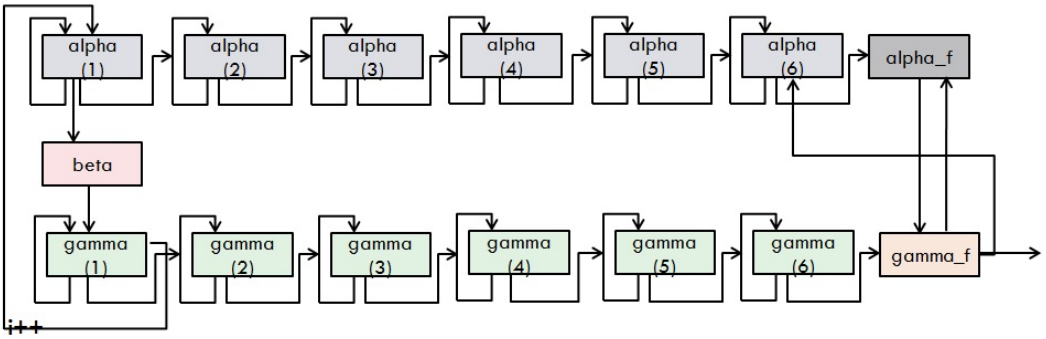

FIGURE C.13 – CIOS NW-16 Architecture.

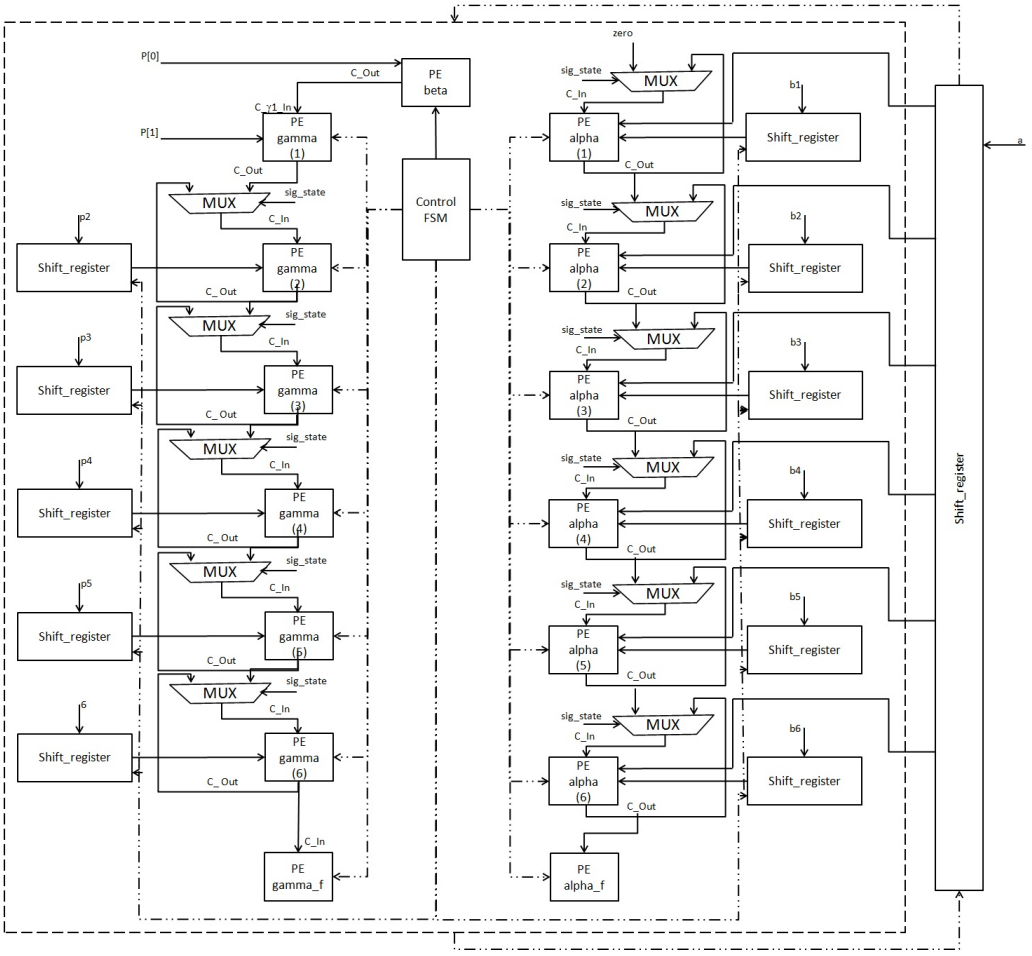

FIGURE C.14 – Proposed Montgomery modular multiplication architecture for the NW-16 version.

Artix-7	DSP	Frequency (MHz)	Clock cycle
MMM (s=8/K=256)	31	105.275	33
Alpha	4	291.023	1
Gamma	4	291.023	1
Beta	4	388.350	1
Alpha_final	1	459.918	1
Gamma_final	2	442.811	1

TABLE C.2 – Implementations of cells and MMM (NW-8).

gamma are the double of the number for NW-8. As said previously, the number of other PE type (alpha_final, beta, gamma_final) remains unchanged whatever the number of words in the design. We illustrate our architecture in Figure C.15. In order to evaluate the number of clock cycles of the NW-16 architecture, the first parameter is max{number of alpha, number of gamma}=6, implying that our algorithm requires to loop $s + 6$ times. We can perform the multiplication with our design in 66 clock cycles since our design requires three states ($66 = 3 \times (s + 6)$). The different results of this architecture in bit-length 256 are given in Table C.3. In this version the operand B was divided into 6 vectors, the first 6 vectors are of size 3 words, and the last vector is of size 1 word. Then we need 6 rotations for this 6 vectors. Similarly, the modulo is divided into 7 vectors. But we used 5 rotations.

Artix-7	DSP	Frequency (MHz)	Clock cycle
MMM (s=16/K=256)	29	145.892	66
Alpha	2	379.341	1
Gamma	2	379.341	1
Beta	2	453.104	1
Alpha_final	1	459.918	1
Gamma_final	2	442.811	1

TABLE C.3 – Implementations of cells and MMM (NW-16).

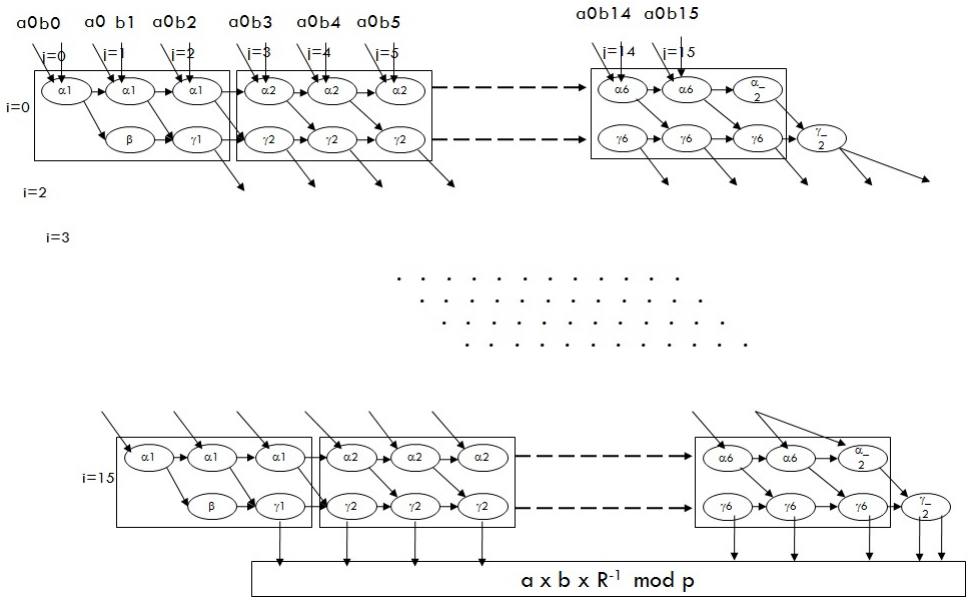

FIGURE C.15 – The data dependency graph of the proposed new Systolic Architecture with a Tow-dimensional array of Processing Elements (NW-16).

NW-32 Architecture

In this architecture, the operands and the modulo are divided in 32 words. The NW-32 architecture is composed of 27 Processing Elements distributed in a two-dimensional array, where every Processing Elements are responsible for the calculus involving w bits words of the input operands. The 27 Processing Elements are divided like this : 12 cells alpha, 1 cell alpha_final, 1 cell beta, 12 cells gamma et cell gamma_final. In order to evaluate the number of clock cycles of the NW-32 architecture, the first parameter as we have seen previously is max{number of alpha, number of gamma}=12, implying that our algorithm require to loop $s+12$ times. We can perform the multiplication with our design in 132 clock cycles since our design requires three states $(132 = 3 \times (s + 12))$.

NW-64 Architecture

In this architecture, the operands and the modulo are divided in 64 words. The NW-64 architecture is composed of 51 Processing Elements distributed in a two-dimensional array, where every Processing Elements are responsible for the calculus involving w bits words of the input operands. The 51 Processing Elements are divided like this : 24 cells alpha, 1 cell alpha_final, 1 cell beta, 24 cells gamma et 1 cell gamma_final. In order to evaluate the number of clock cycles of the NW-64 architecture, the first parameter is max{number of alpha, number of gamma}=24, implying that our algorithm require to loop $s + 24$ times. We can perform the multiplication with our design in 264 clock cycles since our design requires three states $(264 = 3 \times (s + 24))$.

NW-s Architecture

In this subsection, we will generalize our design. With the same manner in the previous architectures we can propose a generic version. The operands and the modulo are divided in s-words. The NW-s architecture is composed of nbc Processing Elements distributed in a two-dimensional array, where every Processing Elements are responsible for the calculus involving w bits words of the input operands. Where nbc=$(s - s/4) + 3$ cells (nbc is the number of cells). The number of clock cycles is $3 \times (s + nb)$ with nb=max{number of cells alpha, number of cells gamma}=$\frac{(s - s/4)}{2}$, implying that our algorithm require to loop $s + nb$ times. We can perform the multiplication with our design in $3 \times (s + nb)$ clock cycles since our design requires three states.

Architectures comparison

The Table C.4 explains a comparison between the different architectures. Number of clock cycles for every architecture equal to 3 × (s+nb), such that nb=max{number of cells alpha, number of cells gamma}, implying that our algorithm require to loop $s + nb$ times. It is interesting to notice that all our architectures are scalable and targeting the different security levels useful in cryptography.

CIOS	s=8	s=16	s=32	s=64
K=256	32	16	8	4
K=512	64	32	16	8
K=1024	128	64	32	16
K=2048	256	128	64	32
Clock cycles= 3 × (s+nb)	33	66	132	264
Number of cells	6 +3	12 +3	24 +3	48 +3

TABLE C.4 – Comparison of our architectures

C.5 Results

The Table C.5 summarizes the FPGA results postimplementation of the proposed versions of modular multiplication architectures. We present a results for the both architectures NW-8 and NW-16. The designs were described in hardware description languages (VHDL) and synthesized for Artix-7 and Virtex-5 Xilinx FPGAs. In order to check the correctness of the result, we compare the results given by the FPGA with

Artix 7- Nexys 4

	NW-8			NW-16		
	128	256	512	256	512	1024
Freq MHz	198	106	65	146	106	65
cycles	33	33	33	66	66	66
Slice Registers	487	870	1614	1123	2164	4208
Slice LUTs	355	809	2650	846	1789	5242
Slices	206	352	878	402	798	2072
DSP	19	31	87	29	57	161

TABLE C.5 – illustration of the scalabilty of our architecture.

the sage code. We present the different results after implementation of bit-length k which are given in Table C.5. These circuits have the advantage of suitability to various applications with different bit lengths like RSA, ECC and pairings. As it is shown in Table C.5, an interesting property of our design is the fact that the clock cycles are independent from the bit length. This property gives to our design the advantage of suitability to different security level. In order to implement the modular Montgomery multiplication for fixed security level, we must choose the most suitable architecture. The results presented in this work are compared with the previous work [161, 157, 103, 99] in the Table C.6. We could notice that our results are better then [161] considering every point of comparison i.e. the number of slice and the number of clock cycles. Considering the number of slices, we recall that [161] used an external memory to optimize the number of slices used by their algorithms. Considering the comparison with [157], our design requires less number of slices, and a better frequency and we really improve the number of clock cycles. Our design performed the Montgomery multiplication in 66 clock cycles for the 512 and 1024 bit length corresponding to AES-256 and AES-512 security level, while [157] performed the multiplication in 1540 clock cycles for the AES-256 security level and 3076 for the AES-512 security level.

C.6 Conclusion

In this paper we have presented an efficient hardware implementation of the CIOS method of Montgomery multiplication Algorithm over large prime characteristic finite fields $\mathbb{F}_p$. We give the results of our design after routing and placement using

Xilinx FPGAs

	Our A7		**Our V5**		[161] V5		[157] VE		[103] VII	[99] VII	[136] V		[26] K
	512	**1024**	**512**	**1024**	512	1024	512	1024	1024	1024	512	1024	512 K
Freq MHz	**106**	**65**	**97**	**65**	95	130	95.229	95.620	116.4	119	72.1	79.2	176
Cycles	**66**	**66**	**66**	**66**	96	384	1540	3076	1088	1167	–	–	66
Speed μs	**0.622**	**1.013**	**0680**	**1.015**	1.010	2.953	16.031	32.021	9.34	9.80	–	–	0.374
Slice Registers	**2164**	**4208**	**3046**	**6072**	3876	6642	–	–	–	–	–	–	5076
Slice LUTs	**1789**	**5242**	**1781**	**5824**	–	–	2972	5706	9319	9271	3125	6243	8757
BRAM	0	0	0	0	128	256	–	–	–	–	–	–	0

TABLE C.6 – Comparaison of our work with state-of-art implementations.

a Artix-7 and Virtex-5 Xilinx FPGAs. Our systolic implementations is suitable for every implementation implying a modular multiplication, for example RSA, ECC and pairing-based cryptography. Our architectures and the designs were matched with features of the FPGAs. The NW-8 design presented a good performance considering *latency* × *area* efficiency. This architecture can run for all the bit length corresponding to classical security levels (128, 256, 512 or 1024 bits) in just 33 clock cycles. On the other hand the NW-16 perform the same bit length in 66 clock cycles, but improve in area compared to NW-8 work. Our systolic design using this method CIOS is scalable for other number of words.

C.7 Appendix 1

C.7.1 Code Sage NW-8

```
#NW-8 Algoritm
s=8
p'
p=[p0,p1,p2,p3,p4,p5,p6,p7]
b=[b0,b1,b2,b3,b4,b5,b6,b7]
a=[a0,a1,a2,a3,a4,a5,a6,a7]
T=[0,0,0,0,0,0,0,0,0,0]
for i in range (s):
    C_S=0
    for j in range (0,s):
```

```
        C_S=T[j]+a[i]*b[j]+(C_S>>32)
        T[j]=C_S%(2^32)
    C_S=T[s]+(C_S>>32)
    T[s]=C_S%(2^32)
    T[s+1]=C_S>>32
    m=(T[0]*p')%(2^32)
    C_S=T[0]+m*p0
    for j in range (1,s):
        C_S=T[j]+m*p[j]+(C_S>>32)
        T[j-1]=C_S%(2^32)
    C_S=T[s]+(C_S>>32)
    T[s-1]=C_S%(2^32)
    T[s]=T[s+1]+(C_S>>32)
```

C.7.2 Code Sage NW-16

```
#NW-16 Algoritm
s=16
p'
p=[p0,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p14,p15]
b=[b0,b1,b2,b3,b4,b5,b6,b7,b8,b9,b10,b11,b12,b13,b14,b15]
a=[a0,a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12,a13,a14,a15]
T=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
for i in range (s):
    C_S=0
    for j in range (0,s):
        C_S=T[j]+a[i]*b[j]+(C_S>>16)
        T[j]=C_S%(2^16)
    C_S=T[s]+(C_S>>16)
    T[s]=C_S%(2^16)
    T[s+1]=C_S>>16
    m=(T[0]*p')%(2^16)
    C_S=T[0]+m*p0
    for j in range (1,s):
        C_S=T[j]+m*p[j]+(C_S>>16)
        T[j-1]=C_S%(2^16)
    C_S=T[s]+(C_S>>16)
    T[s-1]=C_S%(2^32)
    T[s]=T[s+1]+(C_S>>16)
```

C.8 Appendix 2 : architecture

C.8.1 Execution

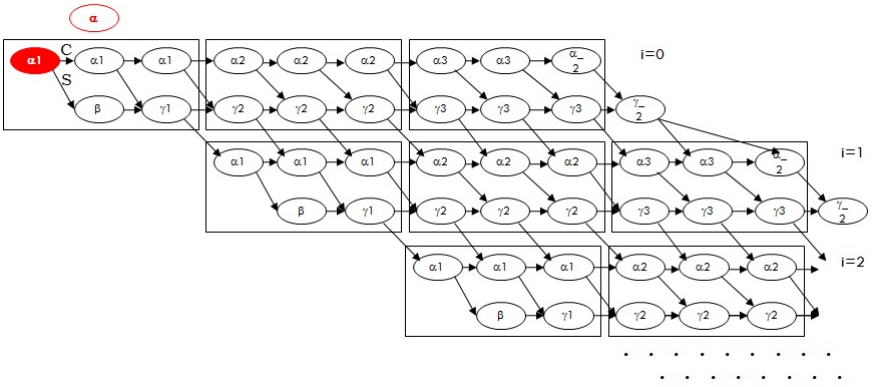

FIGURE C.16 – Step 1.

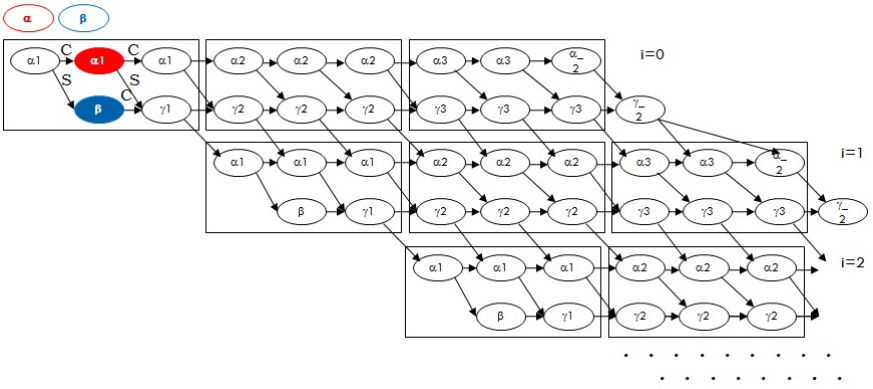

FIGURE C.17 – Step 2.

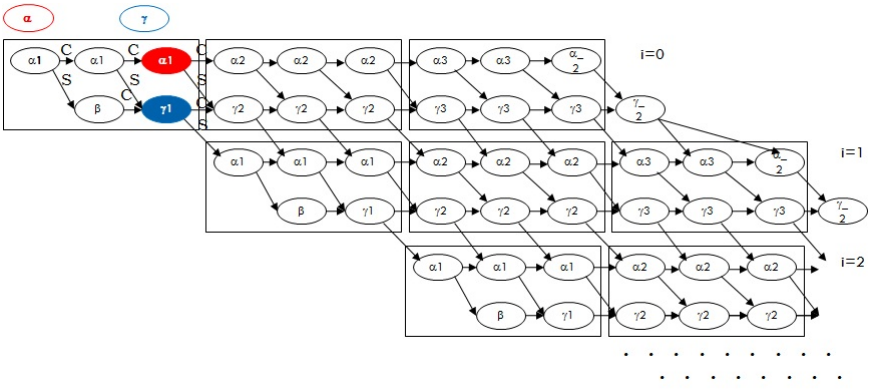

FIGURE C.18 – Step 3.

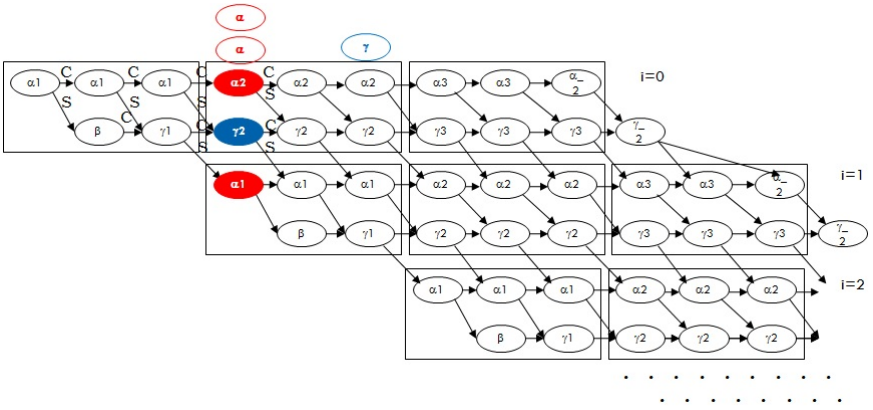

FIGURE C.19 – Step 4.

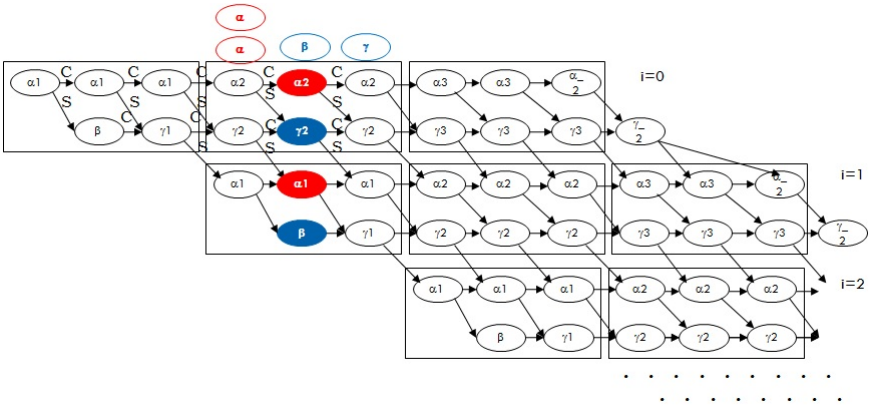

FIGURE C.20 – Step 5.

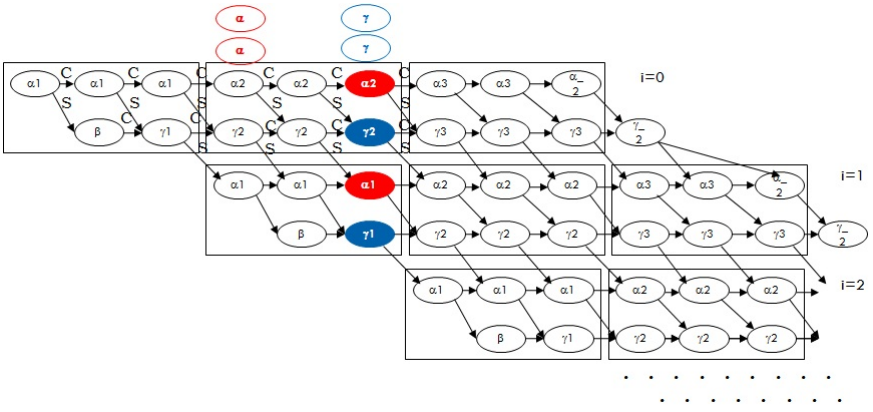

FIGURE C.21 – Step 6.

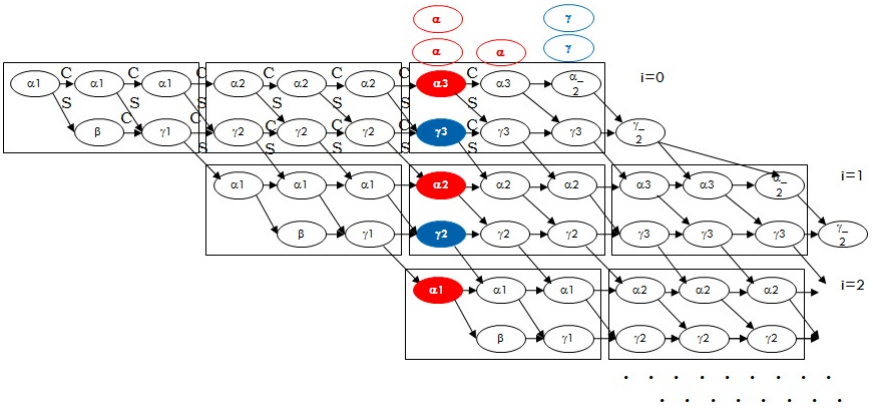

FIGURE C.22 – Step 7.

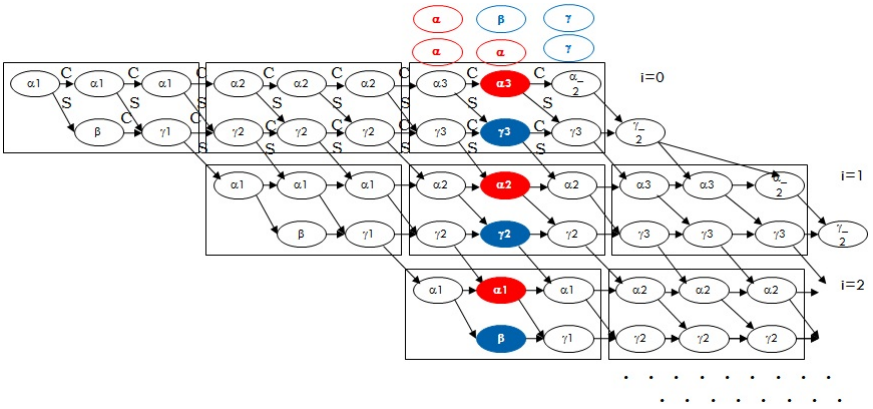

FIGURE C.23 – Step 8.

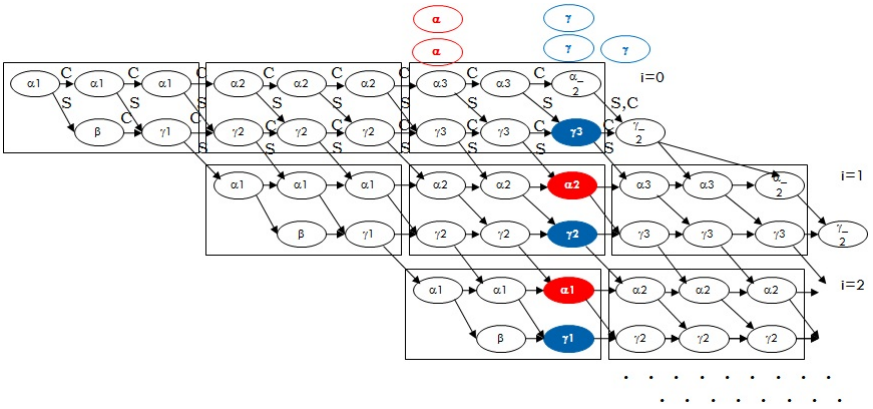

FIGURE C.24 – Step 9.

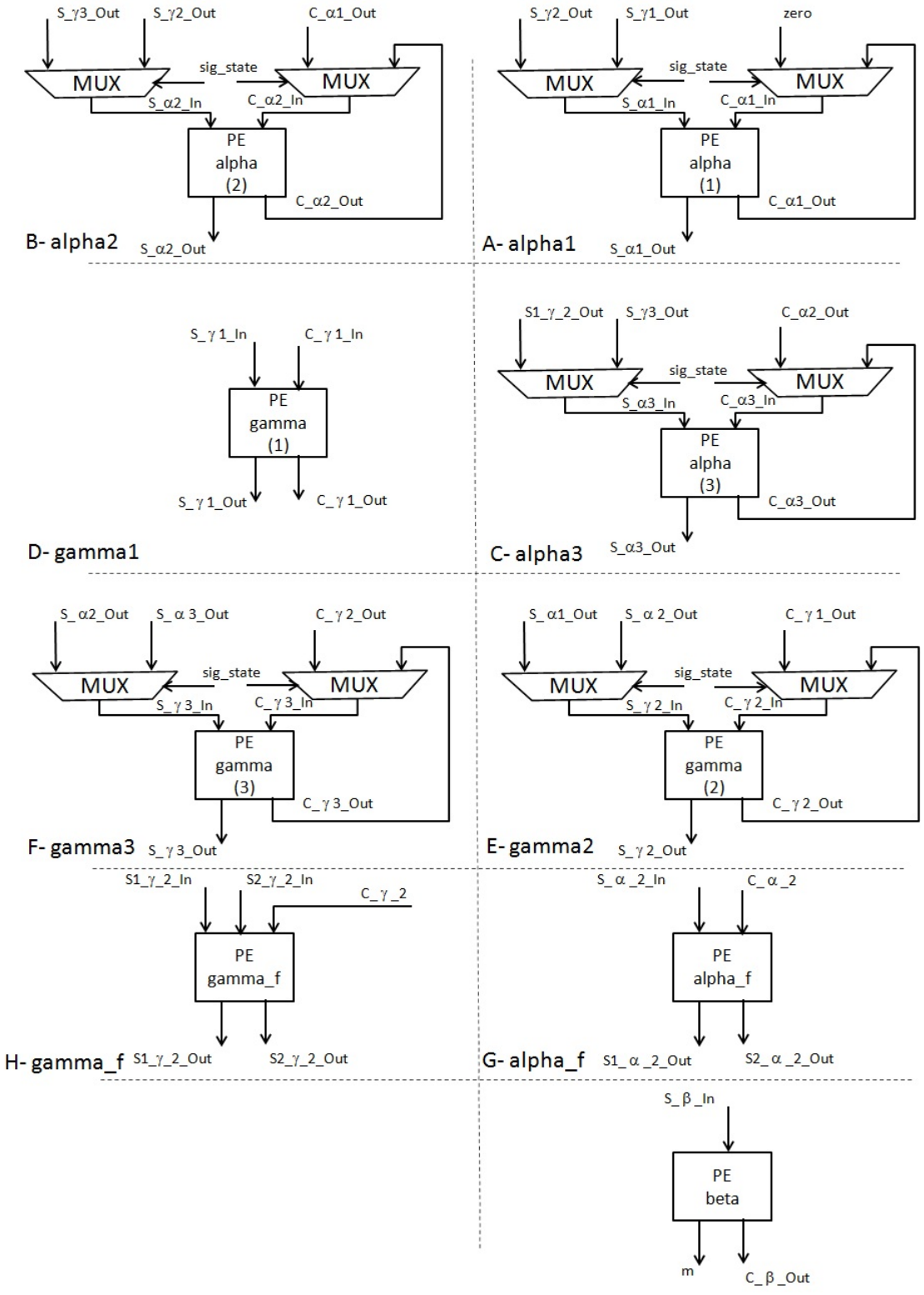

FIGURE C.25 – All Processing Elements.

Annexe D

Efficient Pairings Computation on Jacobi Quartic Elliptic Curves

Journal Mathematical Cryptography 2014 [56]
avec Sylvain Duquesne et Emmanuel Fouotsa[1]

Abstract : This paper proposes the computation of the Tate pairing, Ate pairing and its variations on the special Jacobi quartic elliptic curve $Y^2 = dX^4 + Z^4$. We improve the doubling and addition steps in Miller's algorithm to compute the Tate pairing. We use the birational equivalence between Jacobi quartic curves and Weierstrass curves, together with a specific point representation to obtain the best result to date among curves with quartic twists. For the doubling and addition steps in Miller's algorithm for the computation of the Tate pairing, we obtain a theoretical gain up to 27% and 39%, depending on the embedding degree and the extension field arithmetic, with respect to Weierstrass curves [46] and previous results on Jacobi quartic curves [185]. Furthermore and for the first time, we compute and implement Ate, twisted Ate and optimal pairings on the Jacobi quartic curves. Our results are up to 27% more efficient, comparatively to the case of Weierstrass curves with quartic twists [46].

D.1 Introduction

Bilinear pairings were first used to solve the discrete logarithm problem on elliptic curve groups [138, 83]. But they are now useful to construct many public key protocols for which no other efficient implementation is known [113, 33]. A survey of some of these protocols can be found in [60]. The efficient computation of pairings depends on the model chosen for the elliptic curve. Pairing computation on the Edwards model of elliptic curves has been done successively in [49, 106] and [7]. The recent results on pairing computation using elliptic curves of Weierstrass form can

[1]This work was supported in part by French projects ANR-12-BS01-0010-01 "PEACE", ANR-12-INSE-0014 "SIMPATIC" and by the LIRIMA MACISA project. This work is an improved and extended version of the work [58] appeared in the proceedings of Pairing 2012.

be found in [45, 46]. Recently in [185], Wang et al. have computed the Tate pairing on Jacobi quartic elliptic curves using the geometric interpretation of the group law. In this paper, we focus on the special Jacobi quartic elliptic curve $Y^2 = dX^4 + Z^4$ over fields of large characteristic $p \geq 5$ not congruent to 3 modulo 4.
For pairing computation with embedding degree divisible by 4, we define and use the quartic twist of the curve $Y^2 = dX^4 + Z^4$. Our results improve those obtained by Wang et al. in [185] and they are more efficient than those concerning the Tate pairing computation in Weierstrass elliptic curves [46].

Furthermore, the Miller algorithm is the main tool in the Tate pairing computation, and its efficiency has been successfully improved in the last years leading to other pairings such as :

- The Eta-pairing [16] on supersingular elliptic curves.
- The Ate and twisted Ate pairings introduced in [101] that are closely related to the Eta-pairing, but can be used efficiently with ordinary elliptic curves. These pairings can be more efficient than the Tate pairing, essentially due to the reduction of the number of iterations in the Miller algorithm.
- In [183] and [100], Vercauteren and Hess generalize the method with the notion of optimal pairings and pairings lattices that can be computed using the smallest number of basic Miller's iterations.

The computation of these different pairings has been done by Costello et al. [46] in the case of Weierstrass curves. As a second contribution of this work, we extend the results on the special Jacobi quartic in [58] to the computation of the Ate pairing and its variations. We show that among known curves with quartic twists, the Jacobi model $Y^2 = dX^4 + Z^4$ offers the best performances for all these different pairings.
The rest of this paper is organized as follows : Section D.2 provides a background on the Jacobi elliptic curve, and notions on pairings that are useful in the paper. In Section D.3, we present the computation of the Tate pairing on the Jacobi quartic curve mentioned above using birational equivalence and we compare our results to others in the literature. In Section D.4, we determine the Miller function and rewrite the addition formulas for the Ate pairing. We also provide a comparative study of these pairings on the curves in Jacobi and Weierstrass forms. In Section D.5 we provide an example of pairing friendly curve of embedding degree 8. An implementation of the Tate, Ate and optimal Ate pairings based on this example has been done using the Magma computer algebra system. This enable to verify all the formulas given in this paper. Finally, we conclude in Section D.6.

The following notations are used in this work.
$\mathbb{F}_q$: A finite field of characteristic $p \geq 5$, not congruent to 3 modulo 4.
m_k, s_k : Cost of multiplication and squaring in the field $\mathbb{F}_{q^k}$, for any integer k
mc : Cost of the multiplication by a constant in $\mathbb{F}_q$

D.2 Background on Pairings and on Jacobi Elliptic Curves

In this section, we briefly review pairings on elliptic curves and the Jacobi quartic curves. We also define twists of Jacobi's curves.

D.2.1 The Jacobi Quartic Curve

A Jacobi quartic elliptic curve over a finite field $\mathbb{F}_q$ is defined by

$$E_{d,\mu} : y^2 = dx^4 + 2\mu x^2 + 1$$

with discriminant $\triangle = 256d(\mu^2 - d)^2 \neq 0$. In [27] Billet and Joye proved that if the Weierstrass curve $E : y^2 = x^3 + ax + b$ has a rational point of order 2 denoted $(\theta, 0)$, then it is birationally equivalent to the Jacobi quartic $E_{d,\mu}$ with $d = -(3\theta^2 + 4a)/16$ and $\mu = -3\theta/4$. In the remainder of this paper, we will focus our interest on the special Jacobi quartic curve

$$E_{d,0} : y^2 = dx^4 + 1$$

because this curve has interesting properties such as a quartic twist which will contribute to an efficient computation of pairings.
The addition and doubling formulas on $E_{d,0}$ are deduced from [102].
The point addition $(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$ is given by :

$$x_3 = \frac{x_1^2 - x_2^2}{x_1 y_2 - y_1 x_2} \text{ and } y_3 = \frac{(x_1 - x_2)^2}{(x_1 y_2 - y_1 x_2)^2}(y_1 y_2 + 1 + d x_1^2 x_2^2) - 1.$$

The point doubling $(x_3, y_3) = 2(x_1, y_1)$ on $E_{d,0}$ is given by :

$$x_3 = \frac{2y_1}{2 - y_1^2} x_1 \text{ and } y_3 = \frac{2y_1}{2 - y_1^2}\left(\frac{2y_1}{2 - y_1^2} - y_1\right) - 1.$$

The birational equivalence, deduced from [27], between the Weierstrass curve $W_d : y^2 = x^3 - 4dx$ and the Jacobi quartic curve $E_{d,0}$ is given by

$$\varphi : \begin{array}{ccc} E_{d,0} & \longrightarrow & W_d \\ (0,1) & \longmapsto & P_\infty \\ (0,-1) & \longmapsto & (0,0) \\ (x,y) & \longmapsto & \left(2\frac{y+1}{x^2}, 4\frac{y+1}{x^3}\right) \end{array} \quad \text{and} \quad \varphi^{-1} : \begin{array}{ccc} W_d & \longrightarrow & E_{d,0} \\ P_\infty & \longmapsto & (0,1) \\ (0,0) & \longmapsto & (0,-1) \\ (x,y) & \longmapsto & \left(\frac{2x}{y}, \frac{2x^3 - y^2}{y^2}\right) \end{array}$$

From now on, and for efficiency reasons, we adopt for the first time in pairings computation a specific points representation namely $(x, y) = (\frac{X}{Z}, \frac{Y}{Z^2})$. The curve $E_{d,0}$ is then equivalent to

$$E_d : Y^2 = dX^4 + Z^4.$$

The addition and doubling formulas on E_d are as follows. The point addition $(X_3 : Y_3 : Z_3) = (X_1 : Y_1 : Z_1) + (X_2 : Y_2 : Z_2)$ on E_d is given by :

$$\begin{cases} X_3 &= X_1^2 Z_2^2 - Z_1^2 X_2^2, \\ Z_3 &= X_1 Z_1 Y_2 - X_2 Z_2 Y_1, \\ Y_3 &= (X_1 Z_2 - X_2 Z_1)^2 (Y_1 Y_2 + (Z_1 Z_2)^2 + d(X_1 X_2)^2) - Z_3^2. \end{cases}$$

The point doubling $(X_3 : Y_3 : Z_3) = 2(X_1 : Y_1 : Z_1)$ on E_d is given by :

$$\begin{cases} X_3 &= 2X_1Y_1Z_1, \\ Z_3 &= Z_1^4 - dX_1^4, \\ Y_3 &= 2Y_1^4 - Z_3^2. \end{cases}$$

The birational equivalence between the projective model $E_d : Y^2 = dX^4 + Z^4$ and the Weierstrass curve $W_d : y^2 = x^3 - 4dx$ becomes

$$\begin{array}{rccl} \varphi : & E_d & \longrightarrow & W_d \\ & (0:1:1) & \longmapsto & P_\infty \\ & (0:-1:1) & \longmapsto & (0,0) \\ & (X:Y:Z) & \longmapsto & \left(2\frac{Y+Z^2}{X^2}, 4\frac{Z(Y+Z^2)}{X^3}\right) \end{array} \qquad \begin{array}{rccl} \varphi^{-1} : & W_d & \longrightarrow & E_d \\ & P_\infty & \longmapsto & (0:1:1) \\ & (0,0) & \longmapsto & (0:-1:1) \\ & (x,y) & \longmapsto & (2x:2x^3-y^2:y). \end{array}$$

The Sage software code to verify the correctness of our formulas is available here : `http://www.prmais.org/Implementation-Pairings-Jacobi.txt`.

D.2.2 Pairings on Elliptic Curves

In this section, we first recall the Tate pairing. Then, the notion of twists of elliptic curves is defined to recall the definition of the Ate pairing ant its variations. Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$. The neutral element of the additive group law defined on the set of rational points of E is denoted P_∞. Let r be a large prime divisor of the group order $\sharp E(\mathbb{F}_q)$ and k be the embedding degree of E with respect to r, i.e. the smallest integer such that r divides $q^k - 1$. The set $E\left(\overline{\mathbb{F}_q}\right)[r] = \{P \in E\left(\overline{\mathbb{F}_q}\right) : [r]P = P_\infty\}$ is the set of $r-$torsion points with coordinates in an algebraic closure $\overline{\mathbb{F}_q}$ of $\mathbb{F}_q$, where $[\ ] : P \longmapsto [r]P$ is the endomorphism defined on $E(\mathbb{F}_q)$ which consists to add P to itself r times. The integer k is also the smallest integer such that $E\left(\overline{\mathbb{F}_q}\right)[r] \subset E(\mathbb{F}_{q^k})$ and this is the main property that we use in this work.

The Tate pairing.

Consider a point $P \in E(\mathbb{F}_q)[r]$ and the divisor $D = r(P) - r(P_\infty)$, then according to [175, Corollary 3.5, Page 67], D is principal and so there is a function $f_{r,P}$ with divisor $\mathrm{Div}\,(f_{r,P}) = D$. Let Q be a point of order r with coordinates in $\mathbb{F}_{q^k}$ and μ_r be the group of r-th roots of unity in $\mathbb{F}_{q^k}^*$. The reduced Tate pairing e_r is a bilinear and non degenerate map defined as

$$\begin{array}{rccl} e_r : & E(\mathbb{F}_q)[r] \times E(\mathbb{F}_{q^k})[r] & \to & \mu_r, \\ & (P,Q) & \mapsto & f_{r,P}(Q)^{\frac{q^k-1}{r}}. \end{array}$$

The value $f_{r,P}(Q)$ can be determined efficiently using Miller's algorithm [141]. Indeed, for an integer i, consider the divisor $D_i = i(P) - ([i]P) - (i-1)(P_\infty)$. We observe that D_i is a principal divisor and so there is a function $f_{i,P}$ such that $\mathrm{Div}(f_{i,P}) = i(P) - ([i]P) - (i-1)(P_\infty)$. Observe that

$$\text{for } i = r \text{ one has } D_r = r(P) - r(P_\infty) = \mathrm{Div}(f_{r,P}).$$

Thus, to obtain the value of $f_{r,P}(Q)$, it suffices to apply an iterative algorithm using an *addition chain* for r, that is, a sequence $(1, i_1, i_2, ..., r)$ such that each i_k is the sum of two previous terms of the sequence. This is justified by the fact that the functions $f_{i,P}$ are satisfying the following conditions :

$$f_{1,P} = 1 \text{ and } f_{i+j,P} = f_{i,P} f_{j,P} h_{[i]P,[j]P} \tag{D.2.1}$$

where $h_{R,S}$ denotes a rational function such that

$$\mathrm{Div}(h_{R,S}) = (R) + (S) - (S + R) - (P_\infty),$$

with R and S two arbitrary points on the elliptic curve. In the case of elliptic curves in Weierstrass form, $h_{R,S} = \frac{\ell_{R,S}}{v_{R+S}}$ where $\ell_{R,S}$ is the straight line defining $R + S$ and v_{R+S} is the corresponding vertical line passing through $R + S$.

Miller uses the *double-and-add* method as the addition chains for r (see [44, Chapter 9] for more details on addition chains) and the properties of $f_{i,P}$ to compute $f_{r,P}(Q)$. The Miller algorithm that computes efficiently the pairing of two points is given in Algorithm D.1.

Algorithm 1 : Miller's Algorithm

Input : $P \in E(\mathbb{F}_q)[r]$, $Q \in E(\mathbb{F}_{q^k})[r]$, $r = (1, r_{n-2}, ...r_1, r_0)_2$.

Output : The reduced Tate pairing of P and Q : $f_{r,P}(Q)^{\frac{q^k-1}{r}}$

1 : Set $f \leftarrow 1$ and $R \leftarrow P$
2 : **For** $i = n - 2$ **down to** 0 **do**
3 : $f \leftarrow f^2 \cdot h_{R,R}(Q)$
4 : $R \leftarrow 2R$
5 : **if** $r_i = 1$ **then**
6 : $f \leftarrow f \cdot h_{R,P}(Q)$
7 : $R \leftarrow R + P$
8 : **end if**
9 : **end for**
10 : **return** $f^{\frac{q^k-1}{r}}$

FIGURE D.1 – The Miller algorithm for the computation of the reduced Tate pairing

More informations on pairings can be found in [85] and [55].
Let us now define twists of elliptic curves and specialise to the case of Jacobi quartic curves. This notion of twists enable to work on smaller base fields for pairings computations.

Twists of elliptic curves.

A twist of an elliptic curve E defined over a finite field $\mathbb{F}_q$ is an elliptic curve E' defined over $\mathbb{F}_q$ that is isomorphic to E over an algebraic closure of $\mathbb{F}_q$. The smallest integer δ such that E and E' are isomorphic over $\mathbb{F}_{q^\delta}$ is called the degree of the twist.

Let $E : y^2 = x^3 + ax + b$ be an elliptic curve in Weierstrass form defined over $\mathbb{F}_q$. The equation defining the twist E' has the form $y^2 = x^3 + a\omega^4 x + b\omega^6$ where ω belongs to an extension $\mathbb{F}_{q^k}$ of $\mathbb{F}_q$ and the isomorphism between E' and E is

$$\begin{array}{rccc} \psi : & E' & \longrightarrow & E \\ & (x', y') & \longmapsto & (x'/\omega^2, y'/\omega^3). \end{array}$$

More details on twists can be found in [46].

Twist of Jacobi quartic curves.

To obtain the twist of the Jacobi quartic curve $Y^2 = dX^4 + Z^4$, we use the birational maps defined in Subsection D.2.1 and the twist of Weierstrass curves defined above. Let k be an integer divisible by 4.

Définition D.2.1. [58] A quartic twist of the Jacobi quartic curve $Y^2 = dX^4 + Z^4$ defined over the extension $\mathbb{F}_{q^{k/4}}$ of $\mathbb{F}_q$ is a curve given by the equation

$$E_d^\omega : Y^2 = d\omega^4 X^4 + Z^4$$

where $\omega \in \mathbb{F}_{q^k}$ is such that $\omega^2 \in \mathbb{F}_{q^{k/2}}$, $\omega^3 \in \mathbb{F}_{q^k} \backslash \mathbb{F}_{q^{k/2}}$ and $\omega^4 \in \mathbb{F}_{q^{k/4}}$.
In other terms $\{1, \omega, \omega^2, \omega^3\}$ is a basis of $\mathbb{F}_{q^k}$ as a vector space over $\mathbb{F}_{q^{k/4}}$.

Proposition D.2.2. *Let E_d^ω defined over $\mathbb{F}_{q^{k/4}}$ be a twist of E_d. The $\mathbb{F}_{q^k}$-isomorphism between E_d^ω and E_d is given by*

$$\begin{array}{rccc} \psi : & E_d^\omega & \to & E_d, \\ & (X : Y : Z) & \mapsto & (\omega X : Y : Z). \end{array}$$

We explain in Section D.2.3 and in Section D.3.1 how twists are very useful for an efficient computation of pairings.

The Ate pairing and its variations.

In this section, we briefly define the Ate and twisted Ate pairings. The results in this section are very well described in the original article of Hess et al. [101]. We recall that $f_{i,R}$ is the function with divisor

$$\mathrm{Div}(f_{i,R}) = i(R) - ([i]R) - (i-1)(P_\infty).$$

Let

$$\begin{array}{rccc} \pi_q : & E\left(\overline{\mathbb{F}_q}\right) & \to & E\left(\overline{\mathbb{F}_q}\right) \\ & (x, y) & \mapsto & (x^q, y^q) \end{array}$$

be the Frobenius endomorphism on the curve, and t be its trace. The characteristic polynomial of π_q is $X^2 - tX + q$, see [186, Chapter 4]. Using the fact that π_q satisfies its characteristic polynomial (Cayley Hamilton theorem), we have the following equality :

$$\pi_q^2 - t\pi_q + q = 0.$$

The relation between the trace t of the Frobenius endomorphism and the group order is given by [186, Theorem 4.3] :

$$\sharp E(\mathbb{F}_q) = q + 1 - t.$$

The Frobenius endomorphism π_q has exactly two eigenvalues. Indeed, using the Lagrange theorem in the multiplicative group $(\mathbb{F}_q^*, \times)$, it is clear that 1 is an eigenvalue. We then use the characteristic polynomial to conclude that q is the other one. This enables to consider $P \in \mathbb{G}_1 = E\left(\overline{\mathbb{F}_q}\right)[r] \cap \mathrm{Ker}(\pi_q - [1]) = E(\mathbb{F}_q)[r]$ and $Q \in \mathbb{G}_2 = E\left(\overline{\mathbb{F}_q}\right)[r] \cap \mathrm{Ker}(\pi_q - [q])$. The Ate pairing is defined as follows :

Définition D.2.3. (The Ate pairing) The reduced Ate pairing is the map :

$$\begin{array}{rccl} e_A : & \mathbb{G}_2 \times \mathbb{G}_1 & \to & \mu_r, \\ & (Q, P) & \mapsto & f_{T,Q}(P)^{\frac{q^k-1}{r}}, \end{array}$$

where $T = t - 1$.

The following theorem gives some properties of the Ate pairing, in particular its relation with the Tate pairing. This relation makes sense to Definition D.2.3 : the Ate pairing is a power of the Tate pairing and therefore is a pairing. A complete proof can be found in [101]

Théorème D.2.4. *[101] Let $N = gcd(T^k - 1, q^k - 1)$ and $T^k - 1 = LN$. We have*

- $e_A(Q, P)^{rc} = (f_{r,Q}(P)^{(q^k-1)/r})^{LN}$ *where* $c = \sum_{i=0}^{k-1} T^{k-1-i} q^i \equiv kq^{k-1} \mod r$.
- *for $r \nmid L$, the Ate pairing e_A is non-degenerate.*

Remark 12. *The Tate pairing is defined on $\mathbb{G}_1 \times E(\mathbb{F}_{q^k})$, while Ate pairing is defined on $\mathbb{G}_2 \times \mathbb{G}_1$ with $\mathbb{G}_2 \subseteq E(\mathbb{F}_{q^k})$. This means that during the execution of the Miller algorithm in Ate pairing computation, the point addition is performed in an extension field of $\mathbb{F}_q$ whereas it was performed in $\mathbb{F}_q$ in the case of the Tate pairing. As the arithmetic over $\mathbb{F}_{q^k}$ is much more expensive than the arithmetic over $\mathbb{F}_q$, each step of Ate pairing is more expensive than a step of the Tate pairing. However the Miller loop length in the case of Ate pairing is* $\log_2 T$ *which is less (generally the half) than* $\log_2 r$*, the loop length for the Tate pairing.*

Observe that if the Ate pairing were defined on $\mathbb{G}_1 \times \mathbb{G}_2$, then it will be faster than the Tate pairing since its Miller loop length will be approximately halved. This remark yields to the definition of the twisted Ate pairing [101].

Définition D.2.5. (The twisted Ate pairing) [101] Assume that E has a twist of degree δ and $m = gcd(k, \delta)$. Let $e = k/m$ and $T_e = T^e \bmod r$, then the reduced twisted Ate pairing is defined as follows :

$$\begin{array}{rccl} e_{T_e} : & \mathbb{G}_1 \times \mathbb{G}_2 & \to & \mu_r, \\ & (P, Q) & \mapsto & f_{T_e,P}(Q)^{\frac{q^k-1}{r}}. \end{array}$$

As in the case of Ate pairing, the following theorem ensures that e_{T_e} is a pairing.

Théorème D.2.6. *[101]*

- $e_{T_e}(P,Q)^{rc} = e_T(P,Q)^{LN}$ *where* $e_T(P,Q)$ *is the Tate pairing and* $c = \sum_{i=0}^{m-1} T^{e(m-1-i)}q^{ei} \equiv mq^{e(m-1)} \ mod\ r$.
- *for* $r \nmid L$, *the twisted Ate pairing* e_{T_e} *is non-degenerate.*

Remark 13. *The reduced Tate and twisted Ate pairings are defined on* $\mathbb{G}_1 \times E(\mathbb{F}_{q^k})$ *and* $\mathbb{G}_1 \times \mathbb{G}_2$ *respectively. So they have the same complexity for each iteration of the Miller algorithm but the Miller loop parameter is* $T^e \ mod\ r$ *for the reduced twisted Ate pairing and* r *for the Tate pairing. Consequently, the twisted Ate pairing will be more efficient than the reduced Tate pairing only for curves with trace* t *such that* $T^e \ mod\ r$ *is significatively less than* r.

Optimal pairings.

The reduction of Miller's loop length is an important way to improve the computation of pairings. The latest work is a generalized method to find the shortest loop when possible, which leads to the concept of optimal pairing [183]. Indeed, observe that if k is the embedding degree with respect to r, then $r|q^k-1$ but $r \nmid q^i-1$ for any $1 \leq i < k$. This implies that $r|\Phi_k(q)$ where Φ_k is the $k-th$ cyclotomic polynomial. Since $T \equiv q \bmod r$ where $T = t-1$, we have $r|\Phi_k(T)$. More generally, if we consider the Ate$-i$ pairing, which is a generalisation of the Ate pairing with Miller function $f_{T_i,Q}$ where $T_i \equiv q^i \bmod r$, then

$$r|\Phi_{k/g}(T_i), \text{ where } g = gcd(i,k)$$

so that the minimal value for T_i is $r^{1/\varphi(k/g)}$ (where φ is the Euler's totient function) and the lowest bound is $r^{1/\varphi(k)}$, obtained for $g = 1$. We then give the following definition of optimal pairing, this is a pairing that can be computed with the smallest number of iterations in the Miller loop.

Définition D.2.7. [183] Let $e : G_1 \times G_2 \longrightarrow G_T$ be a non-degenerate, bilinear pairing with $|G_1| = |G_2| = |G_T| = r$, where the field of definition of G_T is $\mathbb{F}_{q^k}$. e is called an optimal pairing if it can be evaluated with about at most $(\log_2 r)/\varphi(k) + \varepsilon(k)$ Miller iterations, where $\varepsilon(k)$ is less than $\log_2 k$.

The lowest bound is attained for several families of elliptic curves. The following theorem gives the construction of an optimal pairing.

Théorème D.2.8. *[183, Theorem 4] Let* E *be an elliptic curve defined over* $\mathbb{F}_q$. *The embedding degree with respect to a large integer* r *dividing the order of the group* $\sharp E(\mathbb{F}_q)$ *is denoted* k. *Let* $\lambda = mr$ *be a multiple of* r *such that* $r \nmid m$ *and write* $\lambda = \sum_{i=0}^{l} c_i q^i$. *Remember* $h_{R,S}$ *is the function with divisor* $\mathrm{Div}(h_{R,S}) = (R) + (S) - (S+R) - (P_\infty)$ *where* R *and* S *are two arbitrary points on the elliptic curve* E. *If* $s_i = \sum_{j=i}^{l} c_j q^j$, *the map* $e_o : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r$ *defined as*

$$(Q,P) \longmapsto \left(\prod_{i=0}^{l} f_{c_i,Q}^{q^i}(P) \cdot \prod_{i=0}^{l-1} h_{[s_{i+1}]Q,[c_i q^i]Q}(P) \right)^{\frac{q^k-1}{r}}$$

defines a bilinear pairing. Furthermore, the pairing is non degenerate if

$$mkq^k \neq ((q^k-1)/r)\cdot\sum_{i=0}^{l} ic_i q^{i-1} \ mod\ r.$$

In Section D.5, we apply the previous theorem to provide an example of optimal pairing on Jacobi quartic curves of embedding degree 8. Observe that the computation of optimal pairings follows the same approach as the computation of the Ate pairing.

D.2.3 Use of Twists for Efficient Computation of Pairings

For the applications of twists, observe that the points input of the Tate pairing, Ate pairing, twisted Ate or optimal pairing on a curve of embedding degree k take the form $P \in E(\mathbb{F}_q)$ and $Q \in E(\mathbb{F}_{q^k})$. In the case of the Tate pairing and the twisted Ate pairing, the evaluation of the Miller function is done at the point Q in the full extention $\mathbb{F}_{q^k}$ whereas in the case of Ate and Optimal Ate pairings, it is the addition of point that is performed there. In both cases, this can affect the efficiency of computations. However many authors (see for example [82] or [46]) have shown that one can use the isomorphism between the curve and its twist of degree δ to take the point Q in a particular form which allows to perform some computations more efficiently in the sub-field $\mathbb{F}_{q^{k/\delta}}$ instead of $\mathbb{F}_{q^k}$. More precisely, if E is an elliptic curve defined over $\mathbb{F}_q$, E' its twist of degree δ defined over $\mathbb{F}_{q^{k/\delta}}$ and $\psi : E' \longrightarrow E$ the isomorphism between E and E', then the point Q is taken as the image by ψ of a point on the twisted curve $E'(\mathbb{F}_{q^{k/\delta}})$. In this case, the present form of Q allows many computations either for additon of points or evaluation of the Miller functions to be done more efficiently in the subfield $\mathbb{F}_{q^{k/\delta}}$. For example in the present case of this work and from Proposition D.2.2, instead of taking Q with full coordinates in $\mathbb{F}_{q^k}$, it can be taken in the form $(\omega X : Y : Z)$ where $X, Y, Z \in \mathbb{F}_{q^{k/4}}$. In this work, we use this technic for the computation of the Tate, Ate, twisted Ate and Optimal pairings. As a consequence, the twists can be use to eliminate the denominator of the function $h_{R,S}$ in the Miller algorithm. See Section D.3.1 for applications.

D.3 The Tate Pairing and Twisted Ate Pairing Computation on $E_d : Y^2 = dX^4 + Z^4$

Wang et al. in [185] considered pairings on Jacobi quartics and gave the geometric interpretation of the group law. We use a different way, namely the birational equivalence between Jacobi quartic curves and Weierstrass curves, of obtaining the formulas. We specialise to the particular curves $E_d : Y^2 = dX^4 + Z^4$ to obtain better results for these up to 39% improvement compared to results in [185]. The results in this section are from [58].

Given two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ on the Weierstrass curve $W_d : y^2 = x^3 - 4dx$ such that $P_3 = (x_3, y_3) = P_1 + P_2$, consider $R = (X_1, Y_1, Z_1)$, $S = (X_2, Y_2, Z_2)$ and $(X_3, Y_3, Z_3) = (X_1, Y_1, Z_1) + (X_2, Y_2, Z_2)$ the corresponding points on the Jacobi

quartic E_d. To derive the Miller function $h_{R,S}(X, Y, Z)$ for E_d, we first write the Miller function $h_{P_1,P_2}(x, y)$ on the Weierstrass curve W_d :

$$h_{P_1,P_2}(x, y) = \frac{y - \lambda x - \alpha}{x - x_3},$$

where $\lambda = \dfrac{y_2 - y_1}{x_2 - x_1}$ if $P_1 \neq P_2$, $\lambda = \dfrac{3x_1^2 - 4d}{2y_1}$ if $P_1 = P_2$ and $\alpha = y_1 - \lambda x_1$. Using the birational equivalence, the Miller function for the Jacobi quartic $E_d : Y^2 = dX^4 + Z^4$ is given by $h_{R,S}(X, Y, Z) = h_{P_1,P_2}(\varphi(X, Y, Z))$. We have :

$$h_{R,S}(X, Y, Z) = \frac{4X_3^2X^2}{2X_3^2(Y + Z^2) - 2X^2(Y_3 + Z_3^2)} \left(\frac{ZY+Z^3}{X^3} - \frac{1}{2}\lambda \left(\frac{Y + Z^2}{X^2} \right) - \frac{\alpha}{4} \right).$$

where

$$\lambda = \begin{cases} \dfrac{-2X_1^3Z_2(Y_2 + Z_2^2) + 2X_2^3Z_1(Y_1 + Z_1^2)}{X_1X_2[-X_1^2(Y_2 + Z_2^2) + X_2^2(Y_1 + Z_1^2)]} & \text{if } P_1 \neq P_2, \\ \dfrac{Y_1 + 2Z_1^2}{X_1Z_1} & \text{if } P_1 = P_2, \end{cases} \tag{D.3.1}$$

and

$$\alpha = \begin{cases} \dfrac{-4(Y_1 + Z_1^2)(Y_2 + Z_2^2)(Z_2X_1 - Z_1X_2)}{X_1X_2[-X_1^2(Y_2 + Z_2^2) + X_2^2(Y_1 + Z_1^2)]} & \text{if } P_1 \neq P_2, \\ \dfrac{-2Y_1(Y_1 + Z_1^2)}{X_1^3Z_1} & \text{if } P_1 = P_2. \end{cases} \tag{D.3.2}$$

Remark 14. *It is simple to verify that our formulas obtained by change of variables is exactly the same result obtained by Wang et al. in [185] using the geometric interpretation of the group law.*
Indeed, by setting $x_1 = \frac{X_1}{Z_1}$, $x_2 = \frac{X_2}{Z_2}$, $y_1 = \frac{Y_1}{Z_1^2}$ *and* $y_2 = \frac{Y_2}{Z_2^2}$ *in their Miller function obtained for the curve* $E_{d,\mu} : y^2 = dx^4 + 2\mu x + 1$ *(by taking* $\mu = 0$*), we get exactly the same result that we found above. However, we take an advantage based on our coordinates system to obtain more efficient formulas in pairings computation.*
The correctness of the formulas in this work can be checked at `http://www.prmais.org/Implementation-Pairings-Jacobi.txt`.

D.3.1 Simplification of the Miller Function

We apply the twist technique described in Section D.2.3 to the present case of quartic twist (see the isomorphism in Proposition D.2.2). This enables the point Q in the Tate and twisted Ate pairings computation to be chosen as $(\omega X_Q : Y_Q : Z_Q)$ or $(x_Q\omega, y_Q, 1)$ in affine coordinates where X_Q, Y_Q, Z_Q, x_Q and y_Q are in $\mathbb{F}_{q^{k/4}}$. Thus

$$h_{R,S}(x_Q\omega, y_Q, 1) = \frac{2X_3^2x_Q^2\omega^2}{X_3^2(y_Q+1) - x_Q^2\omega^2(Y_3+Z_3^2)} \left(-\frac{1}{2}\lambda \left(\frac{y_Q+1}{x_Q^2\omega^4} \right) \omega^2 + \left(\frac{y_Q+1}{x_Q^3\omega^4} \right) \omega - \frac{\alpha}{4} \right).$$

Write $-\frac{\alpha}{4} = \frac{A}{D}$ and $-\frac{1}{2}\lambda = \frac{B}{D}$ then

$$h_{R,S}(x_Q\omega, y_Q, 1) = \frac{2X_3^2x_Q^2\omega^2}{D(X_3^2(y_Q+1) - x_Q^2\omega^2(Y_3+Z_3^2))} \left(B \left(\frac{y_Q+1}{x_Q^2\omega^4} \right) \omega^2 + D \left(\frac{y_Q+1}{x_Q^3\omega^4} \right) \omega + A \right).$$

We can easily see that the denominator $D(X_3^2(y_Q+1)-x_Q^2\omega^2(Y_3+Z_3^2))$ and the factor $2X_3^2x_Q^2\omega^2$ of $h_{R,S}$ belong to $\mathbb{F}_{q^{k/2}}$. As $q^{k/2}-1$ divides q^k-1, they are sent to 1 during the final exponentiation (last step in the Algorithm D.1). So they can be discarded in pairing computation and we only have to evaluate

$$\tilde{h}_{R,S}(x_Q\omega, y_Q, 1) = B\left(\frac{y_Q+1}{x_Q^2\omega^4}\right)\omega^2 + D\left(\frac{y_Q+1}{x_Q^3\omega^4}\right)\omega + A.$$

Since $Q=(x_Q\omega, y_Q, 1)$ is fixed during pairing computation, the quantities $\frac{y_Q+1}{x_Q^3\omega^4}$ and $\frac{y_Q+1}{x_Q^2\omega^4}$ can be precomputed in $\mathbb{F}_{q^{k/4}}$, once for all steps. Note that each of the multiplications $D\left(\frac{y_Q+1}{x_Q^3\omega^4}\right)$ and $B\left(\frac{y_Q+1}{x_Q^2\omega^4}\right)$ costs $\frac{k}{4}m_1$, since $A, B, D \in \mathbb{F}_q$.

Efficient computation of the main multiplication in Miller's algorithm

Depending on the form of the function $\tilde{h}_{R,S}$ and the field $\mathbb{F}_{q^k}$, the main multiplication in Miller's algorithm which enables to update the function f can be done efficiently. In this work, the expression of $\tilde{h}_{R,S}$ has a nice form : the term ω^3 is absent and $A \in \mathbb{F}_q$. So, The multiplication by $\tilde{h}_{R,S}$ will be more efficient than the multiplication with an ordinary element of $\mathbb{F}_{q^k}$ (which is denoted m_k)

- If the schoolbook multiplication is used for the multiplication in $\mathbb{F}_{q^k}$, the cost of the multiplication by $\tilde{h}_{R,S}$ is not m_k but $\left(\frac{1}{k}+\frac{1}{2}\right)m_k$. See Appendix D.7 for details.

- if we are using pairing friendly fields for elliptic curves with quartic twists, the embedding degree will be of the form $k=2^i$ (see [82]). Then we follow [122] and the cost of a multiplication or a squaring in the field $\mathbb{F}_{q^k}$ is 3^i multiplications or squaring in $\mathbb{F}_q$ using Karatsuba multiplication method. Thus, the cost of a multiplication by $\tilde{h}_{R,S}$ is $\left(\frac{2\cdot 3^{i-1}+2^{i-1}}{3^i}\right)m_k$. See Appendix D.7 for details.

In the following of this section β stands for $\frac{1}{k}+\frac{1}{2}$ or $\frac{2\cdot 3^{i-1}+2^{i-1}}{3^i}$ so that the cost of the multiplication of the function f in the Miller algorithm by $\tilde{h}_{R,S}$ is βm_k instead of m_k for an ordinary multiplication in $\mathbb{F}_{q^k}$.
In what follows, we will compute A, B and D. For efficiency the point is represented by $(X:Y:Z:X^2:Z^2)$ with $Z\neq 0$. This is the first time that this representation is used when $d\neq 1$. Thus we will use the points $P_1=(X_1:Y_1:Z_1:U_1:V_1)$ and $P_2=(X_2:Y_2:Z_2:U_2:V_2)$ where $U_i=X_i^2$, $V_i=Z_i^2$, $i=1,2$.

Remark 15. *Note that if X^2 and Z^2 are known, then expressions of the form XZ can be computed using the formula $((X+Z)^2-X^2-Z^2)/2$. This allows the replacement of a multiplication by a squaring presuming a squaring and three additions are more efficient than a multiplication. The operations concerned with this remark are followed by $*$ in Tables D.1 and D.2.*

Operations	*Values*	*Cost*
$U := U_1^2$	$U = X_1^4$	$1s_1$
$V := V_1^2$	$V = Z_1^4$	$1s_1$
$Z_3 := V - dU$	$Z_3 = Z_1^4 - dX_1^4$	$1mc$
$E := ((X_1 + Z_1)^2 - U_1 - V_1)/2$ *	$E = X_1 Z_1$	$1s_1$ (or $1m_1$)
$D := 2U_1 E$	$D = 2X_1^3 Z_1$	$1m_1$
$A := (2Y_1 + V_1)^2/4 - U$	$A = Y_1(Y_1 + Z_1^2)$	$1s_1$
$B := -U_1(Y_1 + 2V_1)$	$B = -X_1^2(Y_1 + 2Z_1^2)$	$1m_1$
$X_3 := 2EY_1$	$X_3 = 2X_1 Y_1 Z_1$	$1m_1$
$V_3 := Z_3^2$	$V_3 = Z_3^2$	$1s_1$
$Y_3 := 2V - Z_3$	$Y_3 = dX_1^4 + Z_1^4 = Y_1^2$	-
$Y_3 := 2Y_3^2 - V_3$	$Y_3 = 2Y_1^4 - Z_3^2$	$1s_1$
$U_3 := X_3^2$	$U_3 = X_3^2$	$1s_1$
Total cost : $3m_1 + 7s_1 + 1mc$ (or $4m_1 + 6s_1 + 1mc$)		

TABLE D.1 – Combined formulas for the doubling step.

D.3.2 Doubling Step in the Miller Algorithm

When $P_1 = P_2$, from Equations (D.3.1) and (D.3.2), we have

$$\begin{aligned} A &= Y_1(Y_1 + Z_1^2), \\ B &= -X_1^2(Y_1 + 2Z_1^2), \\ D &= 2X_1^3 Z_1. \end{aligned}$$

The computation of A, B, D and the point doubling can be done using the algorithm in Table D.1 with $3m_1 + 7s_1 + 1mc$ (or $4m_1 + 6s_1 + 1mc$ according to the Remark 15). Thus, the doubling step in the Miller algorithm requires a total of $\beta m_k + 1s_k + \left(\frac{k}{2} + 3\right) m_1 + 7s_1 + 1mc$ (or $\beta m_k + 1s_k + \left(\frac{k}{2} + 4\right) m_1 + 6s_1 + 1mc$).

D.3.3 Addition step in the Miller algorithm

When $P_1 \neq P_2$, from Equations (D.3.1) and (D.3.2), we have

$$\begin{aligned} A &= (Y_1 + Z_1^2)(Y_2 + Z_2^2)(Z_1 X_2 - Z_2 X_1), \\ B &= X_1^3 Z_2(Y_2 + Z_2^2) - X_2^3 Z_1(Y_1 + Z_1^2), \\ D &= X_1 X_2[-X_1^2(Y_2 + Z_2^2) + X_2^2(Y_1 + Z_1^2)]. \end{aligned}$$

Using the algorithm in Table D.2 the computation of A, B, D and the point addition can be done in $12m_1 + 11s_1 + 1mc$ (or $18m_1 + 5s_1 + 1mc$ according to Remark 15). Applying mixed addition ($Z_2 = 1$) which can always be done in our case, this cost is reduced to $12m_1 + 7s_1 + 1mc$ (or $15m_1 + 4s_1 + 1mc$).

Thus, the addition step in the Miller algorithm requires a total of $\beta m_k + \left(\frac{k}{2} + 12\right) m_1 + 7s_1 + 1mc$ (or $\beta m_k + \left(\frac{k}{2} + 15\right) m_1 + 4s_1 + 1mc$).

Operations	*Values*	*Cost*
$U \leftarrow Y_1 + V_1$	$U = Y_1 + Z_1^2$	-
$V \leftarrow Y_2 + V_2$	$V = Y_2 + Z_2^2$	-
$R \leftarrow ((Z_2 + X_1)^2 - V_2 - U_1)/2$ *	$R = Z_2 X_1$	$1s_1$ (or $1m_1$)
$S \leftarrow ((Z_1 + X_2)^2 - V_1 - U_2)/2$ *	$S = Z_1 X_2$	$1s_1$ (or $1m_1$)
$A \leftarrow S - R$	$A = Z_1 X_2 - Z_2 X_1$	-
$A \leftarrow AV$	$A = (Y_2 + Z_2^2)(Z_1 X_2 - Z_2 X_1)$	$1m_1$
$A \leftarrow AU$	$A = (Y_1 + Z_1^2)(Y_2 + Z_2^2)(Z_1 X_2 - Z_2 X_1)$	$1m_1$
$U \leftarrow U_2 U$	$U = X_2^2(Y_1 + Z_1^2)$	-
$V \leftarrow U_1 V$	$V = X_1^2(Y_2 + Z_2^2)$	$1m_1$
$B \leftarrow RV - SU$	$B = X_1^3 Z_2(Y_2 + Z_2^2) - X_2^3 Z_1(Y_1 + Z_1^2)$	$2m_1$
$D \leftarrow ((X_1 + X_2)^2 - U_1 - U_2)/2$ *	$D = X_1 X_2$	$1s_1$ (or $1m_1$)
$E \leftarrow dD^2$	$E = d(X_1 X_2)^2$	$1mc + 1s_1$
$D \leftarrow D(U - V)$	$D = X_1 X_2[-X_1^2(Y_2 + Z_2^2) + X_2^2(Y_1 + Z_1^2)]$	$1m_1$
$X_3 \leftarrow (R + S)(R - S)$	$X_3 = X_1^2 Z_2^2 - Z_1^2 X_2^2$	$1m_1$
$W_1 \leftarrow ((X_1 + Z_1)^2 - U_1 - V_1)/2$ *	$W_1 = X_1 Z_1$	$1s_1$ (or $1m_1$)
$W_2 \leftarrow ((X_2 + Z_2)^2 - U_2 - V_2)/2$ *	$W_2 = X_2 Z_2$	$1s_1$ (or $1m_1$)
$Z_3 \leftarrow W_1 Y_2 - W_2 Y_1$	$Z_3 = X_1 Z_1 Y_2 - X_2 Z_2 Y_1$	$2m_1$
$U \leftarrow Y_1 Y_2$	$U = Y_1 Y_2$	$1m_1$
$V \leftarrow ((Z_1 + Z_2)^2 - V_1 - V_2)/2$ *	$V = Z_1 Z_2$	$1s_1$ (or $1m_1$)
$V \leftarrow V^2 + E$	$V = (Z_1 Z_2)^2 + d(X_1 X_2)^2$	$1s_1$
$E \leftarrow (R - S)^2$	$E = (X_1 Z_2 - X_2 Z_1)^2$	$1s_1$
$U_3 \leftarrow X_3^2$	$U_3 = X_3^2$	$1s_1$
$V_3 \leftarrow Z_3^2$	$V_3 = Z_3^2$	$1s_1$
$Y_3 \leftarrow E(U + V) - V_3$	$Y_3 = (X_1 Z_2 - X_2 Z_1)^2(Y_1 Y_2 + (Z_1 Z_2)^2 +$ $d(X_1 X_2)^2) - Z_3^2$	$1m_1$
Total cost : $12m_1 + 11s_1 + 1mc$ (or $18m_1 + 5s_1 + 1mc$)		

TABLE D.2 – Combined formulas for the addition step.

D.3.4 Comparison

The comparison of results is summarized in Table D.3 and Table D.4. The costs presented are for one iteration of the Miller algorithm and are both for the Tate and twisted Ate pairings and curves with a quartic twist. In each case, we also present an example of comparison in the cases $k = 8$ and $k = 16$ since these values are the most appropriate for cryptographic applications when a quartic twist is used [82]. In Table D.3, we assume that Schoolbook multiplication method is used for the arithmetic in the extension fields $\mathbb{F}_{q^k}$.

Remark 16. *If we assume that $m_1 = s_1 = mc$ and $k = 16$ then we obtain in this work a theoretical gain of 26% and 27% with respect to Weierstrass curves and previous work on Jacobi quartic curves for the doubling step. Similarly, for the addition step we obtain a theoretical gain of 38% and 39% over Weierstrass and Jacobi quartic curves respectively. In the case $k = 8$, the theoretical gain is 22% and 26% with respect to Weierstrass curves and Jacobi quartic curves for the addition step and 26% for the doubling step, see Table D.3.*

In Table D.4, we assume that Karatsuba method is used for the arithmetic in $\mathbb{F}_{q^k}$ for curves with $k = 2^i$.

Curves	Doubling		Mixed Addition	
Weierstrass (b=0)[46]	$1m_k + 1s_k + (\frac{k}{2}+2)m_1 + 8s + 1mc$		$1m_k + (\frac{k}{2}+9)m_1 + 5s_1$	
Jacobi quartic (a=0)[185]	$1m_k + 1s_k + (\frac{k}{2}+5)m_1 + 6s_1$		$1m_k + (\frac{k}{2}+16)m_1 + 1s_1 + 1mc$	
This work	$(\frac{1}{k}+\frac{1}{2})m_k + 1s_k + (\frac{k}{2}+3)m_1 + 7s_1 + 1mc$		$(\frac{1}{k}+\frac{1}{2})m_k + (\frac{k}{2}+12)m_1 + 7s_1 + 1mc$	
Example 1	$k = 8$	$m_1 = s_1 = mc$	$k = 8$	$m_1 = s_1 = mc$
Weierstrass (b=0)[46]	$98m_1 + 16s_1 + 1mc$	$115m_1$	$77m_1 + 5s_1$	$82m_1$
Jacobi quartic (a=0)[185]	$101m_1 + 14s_1$	$115m_1$	$84m_1 + 1s_1 + 1mc$	$86m_1$
This work	$75m_1 + 15s_1 + 1mc$	$91m_1$	$57m_1 + 6s_1 + 1mc$	$64m_1$
Example 2	$k = 16$	$m_1 = s_1 = mc$	$k = 16$	$m_1 = s_1 = mc$
Weierstrass (b=0)[46]	$386m_1 + 24s_1 + 1mc$	$407m_1$	$273m_1 + 5s_1$	$278m_1$
Jacobi quartic (a=0)[185]	$389m_1 + 22s_1$	$411m_1$	$280m_1 + 1s_1 + 1mc$	$282m_1$
This work	$275m_1 + 23s_1 + 1mc$	$299m_1$	$144m_1 + 27s_1 + 1mc$	$172m_1$

TABLE D.3 – Comparison of our Tate and twisted Ate pairings formulas with the previous fastest formulas using Schoolbook multiplication method

Remark 17. *We assume again that $m_1 = s_1 = mc$. For $k = 8$ and for the doubling step we obtain a theoretical gain of 8% over Weierstrass curves and Jacobi quartic curves (a=0)[185]. For the addition step, the improvement is up to 6% over the result on Jacobi quartic curves in [185]. When $k = 16$ the gain is 11% for the doubling step over Weierstrass curves. The improvement is 16% in addition step over Jacobi quartic curves, see Table D.4.*

Remark 18. *The security and the efficiency of pairing-based systems requires using pairing-friendly curves. The Jacobi models of elliptic curves studied in this work are isomorphic to Weierstrass curves. Thus we can obtain pairing friendly curves of such models using the construction given by Galbraith et al.[86] or by Freeman et al.[82]. Some examples of pairing friendly curves of Jacobi quartic form can be found in [185].*

D.4 Formulas for the Ate Pairing and Optimal Pairing on the Jacobi Quartic Elliptic Curve $Y^2 = dX^4 + Z^4$

In this section, we extend the results of the previous section to the computation of Ate pairing and optimal pairing. Our results show that among known curves with quartic twists, the Jacobi model $Y^2 = dX^4 + Z^4$ offers the best performances for these different pairings. The section is divided as follows : In Section D.4.1, we rewrite the Miller function and the addition formulas for the Ate and optimal pairings. In

Curves	Doubling		Mixed Addition	
Weierstrass (b=0)[46]	$1m_k + 1s_k + (\frac{k}{2}+2)m_1 + 8s + 1mc$		$1m_k + (\frac{k}{2}+9)m_1 + 5s_1$	
Jacobi quartic (a=0)[185]	$1m_k + 1s_k + (\frac{k}{2}+5)m_1 + 6s_1$		$1m_k + (\frac{k}{2}+16)m_1 + 1s_1 + 1mc$	
This work	$\left(\frac{2\cdot 3^{i-1}+2^{i-1}}{3^i}\right)m_k + 1s_k + (\frac{k}{2}+3)m_1 + 7s_1 + 1mc$		$\left(\frac{2\cdot 3^{i-1}+2^{i-1}}{3^i}\right)m_k + (\frac{k}{2}+12)m_1 + 7s_1 + 1mc$	
Example 1	$k = 8$	$m_1 = s_1 = mc$	$k = 8$	$m_1 = s_1 = mc$
Weierstrass (b=0)[46]	$33m_1 + 35s_1 + 1mc$	$69m_1$	$40m_1 + 5s_1$	$45m_1$
Jacobi quartic (a=0)[185]	$36m_1 + 33s_1$	$69m_1$	$47m_1 + 1s_1 + 1mc$	$49m_1$
This work	$29m_1 + 34s_1 + 1mc$	$64m_1$	$38m_1 + 7s_1 + 1mc$	$46m_1$
Example 2	$k = 16$	$m_1 = s_1 = m_c$	$k = 16$	$m_1 = s_1 = m_c$
Weierstrass (b=0)[46]	$91m_1 + 89s_1 + 1mc$	$181m_1$	$98m_1 + 5s_1$	$103m_1$
Jacobi quartic (a=0)[185]	$94m_1 + 87s_1$	$181m_1$	$105m_1 + 1s_1 + 1mc$	$107m_1$
This work	$73m_1 + 88s_1 + 1mc$	$162m_1$	$82m_1 + 7s_1 + 1mc$	$90m_1$

TABLE D.4 – Comparison of our formulas for theTate and twisted Ate pairings with the previous fastest formulas using Karatsuba multiplication method.

Section D.4.2 we give the cost of the Ate pairing. The Section D.4.3 is devoted to a comparative study of these pairings on the curves of Jacobi and Weierstrass forms.

D.4.1 Ate Pairing Computation on $E_d : Y^2 = dX^4 + Z^4$

According to the definition of the Ate and optimal pairing, the point addition and point doubling are performed in $\mathbb{F}_{q^k}$. But thanks to the twist we will consider the points $(\omega X_i : Y_i : Z_i)$ where X_i, Y_i and Z_i belong to $\mathbb{F}_{q^{k/4}}$, $i = 1, 2, 3$ (see Proposition D.2.2). We also know that for Ate and optimal pairings the point P is fixed during computations and has its coordinates in the base field $\mathbb{F}_q$. Thus this point can be taken in affine coordinates $(x_P, y_P, 1)$.

Point addition and point doubling on E_d for Ate and optimal pairings.

We rewrite here formulas for point doubling and point addition on the curve E_d from those in Section D.2.1 with the difference that points have the form $(\omega X_i : Y_i : Z_i)$ where X_i, Y_i and Z_i belong to $\mathbb{F}_{q^{k/4}}$, $i = 1, 2, 3$.

Doubling.

$(\omega X_3 : Y_3 : Z_3) = 2(\omega X_1 : Y_1 : Z_1)$ such that

$$\begin{aligned} X_3 &= 2X_1Y_1Z_1, \\ Z_3 &= Z_1^4 - dX_1^4\omega^4, \\ Y_3 &= 2Y_1^4 - Z_3^2. \end{aligned}$$

Addition.

$(\omega X_3 : Y_3 : Z_3) = (\omega X_1 : Y_1 : Z_1) + (\omega X_2 : Y_2 : Z_2)$ such that

$$\begin{aligned} X_3 &= X_1^2 Z_2^2 - Z_1^2 X_2^2, \\ Z_3 &= X_1 Z_1 Y_2 - X_2 Z_2 Y_1, \\ Y_3 &= (X_1 Z_2 - X_2 Z_1)^2 (Y_1 Y_2 + (Z_1 Z_2)^2 + d\omega^4 (X_1 X_2)^2) - Z_3^2. \end{aligned}$$

The Miller function for Ate and optimal pairings computation on E_d.

The Miller function on the Jacobi quartic E_d is given in Section D.3 :

$$h_{R,S}(X, Y, Z) = \frac{4X_3^2 X^2}{2X_3^2(Y + Z^2) - 2X^2(Y_3 + Z_3^2)} \left(\frac{ZY + Z^3}{X^3} - \frac{1}{2}\lambda \left(\frac{Y + Z^2}{X^2} \right) - \frac{\alpha}{4} \right).$$

We follow the notations of Section D.3.1 by setting $-\frac{\alpha}{4} = \frac{A}{D}$ and $-\frac{1}{2}\lambda = \frac{B}{D}$. When we replace $(X_i : Y_i : Z_i)$ by $(\omega X_i : Y_i : Z_i)$ and $(X : Y : Z)$ by $(x_P, y_P, 1)$, a carefully calculation yields to :

$$h_{R,S}(x_P, y_P, 1) = \frac{2X_3^2 x_P^2}{D\omega^2[X_3^2(y_P + 1) - x_P^2(Y_3 + Z_3^2)]} \left(B\left(\tfrac{y_P+1}{x_P^2} \right) \omega^3 + A\omega + D\omega^4 \left(\tfrac{y_P+1}{x_P^3} \right) \right).$$

The factors A, B and D are exactly the same as in the case of the Tate pairing but with the main difference that they are in $\mathbb{F}_{q^{k/4}}$ instead of $\mathbb{F}_q$. The addition and doubling formulas for $(\omega X_i : Y_i : Z_i)$ where X_i, Y_i and Z_i belong to $\mathbb{F}_{q^{k/4}}$, $i = 1, 2, 3$ clearly show that X_3^2 and $Y_3 + Z_3^2$ are also in $\mathbb{F}_{q^{k/4}}$ such that $\frac{2X_3^2 x_P^2}{D\omega^2[X_3^2(y_P + 1) - x_P^2(Y_3 + Z_3^2)]} \in \mathbb{F}_{q^{k/2}}$. Then it can be discarded in pairing computation thanks to the final exponentiation, as we explained in the case of the Tate pairing. Thus we only have to evaluate

$$\bar{h}_{R,S}(x_P, y_P, 1) = B\left(\frac{y_P + 1}{x_P^2} \right) \omega^3 + A\omega + D\omega^4 \left(\frac{y_P + 1}{x_P^3} \right).$$

Since $P = (x_P, y_P, 1)$ is fixed during pairing computation, the quantities $\frac{(y_P + 1)}{x_P^3}$ and $\frac{(y_P + 1)}{x_P^2}$ can be precomputed in $\mathbb{F}_q$ once for all steps. Note that each of the multiplications $D\left(\frac{y_P + 1}{x_P^3} \right)$ and $B\left(\frac{y_P + 1}{x_P^2} \right)$ costs $\frac{k}{4} m_1$.

Remark 19. *We can use the fact that in the expression of $\bar{h}$ the term ω^2 is absent. In this case, in Miller's algorithm, the cost of the main multiplication in $\mathbb{F}_{q^k}$ is not $1m_k$ but $(3/4)m_k$ if we use schoolbook method and is $(8/9)m_k$ if we use Karatsuba multiplication with pairing friendly curves, i.e $k = 2^i$. See Appendix D.8 for details.*

Remark 20. *Since the coefficients of the Miller function for Ate pairing are the same as for the Tate pairing, these coefficients and points operations can be computed in the same manner it was done in the previous section with the main difference that computations are done in $\mathbb{F}_{q^{k/4}}$.*

D.4.2 Cost of the Ate and Optimal Pairing on E_d

In Table D.5 and Table D.6, we summarise and compare the costs for one iteration for both Ate and optimal Ate pairings on the Jacobi curve $E_d : Y^2 = dX^4 + Z^4$ and on the Weierstrass curve $W_d : y^2 = x^3 - 4dx$. We also present these costs in the cases of elliptic curves of embedding degrees 8 and 16.
In Table D.5 we assume that computations are made in $\mathbb{F}_{q^k}$ using schoolbook method.

Pairings	Doubling		Mixed Addition	
Ate(Q,P) Weierstrass (b=0)[46]	$1m_k + 1s_k + 2m_e + 8s_e + 2em_1 + 1mc$		$1m_k + 9m_e + 5s_e + 2em_1$	
Ate(Q,P) **(This work)**	$3/4m_k + 1s_k + 3m_e + 7s_e + 2em_1 + 1mc$		$3/4m_k + 12m_e + 7s_e + 2em_1 + 1mc$	
Example 1	$k = 8$	$m_1 = s_1 = mc$	$k = 8$	$m_1 = s_1 = mc$
Ate(Q,P) Weierstrass (b=0)[46]	$112m_1 + 24s_1 + 1mc$	$137m_1$	$109m_1 + 10s_1$	$119m_1$
This work	$99m_1 + 22s_1 + 1mc$	$122m_1$	$107m_1 + 14s_1 + 1mc$	$122m_1$
Example 2	$k = 16$	$m_1 = s_1 = m_c$	$k = 16$	$m_1 = s_1 = m_c$
Ate(Q,P) Weierstrass (b=0)[46]	$464m_1 + 48s_1 + 1mc$	$513m_1$	$438m_1 + 20s_1$	$458m_1$
This work	$410m_1 + 44s_1 + 1mc$	$455m_1$	$430m_1 + 28s_1 + 1mc$	$459m_1$

TABLE D.5 – Comparisons of Ate and optimal Ate pairings formulas on Jacobi quartic and Weierstrass elliptic curves using Schoolbook method

In Table D.6 we assume that computations are made in $\mathbb{F}_{q^k}$ using Karatsuba method.

Pairings	Doubling		Mixed Addition	
Ate(Q,P) Weierstrass (b=0)[46]	$1m_k + 1s_k + 2m_e + 8s_e + 2em_1 + 1mc$		$1m_k + 9m_e + 5s_e + 2em_1$	
Ate(Q,P) **(This work)**	$8/9m_k + 1s_k + 3m_e + 7s_e + 2em_1 + 1mc$		$8/9m_k + 12m_e + 7s_e + 2em_1 + 1mc$	
Example 1	$k = 8$	$m_1 = s_1 = mc$	$k = 8$	$m_1 = s_1 = mc$
Ate(Q,P) Weierstrass (b=0)[46]	$37m_1 + 51s_1 + 1mc$	$89m_1$	$58m_1 + 15s_1$	$73m_1$
Ate(Q,P) **This work**	$37m_1 + 48s_1 + 1mc$	$85m_1$	$64m_1 + 21s_1 + 1mc$	$86m_1$
Example 2	$k = 16$	$m_1 = s_1 = m_c$	$k = 16$	$m_1 = s_1 = m_c$
Ate(Q,P) Weierstrass (b=0)[46]	$107m_1 + 153s_1 + 1mc$	$261m_1$	$170m_1 + 45s_1$	$215m_1$
Ate(Q,P)**This work**	$107m_1 + 144s_1 + 1mc$	$252m_1$	$188m_1 + 63s_1 + 1mc$	$252m_1$

TABLE D.6 – Comparisons of Ate and optimal Ate pairings formulas on Jacobi quartic and Weierstrass elliptic curves using Karatsuba method

Remark 21. *If we assume that $m_1 = s_1 = mc$ and Schoolbook multiplication method is used then for the Ate pairing computation we obtain in this work a theoretical gain of* 11% *with respect to Weierstrass curves for the doubling step. The improvement is* 4% *when Karatsuba method is used. Our addition step is not better. See Table D.5 and Table D.6.*

D.4.3 Comparison

Let us now compare different pairings on Jacobi quartic curves and Weierstrass elliptic curves with quartic twists. Especially we determine the operation counts for the Tate, twisted Ate, Ate and optimal Ate pairings in a full loop of Miller's algorithm, based on the fastest operations counts summarized in Tables D.3, D.4, D.5 and D.6. We suppose that we are in the context of optimized pairing such that we can restricted ourselves to the cost of the doubling step. Indeed, in this case r is chosen to have a lower Hamming weight such that the computation in Miller algorithm can be done quickly by skipping many addition steps. For elliptic curves with embedding degrees $k = 8$, we consider the parameters for 112 bits and 128 bits security level. We also consider elliptic curves with embedding degrees $k = 16$ at 128 bits and 192 bits security levels. These values have been selected such that we obtain approximately the same security level both in the elliptic curve defined over the base field $\mathbb{F}_q$ and in the multiplicative group of the finite field $\mathbb{F}_{q^k}$.
For these parameters we give the approximate number of operations in the base field for all the Miller iterations. For the Miller loop in Ate pairing computation we consider an average trace $t \sim \sqrt{q}$. For the values in Table D.7, we assume that $m_1 = s_1 = mc$. The rows with abbreviation **Kar** means that the values in these rows are obtained using Karatsuba multiplication method whereas the rows started with **Sco** means that the values in these rows are obtained using schoolbook multiplication method. W and J stand for Weierstrass [46] and Jacobi elliptic (this work) curves models respectively, since this work is the first that present the computation of Ate pairing and its variations on Jacobi elliptic curves.

			Tate		twisted Ate		Ate		Optimal Ate	
Parameters	Sec. levels	Arith. in $\mathbb{F}_{q^k}$	W [46]	J (This work)	W [46]	J (This work)	W [46]	J (This work)	W [46]	J (This work)
$k = 8, r \approx 2^{224}$ $q \approx 2^{336}$	112	Kar.	15456	14336	23184	21504	14952	14448	4984	4816
		Sco.	25760	20384	38640	30576	23016	20496	7672	6832
$k = 8, r \approx 2^{256}$ $q \approx 2^{384}$	128	Kar.	17664	16384	26496	24576	17088	16512	5696	5504
		Sco.	29440	23296	44160	34944	26304	23424	8768	7808
$k = 16, r \approx 2^{256}$ $q \approx 2^{320}$	128	Kar.	46336	41472	115840	103680	41760	40320	8352	8064
		Sco.	105216	76544	263040	191360	82080	72800	16416	14560
$k = 16, r \approx 2^{384}$ $q \approx 2^{480}$	192	Kar.	69504	62208	173760	155520	62640	60480	12528	12096
		Sco.	157824	114816	394560	287040	123120	109200	24624	21840

TABLE D.7 – Comparison of the cost of the various Miller algorithms for pairings on Jacobi quartic curves and Weierstrass curves : $s_1 = m_1 = mc$

From the values in Table D.7 we draw the following observation : The different pairings computed in this work are always faster in the Jacobi quartic elliptic curves with respect to the Weierstrass elliptic curves. The gain obtained is up to 27% and depends on the method used for multiplications and the security level.

D.5 Implementation and Example

In this section we consider the familly of elliptic curves of embedding degree 8 described in [82] to verify our formulas and to implement the Tate, Ate and optimal Ate pairings. This familly of curves has the following parameters :

$$\begin{aligned} r &= 82x^4 + 108x^3 + 54x^2 + 12x + 1, \\ q &= 379906x^6 + 799008x^5 + 705346x^4 + 333614x^3 + 88945x^2 + 12636x + 745, \\ t &= -82x^3 - 108x^2 - 54x - 8. \end{aligned}$$

For $x = 24000000000010394$, the values of r, q, the trace t and the curve coefficient d are :

$$\begin{aligned} r &= 27205632000047130716160030618261401480840452517707677193482845476817, \\ q &= 72601167200444660495170346479178932899121731377660276881150532069758156754787842298703647640196322590069, \\ d &= 45375729500277912809481466549486833061951082111037673050719082543598847971742401436689779775122701618793, \\ t &= -113356800000147285043200063789391713609209096429146 0. \end{aligned}$$

We recall that $\mathbb{G}_1 = E(\mathbb{F}_q)[r]$ and $\mathbb{G}_2 = E\left(\overline{\mathbb{F}_q}\right)[r] \cap \mathrm{Ker}(\pi_q - [q])$. To obtain an optimal pairing in the Jacobi quartic curve E_d with embedding degree 8, we follow the approach described by Vercauteren in [183]. Applying the `ShortestVectors()` function in Magma to the lattice

$$L = \begin{pmatrix} r & 0 & 0 & 0 \\ -q & 1 & 0 & 0 \\ -q^2 & 0 & 1 & 0 \\ -q^3 & 0 & 0 & 1 \end{pmatrix},$$

we obtain the following vector

$$V = [c_0, c_1, c_2, c_3] = [x, 0, 0, 3x + 1].$$

An optimal pairing is then given by :

$$\begin{array}{rccc} e_o : & \mathbb{G}_2 \times \mathbb{G}_1 & \to & \mu_r \\ & (Q, P) & \mapsto & \left(f_{x,Q}^{3q^3+1}(P) \cdot H_1\right)^{\frac{q^8-1}{r}}, \end{array}$$

where $H_1 = (\overline{h}_{[x]Q,[x]Q}(P) \cdot \overline{h}_{[x]Q,[2x]Q}(P) \cdot \overline{h}_{[3x]Q,[1]Q}(P))^{q^3}$ and $s_1 = (3x+1)q^3$. Indeed, this is a straightforward application of Theorem D.2.8. From that theorem we have $c_0 = x, c_1 = c_2 = 0, c_3 = 3x+1$ and $s_i = \sum_{j=i}^{3} c_j q^j$. Observe that for our example $s_1 = s_2 = s_3 = c_3 q^3 = (3x+1)q^3$. We then apply Theorem D.2.8 to obtain the following

$$e_o(Q, P) = \left(f_{x,Q}(P) \cdot f_{3x+1,Q}^{q^3}(P) \cdot h_{[s_1]Q,[x]Q}(P) \cdot h_{[s_1]Q,P_\infty}^2(P)\right)^{\frac{q^8-1}{r}}.$$

Observe also that $f_{1,Q} = 1$ and $h^2_{[s_1]Q,P_\infty}(P) = 1$. Also, $h_{[s_1]Q,[x]Q}(P)$ will be sent to 1 during the final exponentiation because from $\lambda = mr = \sum_{i=0}^{l} c_i q^i = x + s_1$, we get $[s_1]Q + [x]Q = P_\infty$. We then apply the Property D.2.1 to express $f_{3x+1,Q}$ in terms of $f_{x,Q}$ as follows : $f_{3x+1,Q} = f^3_{x,Q} \cdot h_{[x]Q,[x]Q} \cdot h_{[x]Q,[2x]Q} \cdot h_{[3x]Q,[1]Q}$. Finally, by using the explanation in Section D.4.1, the function $h_{R,S}$ is simplified to $\overline{h}_{R,S}$. We can also observe that, if x is negative then by using the divisors we can take $f_{x,Q} = 1/(f_{-x,Q} \cdot h_{[x]Q,[-x]Q})$ and $h_{[x]Q,[-x]Q}$ is also sent to 1 during the final exponentiation. We remark that for this example, we have $\log_2(x) \approx 54$ iterations of Miller's algorithm which is equal to $\log_2(r)/\varphi(8)$, and this agree with the definition of an optimal pairing.
The Magma code for the implementation of the Tate, Ate and optimal Ate pairings is available at
`http://www.prmais.org/Implementation-Pairings-Jacobi.txt`.

D.6 Conclusion

In this paper we have computed and implemented the Tate, Ate, twisted Ate and optimal pairings on the Jacobi quartic curve $E_d : Y^2 = dX^4 + Z^4$. The result in Tate pairing computation is a significative improvement up to 39% of the results of Wang et al. [185] on the same curve. Comparatively to the Weierstrass curve, our result is 27% more efficient. Ate pairing, twisted and optimal Ate pairings are computed on this curve for the first time. Our results are 27% more faster than in the case of Weierstrass curves [46]. According to our results the Jacobi quartic curve is then, to date, the best curve among curves with quartic twists which gives the most efficient result in pairings computation.

D.7 Appendix 1 :Cost of the Main Multiplication in Miller's Algorithm for the Tate and Twisted Ate pairings

The main multiplication in Miller's algorithm is of the form $f \cdot \tilde{h}$ where f and $\tilde{h}$ are in $\mathbb{F}_{q^k}$. Since $\mathbb{F}_{q^k}$ is a $\mathbb{F}_{q^{k/4}}$-vector space with basis $\{1, \omega, \omega^2, \omega^3\}$, f and $\tilde{h}$ can be written as : $f = f_0 + f_1\omega + f_2\omega^2 + f_3\omega^3$ and $\tilde{h} = h_0 + h_1\omega + h_2\omega^2 + h_3\omega^3$ with f_i and h_i in $\mathbb{F}_{q^{k/4}}$, $i = 0, 1, 2, 3$. However in our case $h_3 = 0$, $h_0 \in \mathbb{F}_q$ and $k = 2^i$.

Schoolbook method :

A full multiplication $f.\tilde{h}$ costs k^2 multiplications in the base field $\mathbb{F}_q$ using schoolbook method. But thanks to the particular form of h_0 and h_3, each of the multiplications $f_i \cdot h_0$ costs $\frac{k}{4}m_1$ and each of the multiplications $f_i \cdot h_1$, $f_i \cdot h_2$ costs $\frac{k^2}{16}m_1$. The final cost of the product $f \cdot \tilde{h}$ in the base field $\mathbb{F}_q$ is $(8\frac{k^2}{16} + 4\frac{k}{4})m_1 = (\frac{k^2}{2} + k)m_1$. Finally the ratio of the cost in this case by the cost of the general multiplication is $\frac{\frac{k^2}{2}+k}{k^2} = \frac{1}{2} + \frac{1}{k}$.

Karatsuba method :

The computation of $f \cdot \tilde{h}$ is done here using a particular Karatsuba multiplication. Instead of writing $f \cdot \tilde{h}$ in the classical way (see for example Appendix D.8), we write it as follows :

$$f \cdot \tilde{h} = (f_0 + f_1\omega + f_2\omega^2 + f_3\omega^3)(h_0 + h_1\omega + h_2\omega^2) = \\ (f_0 + f_1\omega + (f_2 + f_3\omega)\omega^2)(h_0 + (h_1 + h_2\omega)\omega)$$

In this form, the product is obtained using the following three products computed using a classical Karatsuba multiplication : $h_0(f_0 + f_1\omega)$ which costs $2^{i-1}m_1$, $(f_2 + f_3\omega)(h_1 + h_2\omega)$ which costs $3(3^{i-2})m_1$ and $(f_0 + f_2 + (f_1 + f_3)\omega)(h_1 + (h_0 + h_2)\omega)$ which costs $3(3^{i-2})m_1$. The final cost is then $2 \cdot 3^{i-1} + 2^{i-1}$.
The ratio is $\frac{2 \cdot 3^{i-1} + 2^{i-1}}{3^i}$.

D.8 Appendix 2 : Cost of the Main Multiplication in Miller's Algorithm for the Ate pairing

The main multiplication in Miller's algorithm is of the form $f \cdot \bar{h}$ where f and $\bar{h}$ are in $\mathbb{F}_{q^k}$. Since $\mathbb{F}_{q^k}$ is a $\mathbb{F}_{q^{k/4}}$-vector space with basis $\{1, \omega, \omega^2, \omega^3\}$, f and $\bar{h}$ can be written as : $f = f_0 + f_1\omega + f_2\omega^2 + f_3\omega^3$ and $\bar{h} = h_0 + h_1\omega + h_2\omega^2 + h_3\omega^3$ with f_i and h_i in $\mathbb{F}_{q^{k/4}}$, $i = 0, 1, 2, 3$ and $h_2 = 0$.

Schoolbook method :

A full multiplication $f.\bar{h}$ in $\mathbb{F}_{q^k}$ costs k^2 multiplications in the base field $\mathbb{F}_q$ using schoolbook method. But thanks to the fact that $h_2 = 0$, each of the 12 multiplications $f_i \cdot h_i$ costs $\frac{k^2}{16}m_1$, $i = 0, 1, 2, 3$. Then the total cost of the product $f \cdot \bar{h}$ is $12\frac{k^2}{16}m_1 = \frac{3k^2}{4}m_1$. Finally the ratio of the cost in this case by the cost of the general multiplication is $\frac{\frac{3k^2}{4}}{k^2} = \frac{3}{4}$.

Karatsuba method :

$k = 2^i$. A full multiplication $f.\bar{h}$ in $\mathbb{F}_{q^k}$ is computed using Karatsuba multiplication as follows :

$$f \cdot \bar{h} = (f_0 + f_1\omega + f_2\omega^2 + f_3\omega^3)(h_0 + h_1\omega + h_2\omega^2 + h_3\omega^3) = \\ (f_0 + f_1\omega + (f_2 + f_3\omega)\omega^2)(h_0 + h_1\omega + (h_2 + h_3\omega)\omega^2)$$

In this form, this product is obtained by computing the three products $u_1 = (f_0 + f_1\omega)(h_0 + h_1\omega)$, $v_1 = (f_2 + f_3\omega)(h_2 + h_3\omega)$ and $w_1 = (f_0 + f_2 + (f_1 + f_3)\omega)(h_0 + h_2 + (h_1 + h_3)\omega)$. Applying again Karatsuba multiplication to u_1, v_1 and w_1, this costs $3(3^{i-2})m_1$ for each product such that the cost of the main multiplication $f \cdot \bar{h}$ using Karatsuba is $3^i m_1$.
Now in our case, $h_2 = 0$ so that the computation of v_1 costs only $2(3^{i-2})$ and the total cost for computing $f \cdot \bar{h}$ is $8 \cdot 3^{i-2}m_1$.
The ratio is then 8/9.

Annexe E

A survey of Fault Attacks in Pairing Based Cryptography

Cryptography and Communication 2015 [71]
avec Jacques J.A. Fournier, Louis Goubin et Ronan Lashermes[1]

Abstract : The latest implementations of pairings allow efficient schemes for Pairing Based Cryptography. These make the use of pairings suitable for small and constrained devices (smart phones, smart cards...) in addition to more powerful platforms. As for any cryptographic algorithm which may be deployed in insecure locations, these implementations must be secure against physical attacks, and in particular fault attacks. In this paper, we present the state-of-the-art of fault attacks against pairing algorithms, more precisely fault attacks against the Miller algorithm and the final exponentiation which are the two parts of a pairing calculation.[2]

E.1 Introduction

In 1984, A. Shamir challenged the cryptography community to find a protocol based on the user identity [172]. This challenge was solved nearly twenty years later by D. Boneh and M. Franklin. In 2003, D. Boneh and M. Franklin created an identity-based encryption (IBE) scheme based on pairings [35]. The general scheme of an identity based encryption is described in [35] and several protocols based on pairings have been developed since [115]. A feature of Identity Based protocols is that a computation of a pairing involving the private key and the ciphertext is performed in order to decipher a message. A pairing is a bilinear map e taking as inputs two points P and Q of an elliptic curve. The pairing computation gives the result $e(P, Q)$. Several pairings have been described in the literature. The Weil and the Tate pairing were developed [175] without any considerations for the efficiency of the computation. Once pairings were used to construct protocols, cryptographers sought more

[1]This work was supported in part by French project ANR-12-INSE-0014 "SIMPATIC".

[2]"The final publication is available at Springer via http ://dx.doi.org/10.1007/s12095-014-0114-5".

efficient algorithms. In chronological order, the Duursma and Lee algorithm [61], the Eta [16], Ate, twisted Ate [101], optimal pairings [183] and pairing lattices [100] were discovered. Recently, a construction of pairing over a general abelian variety was proposed in [135]. The latest implementations results [5, 14, 41, 92, 168] of pairing computations are fast enough to consider the use of pairing based protocols in embedded devices. Consequently, it seems fair to wonder if pairing based protocols involving a secret are secure against physical attacks in general and fault attacks in particular. Side channel attacks have been analysed in [190] where they conclude that an efficient countermeasure would be to set the secret as the first parameter. In [119], Kim *et al.* analyse the effect of side channel attacks against pairings over binary fields. According to the recent work of Joux [114], pairings over binary fields are not secure. We focus here on fault attacks against pairings in fields with a large prime characteristic.

Since 2006, several fault attacks against pairings have been proposed. In this article, we will present what are in our opinion the most significant ones. For each attack, we assume that the pairing is used during an Identity Based Protocol. The secret point is stored into a smart card or an electronic device that can be attacked with fault attacks. The location of the secret is in practice not important. Indeed, the equations that leak information about the secret can provide information whether the secret is the first or the second parameter. Often, the attack is easier when the secret is the second parameter. That is why we consider the cases where the first parameter is the secret argument.

The article is organized as follows. We briefly recall the background necessary to understand pairings and IBE in Section E.2. The first fault attack against a pairing was proposed by Page and Vercauteren [159] and is presented in Section E.3.1. Then, we describe the adaptations of the previous attack against the Miller algorithm in Section E.3.2. Whelan et Scott [189] highlighted the fact that pairings without a final exponentiation are more sensitive to a sign change fault attack. After that, El Mrabet [66] generalized the attack of Page and Vercauteren to the Miller algorithm used to compute all the recent optimizations of pairings. Another method is adopted in [9], based on instruction skips, and presented in Section E.3.2. In [125], Lashermes *et al.* proposed a fault attack against the final exponentiation during a Tate-like pairing. Their attack is described in Section E.4. In Section E.5, we present the attack against the pairing defined over a general abelian variety. Finally, we conclude in Section E.6.

E.2 Background on pairings

In this section, the definition and construction of a pairing is presented. We briefly recall the interest of pairing based cryptography and we present the security issues.

E.2.1 Short introduction to pairings

We consider pairings defined over an elliptic curve $E(\mathbb{F}_p)$, for p a large prime number. The point at infinity of E is denoted P_∞. In order to illustrate the attacks, we

consider the short Weierstrass equation of E which is in Jacobian coordinates : $Y^2 = X^3 + aXZ^4 + bZ^6$, with $a, b \in \mathbb{F}_p$.

Remark 22. *Pairings can be defined for $E(\mathbb{F}_q)$, with q a power of a prime number. In practice the small characteristics are not secure [114] so we do not take these cases into consideration. Today, a pairing is not constructed over $\mathbb{F}_q$, for q a power of a medium prime. As far as we know it has never been realized in practice but it could be interesting for efficiency reasons. Nevertheless, the attacks we describe here can be adapted to these cases.*

Remark 23. *The equation of E does not influence the scheme of the attacks described in this paper. We describe them considering the short Weierstrass equation, but the same attack can be adapted for any other kind of equation like Edwards [88] for instance. Furthermore, a recent work [135] proposes the computation of pairings over theta functions. We show in Section E.5 that the pairing defined over theta functions are sensitive to fault attacks too. The attack is also independent from the choice of the coordinates. As the pairing is often more efficient in Jacobian coordinates, we choose to work with those. But the same attacks are effective for affine, projective or any other coordinates [68].*

Let r be a prime number dividing $\text{card}(E(\mathbb{F}_p))$. Let k be the smallest integer such that r divides $(p^k - 1)$. The integer k is called the embedding degree of E with respect to r. Let $\mathbb{G}_1 \subset E(\mathbb{F}_p)$, $\mathbb{G}_2 \subset E(\mathbb{F}_{p^k})$, $\mathbb{G}_3 \subset \mathbb{F}^*_{p^k}$, be three groups of order r.

Définition E.2.1. A pairing is a bilinear and non degenerate function : $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$.

Initially, pairings were exclusively defined as mathematical functions, e.g. the Weil and the Tate pairings [175]. With the development of pairing based cryptography, researchers were aiming for an efficient implementation and this brought out the definition of the Duursma and Lee algorithm [61], the Eta [16], Ate, twisted Ate [101], optimal pairing [183] and pairing lattices [100] (in chronological order and from the less to the most efficient pairing). The Weil and the Tate pairings are constructed using the Miller algorithm [143], as for the Ate, twisted Ate, optimal pairing and pairing lattices. The Duursma and Lee algorithm and the Eta pairing are not based on the Miller algorithm. The most efficient pairings are constructed on the Tate model : Ate, twisted Ate, optimal pairing and pairing lattices. So we only recall here the definition of the reduced Tate pairing. A more complete definition of the Tate pairing can be found in [29, §IX.5].

Définition E.2.2. Let $E(\mathbb{F}_p)$ be an elliptic curve over the finite field $\mathbb{F}_p$ for p a prime number, r a divisor of $\text{card}(E(\mathbb{F}_p))$, k the embedding degree of E relatively to r. Let $\mathbb{G}_1 = E(\mathbb{F}_p)[r]$, $\mathbb{G}_2 = E(\mathbb{F}_{p^k})/rE(\mathbb{F}_{p^k})$ and $\mathbb{G}_3 = \{\mu \in \mathbb{F}_{p^k} \text{ such that } \mu^r = 1\}$. The reduced Tate pairing is defined as

$$\begin{aligned} e_T : \mathbb{G}_1 \times \mathbb{G}_2 &\to \mathbb{G}_3, \\ (P, Q) &\to f_{r,P}(Q)^{\frac{p^k-1}{r}} \end{aligned}$$

where $f_{r,P}(Q)$ is the Miller function defined by the divisor $D = r(P) - (rP) - (r - 1)(P_\infty)$.

The Miller function is computed through the Miller algorithm presented in Algorithm 17 and referenced in [44]. The Miller algorithm is constructed on the double and add scheme using the construction of rP. The Miller algorithm is based on the notion of divisors. We only give here the essential elements for the pairing computation.

The Miller algorithm constructs the rational function $f_{r,P}$ associated to the point P, where P is a generator of $\mathbb{G}_1$; and at the same time, it evaluates $f_{r,P}(Q)$ for a point $Q \in G_2 \subset E(\mathbb{F}_{p^k})$.

Algorithme 17 : Miller(P, Q, r)

Data : $r = (r_n \dots r_0)$(radix 2 representation), $P \in \mathbb{G}_1(\subset E(\mathbb{F}_p))$ and $Q \in \mathbb{G}_2(\subset E(\mathbb{F}_{p^k}))$;
Result : $f_{r,P}(Q) \in \mathbb{G}_3(\subset \mathbb{F}^*_{p^k})$;

$T \leftarrow P$;
$f \leftarrow 1$;
for $i = n-1$ **to** 0 **do**
  $f \longleftarrow f^2 \times \overline{h_1}(Q)$, $h_1(x, y)$ is the equation of the tangent at the point T;
  $T \leftarrow [2]T$;
  if $r_i = 1$ **then**
    $f \longleftarrow f \times h_2(Q)$, $h_2(x, y)$ is the equation of the line (PT);
    $T \leftarrow T + P$;
return f

The Ate, twisted Ate, optimal pairing and pairing lattices are constructed on the model of the Tate pairing. They are composed of one Miller algorithm execution followed by a final exponentiation. They differ by their number of iterations and sometimes by the role of P and Q. More details can be found in [100, 101, 183].

The final exponentiation is used to ensure the uniqueness of the resulting value of two equal pairing computations (e.g. $e(P, [2]Q) = e([2]P, Q)$). The final exponentiation maps the result of the Miller algorithm into the group formed by the r^{th} roots of unity in $\mathbb{F}^*_{p^k}$. The exponentiation $f^{\frac{p^k-1}{r}}$ is often used [171].

The most useful property in pairing based cryptography is its bilinearity : $e([n]P, [m]Q) = e(P, Q)^{nm}$, with n and m integers. Pairings can be used to construct several protocols and in particular those allowing identity based cryptography [115]. During an identity based protocol, one of the argument of the pairing is secret, it can be the point P or the point Q. All the other parameters like the elliptic curve $E(\mathbb{F}_p)$, r, k the embedding degree and the implementation of the pairing are public. We present here existing fault attacks that allow the recovery of the secret.

E.2.2 Identity based cryptography

The first use of pairings was for the cryptanalysis of Elliptic Curves Cryptography : the Weil pairing shifts the discrete logarithm problem (DLP) from an elliptic curve to a finite field. After that the pairing was used to improve existing protocols as tri-partite Diffie Hellman key exchange [113], and to construct original protocol like identity based encryption [35, 29].

The aim of identity based encryption is that a person λ can use his own identity λ as a public key. His private key would be sent to him by a trusted authority T. This trusted authority will have all the private keys related to the identity based protocol. In this protocol, there is no need to maintain a public directory of all public keys linked to the identities. The general scheme of an identity based key exchange is the following.

The public data is an elliptic curve E over a finite field $\mathbb{F}_q$, a pairing e, and a hash function H, this hash function associates a point of $E(\mathbb{F}_q)$ to an identity : $H : \{Identity\} \rightarrow E(\mathbb{F}_q)$. We consider that two persons, Alice and Bob, want to have a common secret key in order to have a secure communication.

With the public data, Alice can compute $Q_B = H(Bob)$ the public key of Bob, and Bob can compute $Q_A = H(Alice)$ the public key of Alice.
Alice and Bob ask the trusted authority to receive their secret key. The secret key is a point of $E(\mathbb{F}_q)$.

The trusted authority chooses s, its own secret key, then it generates $P_A = [s]Q_A$ the secret key of Alice, and $P_B = [s]Q_B$ the secret key of Bob.

Then Alice (respectively Bob) can compute $e(P_A, Q_B)$ (resp. $e(Q_A, P_B)$). By bilinearity, Alice and Bob possess a shared key : $e(Q_A, Q_B)^s$. Indeed :

$$e([s]H(A), H(B)) = e(H(A), [s]H(B)) = e(H(A), H(B))^s$$

The particularity of identity based cryptography is that a potential spy knows the algorithm used, the number of iterations and the exponent. The secret is only one of the argument of the pairing. The secret key does not influence the execution time nor the number of iterations in the algorithm, which is different from RSA protocols.

In this simple protocol, the pairing computation involves the secret point and a public point. If the secret point is discovered by an attacker, he can then impersonate the target.

E.2.3 Fault attacks

The goal of a fault attack is to inject errors during the calculation of an algorithm in order to reveal sensitive data. At first these attacks required a very precise positioning and expensive apparatuses to be performed, but now even some cheap apparatuses allow to perform them [96]. The faults can be performed using a laser, an electromagnetic pulse, power or clock glitches [50, 51, 117].

The effect of a fault can be permanent, i.e. a modification of a value in memory, or transient, i.e. a modification of a data which is not stored into memory at one precise moment.

At the bit level, a fault can be a bit-flip if the value of a bit is complemented. Or it can be stuck-at (0 or 1) if the bit modification depends on its value.

The fault can not only modify the data manipulated but also modify the program execution. As an example in a microcontroller, if a fault occurs on the opcode and modifies it, the executed instruction will be modified. This method gives rise to what

is called an instruction skip fault model where an instruction is skipped by modifying its opcode to a value representing an instruction without effect (*e.g.* NOP) or invalid.

E.3 Fault attacks against the Miller algorithm

In this section we present the existing attacks against the Miller algorithm. We describe in Section E.3.1 an attack against the Duursma and Lee algorithm since it was the first attack against a pairing and, more importantly, all the following attacks are constructed on this scheme. Then in Section E.3.2 are described the attacks against the Miller algorithm.

E.3.1 Attack against the Dursma and Lee algorithm

The Duursma and Lee algorithm is not constructed using the Miller algorithm. But it was the first implementation of a pairing to be attacked. The attack was developed by Page and Vercauteren in [159].

Duursma and Lee [61] define a pairing over hyperelliptic curves and in particular over super singular elliptic curves over finite fields of characteristic 3. For $\mathbb{F}_q$ with $q = 3^m$ and $k = 6$, suitable curves are defined by

$$E : y^2 = x^3 - x + b$$

with $b = \pm 1 \in \mathbb{F}_3$. If $\mathbb{F}_{q^3} = \mathbb{F}_q[\rho]/(\rho^3 - \rho - b)$ and $\mathbb{F}_{q^6} = \mathbb{F}_{q^3}[\sigma]/(\sigma^2 + 1)$. The distortion map $\phi : E(\mathbb{F}_q) \to E(\mathbb{F}_{q^6})$ is defined by $\phi(x, y) = (\rho - x, \sigma y)$. Then, with $\mathbb{G}_1 = \mathbb{G}_2 = E(\mathbb{F}_{3^m})$ and $\mathbb{G}_T = \mathbb{F}_{q^6}$, Algorithm 18 computes an admissible, symmetric pairing.

Algorithme 18 : The Duursma-Lee pairing algorithm.

Input : $P = (x_P, y_P) \in \mathbb{G}_1$ and $Q = (x_Q, y_Q) \in \mathbb{G}_2$.
Output : $e(P, Q) \in \mathbb{G}_3$.

$f \leftarrow 1$;
for $i = 1$ **upto** m **do**
  $x_P \leftarrow x_P^3$, $y_P \leftarrow y_P^3$;
  $\mu \leftarrow x_P + x_Q + b$;
  $\lambda \leftarrow -y_P y_Q \sigma - \mu^2$;
  $g \leftarrow \lambda - \mu\rho - \rho^2$;
  $f \leftarrow f \cdot g$;
  $x_Q \leftarrow x_Q^{1/3}$, $y_Q \leftarrow y_Q^{1/3}$;
return f^{q^3-1};

The attack developed by Page and Vercauteren in [159] consists in modifying the number of iterations during the Duursma and Lee algorithm. The hypotheses to perform the attack are that

- the two inputs parameters (points P and Q) are fixed, one is secret and the other public ;

- the pairing implementation is public ;
- two pairing computations are done, one valid and one faulty.

The analysis of the quotient of the two results gives information about the secret. Indeed, the quotient of the two results cancel terms that are not influenced by the fault. In a first time, Page and Vercauteren described how to recover the secret point if the final exponentiation is not performed (i.e. Line 9 of Algorithm 18). Then they explain how to reverse the final exponentiation for a complete attack.

Attack without the final exponentiation

Let $P = (x_P, y_P)$ be the secret input during the pairing computation and let $Q = (x_Q, y_Q)$ be selected by the attacker. We consider the Duursma and Lee algorithm without the final exponentiation (Line 9).

Let $\overline{e}[\Delta]$ be the execution of Algorithm 18 where the fault replaces the loop bound m (in Line 2) with Δ. Then the result of the Duursma and Lee algorithm without the final exponentiation instead of being a product of polynomials of the form

$$\prod_{i=1}^{m}\left[(-y_P^{3^i}\cdot y_2^{3^{m-i+1}}\sigma-(x_P^{3^i}+x_2^{3^{m-i+1}}+b)^2)-(x_P^{3^i}+x_2^{3^{m-i+1}}+b)\rho-\rho^2\right]$$

is a product of the form

$$\prod_{i=1}^{\Delta}\left[(-y_P^{3^i}\cdot y_2^{3^{m-i+1}}\sigma-(x_P^{3^i}+x_2^{3^{m-i+1}}+b)^2)-(x_P^{3^i}+x_2^{3^{m-i+1}}+b)\rho-\rho^2\right]$$

for a random integer Δ.

If $\Delta = m+1$, then recovering the secret point P is easy. We have two results

$$\begin{aligned} R_1 &= \overline{e}[m](P,Q) \\ R_2 &= \overline{e}[m+1](P,Q) \end{aligned}$$

where R_1 is correct and R_2 is faulty. Let $g_{(i)}$ be the i-th factor of a product produced by the algorithm. The quotient of the two results produces a single factor

$$g_{(m+1)} = (-y_P^{3^{m+1}}\cdot y_2\sigma-(x_P^{3^{m+1}}+x_2+b)^2)-(x_P^{3^{m+1}}+x_2+b)\rho-\rho^2.$$

Given that $\forall z \in \mathbb{F}_q, z^{3^m} = z$, the attacker can easily extract x_P or y_P based on the knowledge of x_Q and y_Q.

In practice, the faulty result Δ cannot be forced to $m+1$. It is more realistic to assume that the fault gives $\Delta = m \pm \tau$ for a random unknown integer τ. As a consequence, the attacker compute two results

$$\begin{aligned} R_1 &= \overline{e}[m \pm \tau](P,Q) \\ R_2 &= \overline{e}[m \pm \tau + 1](P,Q), \end{aligned}$$

and once again, considering the quotient, the attacker obtains a single term $g_{(m\pm\tau+1)}$.

In order to apply the same approach, the attacker should discover the exact value of τ. Indeed, this value is needed to correct the powers of x_P, y_P, x_Q and y_Q. As the implementation of Duursma and Lee algorithm is supposed public, the number of operations performed during the faulty execution leaks the value of τ. Then the attack consists in several faulty executions of Algorithm 18 until we find two results R_1 and R_2 satisfying the requirements. The probability to obtain two values R_1 and R_2 after a realistic number of tests was computed in [66].
The probability to obtain two consecutive numbers after n picks among N integers is

$$P(n,N) = 1 - \frac{B(n,N)}{C_{n+N}^{n}},$$

where

$$\begin{cases} N \leq 0, n > 0, B(n,N) &= 0, \\ \forall N, n = 0, B(n,N) &= 1 \\ B(n,N) &= \sum_{j=1}^{N} \sum_{k=1}^{n} B(n-k, j-2). \end{cases}$$

For instance, for an 8-bits architecture only 15 tests are needed to obtain a probability larger than one half, $P(15, 2^8) = 0.56$, and only 28 for a probability larger than 0.9.

Reversing the final exponentiation

The attack described above is efficient without the final exponentiation. But since the final exponentiation is a part of the Duursma and Lee algorithm, Page and Vercauteren present a method to reverse it. The problem is that given the result $R = e(P,Q)$ the attacker want to recover S, the value obtained Line 7 of Algorithm 18 before the final exponentiation (i.e. $R = S^{q^3-1}$). Given R, the value of S is only determined up to a non zero factor in $\mathbb{F}_{q^3}$. Indeed, the Fermat little theorem implies that $\forall c \in \mathbb{F}_{q^3} \setminus \{0\}$, $c^{q^3-1} = 1$. Furthermore, for one solution S of the equation $X^{q^3-1} - R = 0$, all the other solutions are of the form cS, for $c \in \mathbb{F}_{q^3} \setminus \{0\}$. At first sight, the attacker would not be able to choose the correct value S among the $q^3 - 1$ possibilities. However, given the description of the attack, the attacker does not need to reverse the powering of a full factor, but only a single factor with a special form :

$$R = \frac{R_2}{R_1} = \frac{\overline{e}[m \pm \tau + 1](P,Q)}{\overline{e}[m \pm \tau](P,Q)} = g_{(m\pm\tau+1)}^{q^3-1}.$$

We want to recover $g_{(m\pm\tau+1)}$, in order to find the coordinates of the secret point x_P and y_P.

In order to solve this problem, Page and Vercauteren split it in two

1. a method to compute one valid root of $R = g^{q^3-1}$ for some factor g, and
2. a method to derive the correct value of g from among all possible solutions.

The first problem is solved throughout the method of Lidl and Niederreiter [131] to compute roots of the linear operator $X^{q^3} - R \cdot X$ on the vector space $\mathbb{F}_{q^6}/\mathbb{F}_{q^3}$. They use a matrix representation of the problem to find all the solution of the equation

$X^{q^3-1} - R = 0$. Then in order to find the correct root among the $q^3 - 1$ possibilities, Page and Vercauteren use the specific form of the factors in the product. Indeed, the terms $\rho\sigma$ and $\rho^2\sigma$ do not appear in the correct value and this gives a linear system of equations providing the solution. As the method to reverse the final exponentiation is specific to the Duursma and Lee algorithm, we do not give the equations. They are presented with examples in [77, 159].

E.3.2 Attacks against the Miller algorithm

A specific sign change attack

The first attack against the Miller algorithm was developed by Whelan and Scott [189]. They use the same approach than the attack against Duursma and Lee. They compute two pairing values, one correct and one faulty. However the fault is no longer on the Miller loop bound but into the Miller variable. Whelan and Scott analyse several pairings and study the success of the attack whether the secret is the point P or Q. They consider the case of the Eta pairing [16]. This pairing is defined over super singular curves for small characteristics. Considering the recent result on the discrete logarithm problem [114] and the fact that the attack is based on the scheme of Page and Vercauteren attack, we do not describe it. Whelan and Scott target the Weil pairing. First they try to describe a general fault model : any fault is injected during any iteration of the Miller algorithm. The attacker needs to solve a non linear system and they conclude that it could not be done. So they consider a more specific attack : a sign change fault attack (a single sign bit is flipped [31]). They consider that the attacker modifies the sign of one of the coordinates of the point P or Q. This attack is the most efficient when exactly the last iteration of the Miller algorithm is corrupted. They consider the ratio between a valid and a faulty executions of the Weil pairing and, using the equations, they obtain a linear system in the coordinates of the secret point. In this case, the attack is successful. If the fault is injected earlier in the Miller algorithm, the analysis is more complex, as several square and cubic roots have to be computed, but possible. Then they consider the Tate pairing. As the Tate pairing is also constructed using the Miller algorithm, the attack described for the Weil pairing should be efficient. However, due to the complex final exponentiation they conclude that the Tate pairing is efficiently protected against the sign change fault they propose.

A general fault attack

In [66], El Mrabet considers a fault attack based on the Page and Vercauteren attack [159]. The fault consists in modifying the number of iterations during the execution of the Miller algorithm. As the Miller algorithm is the central step for the Weil, the Tate, Ate, twisted Ate, optimal pairings and pairing lattices, the fault model is valuable for a wide class of pairings. However, the attack targets only the Miller algorithm, the final exponentiation is not reversed cryptanalytically and the author assume that another attack could annihilate it. In Section E.4 we describe a recent attack that reverse the final exponentiation. We describe here the general attack against the Miller algorithm. The difficulty of the attack relates to the

resolution of a non linear system.

El Mrabet considers that the number of iterations in the Miller algorithm is modified by a fault attack and denotes τ the new number of iterations. The value of τ is random but can be determined afterwards if the attacker knows the number of iterations, by monitoring the timing of the computation for example. The goal is to obtain a pair of results, $F_{\tau,P}(Q)$ and $F_{\tau+1,P}(Q)$, of two executions of the Miller algorithm. As in the attack on the Duursma and Lee algorithm, we consider the ratio $\frac{F_{\tau+1,P}(Q)}{F_{\tau,P}(Q)^2}$. Then an identification in the basis of $\mathbb{F}_{p^k}$ leads to a system which reveals the secret point.

Without loss of generality, we describe the attack when the embedding degree of the curve is $k = 4$. This allows the description of the equation. As the important point of the method is the identification of the decomposition in the basis of $\mathbb{F}_{p^k}$, it is easily applicable when k is larger than 3. Indeed, $k = 3$ is the minimal value of the embedding degree for which the system obtained can be solved. At the τ-th step, the Miller algorithm calculates $[j]P$. During the $(\tau+1)^{th}$ iteration, it calculates $[2j]P$ and considering the value of the $(\tau+1)^{th}$ bit of $\log_2(r)$, it either stops at this moment, or it calculates $[2j+1]P$.

Let $B = \{1, \xi, \sqrt{\nu}, \xi\sqrt{\nu}\}$ be the basis of $\mathbb{F}_{p^k}$, this basis is constructed using tower extensions. The point $P \in E(\mathbb{F}_p)$ is given in Jacobian coordinates, $P = (X_P, Y_P, Z_P)$ and the point $Q \in E(\mathbb{F}_{p^k})$ is in affine coordinates. As k is even, we can use a classical optimisation in pairing based cryptography which consists in using the twisted elliptic curve to write $Q = (x, y\sqrt{\nu})$, with $x,\ y$ and $\nu \in \mathbb{F}_{p^{k/2}}$ and $\sqrt{\nu} \in \mathbb{F}_{p^k}$[10]. We will consider here only the case where $r_{\tau+1} = 0$. The case where $r_{\tau+1} = 1$ can be treated similarly and is described in [66]. The non linear system in the case $r_{\tau+1} = 1$ is a bit more complex and must be solved using the discriminant theory.

When $r_{\tau+1} = 0$, we have that $F_{\tau+1,P}(Q) = (F_{\tau,P}(Q))^2 \times h_1(Q)$, $[j]P = (X_j, Y_j, Z_j)$, where j is obtained by reading the τ first bits of r and $T = [2j]P = (X_{2j}, Y_{2j}, Z_{2j})$.

Using the equation of h_1, we obtain the following equality :

$$F_{\tau+1,P}(Q) = (F_{\tau,P}(Q))^2 \times$$

$$\left(Z_{2j}Z_j^2 y\sqrt{\nu} - 2Y_j^2 - 3(X_j - Z_j^2)(X_j + Z_j^2)(xZ_j^2 - X_j)\right).$$

Considering that the secret is the point P, we know j, τ, the coordinates of Q. The Miller algorithm gives us $F_{\tau+1,P}(Q)$ and $F_{\tau,P}(Q)$. We calculate the ratio $R = \frac{F_{\tau+1,P}(Q)}{(F_{\tau,P}(Q))^2}$. Using the theoretical form of R and its decomposition in the base B, by identification we can obtain after simplification the following system :

$$\begin{cases} Y_j Z_j^3 = \lambda_2, \\ Z_j^2(X_j^2 - Z_j^4) = \lambda_1, \\ 3X_j(X_j^2 - Z_j^4) + 2Y_j^2 = \lambda_0, \end{cases}$$

where we know the three values λ_0, λ_1 and λ_2.

The resolution[66] of this non linear system gives the following equation :

$$(\lambda_0^2 - 9\lambda_1^2)Z^{12} - (4\lambda_0\lambda_2^2 + 9\lambda_1^3)Z^6 + 4\lambda_1^4 = 0.$$

Solving the equation in Z_j, we find at most $24 = 12 \times 2 \times 1$ possible triplets (X_j, Y_j, Z_j) for the coordinates of the point $[j]P$. Once we have the coordinates of $[j]P$, to find the possible points P, we have to find j' the inverse of j modulo r, and then calculate $[j'][j]P = [j'j]P = P$. Using the elliptic curve equation, we eliminate triplets that do not lie on E. Then we just have to perform the Miller algorithm with the remaining points and compare with the result obtained with the secret point P. So we recover the secret point P, in the case where $r_{\tau+1} = 0$. The case of $r_{\tau+1} = 1$ leads also to a non linear system that can be solved using a Grobner basis.

Remark 24. *We present the attack in Jacobian coordinates. As the attack is not dependent on the system of coordinates, it will be successful for other systems. In [66], the affine, projective and Edwards coordinates are also treated. In the paper [188], the authors consider Hessian coordinates.*

Remark 25. *We describe the attack with the secret point being P. If the secret is the point Q the attack is also valid, we just obtain an easier system to solve.*

The attack against the Miller algorithm is efficient. A model of the attack was implemented in [160]. It is fair to wonder if this attack can be applied to a complete pairing. As the Weil pairing consists in two applications of the Miller algorithm, the Weil pairing is sensitive to this attack. For the Tate-like pairings (Ate, twisted Ate,...) the final exponentiation must be cancelled for the attack to be efficient. As the result of the Miller algorithm has no particular form, it seems difficult to cryptanalytically reverse the final exponentiation. As far as we know it has not be done yet. El Mrabet cites several works in microelectronics that would give the result of the Miller algorithm during a Tate-like pairing computation : for example the scan attack [191] or the under voltage technique [2]. We describe in Section E.4 a recent fault attack against the final exponentiation.

Attack against the *if* instruction

In [9], the authors propose a new fault model as well as an implementation of their fault attack.

The *if* skip fault model In the Miller algorithm, the addition step is performed or not according to the bits of r. This decision is usually implemented with an *if* instruction. If an attacker is able to skip an *if* instruction he can avoid the addition step.

This fault model has several advantages. It can target the last iteration only of the Miller algorithm and as a consequence only one fault injection is required to find the value $h_2(Q)$. This is better than altering the counter value, where the attacker had to perform fault attacks until he find the faulty results for two consecutive iterations. Then it is not as easy to develop a countermeasure against it as for an attack on the loop counter. In the latter case, it is enough to check the number of iterations that the chip executed. In the *if* skip case, the number of addition steps is highly dependant on the l value and can vary even if the security level of the parameters do not.

Recovery of $h_2(Q)$ Let $F_P(Q) = f^2 \cdot h_1(Q) \cdot h_2(Q)$ be the result of the (correct) Miller algorithm expressed with the variables of the last iteration. As we are describing the attack for the Tate pairing, the function h_1 is the tangent at the current point $T = \frac{(r-1)}{2}P$ and h_2 is the equation of the vertical line passing through the points P and $T = (r-1)P$. For any other Tate-like pairing, we should substitute to the previous equation r by ρ which is the integer that gives the number of iteration of the Miller algorithm.

If an attacker skip the *if* instruction in the last iteration, he obtains the value $F_P(Q)^* = f^2 \cdot h_1(Q)$.

With a faulty result and a correct one, he can then compute the ratio

$$\frac{F_P(Q)}{F_P(Q)^*} = \frac{f^2 \cdot h_1(Q) \cdot h_2(Q)}{f^2 \cdot h_1(Q)} = h_2(Q). \tag{E.3.1}$$

Finding the secret with $h_2(Q)$ With the value $h_2(Q)$, the attacker still has to find the secret (the point P in our case). The following computations are done for the Tate pairing in particular. In this case the value r is the order of the groups used in the pairing. As a consequence, in the last iteration, the equation $T = -P$ holds.

In affine coordinates in the last iteration, with an embedding degree 2, $h_2(Q) = x_Q - x_P$ since $T = -P$: the line is the vertical passing through P. So knowing the value $h_2(Q)$ the attacker can find x_P with x_Q known. Using the elliptic curve equation, two candidates are possible for the y_P value. By trying the two possible input points in the Miller algorithm, he can find y_P with the comparison of these two Miller results and the correct one.

The result in Jacobian coordinates is slightly different. The equations are computed with an embedding degree 4 and the basis $B = \{1, \xi, \sqrt{\nu}, \xi\sqrt{\nu}\}$. The point P has Jacobian coordinates (x_P, y_P, z_P) and Q has coordinates $(x_Q, y_Q\sqrt{\nu})$.

In the last iteration, the simplified value $h_2(Q)$ is $h_2(Q) = z_P^2 x_Q - x_P$. When the attacker computes the ratio $R = \frac{F_P(Q)}{F_P(Q)^*}$, he finds a value which can be decomposed on the basis B.

$$R = R_0 + R_1\xi + R_2\sqrt{\nu} + R_3\xi\sqrt{\nu}.$$

The decomposition of $h_2(Q)$ on the basis B yields the system

$$R_1 = z_P^2 x_{Q_1} \tag{E.3.2}$$
$$R_2 = z_P^2 x_{Q_0} - x_P, \tag{E.3.3}$$

where $x_Q = x_{Q_0} + x_{Q_1}\xi$.

Since Q is known to the attacker, this system can be solved to provide the values z_P^2 and then x_P. There are 4 possible candidates for the point P which have to be verified by comparing with the correct result of the Miller algorithm.

Remark 26. *In the case of other pairings (Ate,...), the same attack can be applied. The main difference is that we find a point multiple of P : λP for a public integer λ. Indeed, we consider that except the secret point, every detail of the implementation is public.*

An implementation of the attack The authors of this attack [9] implemented their attack on a chip, an ATmega128L, with a laser fault injection. They demonstrated the feasibility of the *if* instruction skip on a dummy algorithm mimicking the structure of the Miller algorithm. After locating the right spot for the laser fault injection, they were able to successfully skip an *if* instruction.

The *if* instruction skip has two big advantages. It easily target a specific iteration in the Miller algorithm. It is possible to combine it with another instruction skip in the final exponentiation in order to realise a full attack on the pairing computation algorithm. But this later possibility is yet to be proven experimentally.

E.3.3 Countermeasures

Several countermeasures can be implemented to prevent a fault attack. They are referred in [77], we briefly recall them here. We can preventively use randomization of the inputs in order to prevent any leakage of information or detect any alteration of the circuit and then abort the pairing computation.

In order to detect any alteration of the computation we can

a) use fault resilient counters to avoid attacks focused on changing the Miller loop bound [158, Section 5.3],

b) implement the algorithm to perform a random number of iterations greater than the correct one [88, Section 4].

c) check intermediate results during the computation : verify that the points are still on the elliptic curve, compare the last point T with $(r-1)P$ [77],

d) duplicate the computation using bilinerarity : $R_1 = e(P,Q)$, $R_2 = e(aP,bQ)$ and check if $R_2 = R_1^{ab}$ [77],

Two realizations of fault attacks are based on the perturbations of the iteration of Miller's algorithm. As a consequence, the countermeasure a) should always be implemented. The countermeasure b) alone does not seem accurate for an efficient implementation of pairings. Indeed, it induces extra computation that do not improve so much the security. The countermeasure c) would be interesting for instance, in the case of a Goubin attack, but as far as we know the Goubin attack has not been developed in the context of pairing based cryptography. No attack used the alteration of the input of a pairing. Consequently this countermeasure seems useless in the context of pairings. The countermeasure d) is the strongest with the drawback of a double pairing computation.

The randomization and blinding methods are both based on the bilinearity of pairings :

α) use the homogeneity property of Jacobian and projective coordinates to represent the point P,

β) use the homogeneity property of Jacobian and projective coordinates to represent the point Q (with a modification of the equations in the Miller algorithm),

γ) randomize the inputs points using a random field element and modify the pairing algorithm in order to cancel out the effects [173],

δ) choose integers a and b such that $ab = 1 \mod (r)$ and compute $e(P,Q) = e(aP, bQ)$ [159],

Ω) choose a random point R such that $S = e(P,R)^{-1}$ is defined and compute $e(P,Q) = e(P,Q+R)S$,

The two blinding methods α and β are the lighter one, only 3 multiplications in the finite field $\mathbb{F}_p$ or 3 multiplications between an element of $\mathbb{F}_p$ and an element of $\mathbb{F}_{p^k}$. Unfortunately, they are not sufficient to prevent a fault attack. Indeed in [126] the authors demonstrate that the blinding of the coordinates using their homogeneity is not sufficient to protect a pairing against fault attacks. The countermeasure γ implies a modification of the Miller algorithm and it could be tricky to find efficient and secure equations. The two countermeasures δ and Ω seem to be the strongest. The method Ω has the drawback of requiring two pairing executions but could also prevent any alteration of the computation. As a consequence, we think that in order to assure a secure implementation of pairings, the countermeasure Ω should be considered.

E.4 A fault attack against the final exponentiation

The main difficulty faced by fault attacks on the pairing is the final exponentiation. Even if efficient schemes are able to reverse the Miller algorithm, they still require the attacker to have access to the result of the Miller algorithm, correct or faulty.

Several possibilities have been proposed to access these values. First, for some exponents (e.g. $q^3 - 1$), it is possible to reverse the final exponentiation by using the structure of the Miller result as shown in [159]. A more implementation dependant approach has been proposed in [66] where the authors propose to realise a scan chain attack or to override completely the final exponentiation to directly read the result of the Miller algorithm.

Despite considered previously unrealistic, multiple fault injections during one execution of an algorithm seem to be more and more feasible with some new results in this direction [30, 179]. This new possibility opens the door to a new scheme where two fault attacks are combined : one to reverse the final exponentiation, one to reverse the Miller algorithm.

Until recently, the final exponentiation was thought to be an efficient countermeasure against the fault attacks on the Miller algorithm since it is mathematically impossible to find the unique preimage of the exponentiation and thus the result of the Miller loop. However in [125], the authors propose a fault attack to reverse the final exponentiation.

E.4.1 Description of the attack

They chose the case were the embedding degree $k = 2d$ is even and they attack the final exponentiation algorithm proposed in [171].

The exponent is $\frac{p^k-1}{r}$ and can be decomposed as $\frac{p^k-1}{r} = (p^d-1)\cdot\frac{p^d+1}{r}$. If the result of the Miller algorithm is noted f, we choose the following notation : $f_2 = f^{p^d-1}$ and $f_3 = f_2^{\frac{p^d+1}{r}}$ (f_3 is the pairing result observed at the end of the computation). Since $f \in \mathbb{F}_{p^k}^*$, f, f_2 and f_3 satisfy the relations

$$f^{p^k-1} = 1\,;\; f_2^{p^d+1} = 1\,;\; f_3^r = 1. \tag{E.4.1}$$

These relations shows that these intermediary values belongs to the groups noted $f_2 \in \mu_{p^d+1}$ and $f_3 \in \mu_r$.

Let $\mathbb{F}_{p^k} = \mathbb{F}_{p^d}[w]/(w^2 - v)$ be the construction rule for the $\mathbb{F}_{p^k}$ extension field. v is a quadratic nonresidue in $\mathbb{F}_{p^d}$ and is a public parameter.

Let $f_2 = g_2 + h_2 \cdot w$ with $g_2, h_2 \in \mathbb{F}_{p^d}$. Then $f_2^{p^d+1} = 1$ implies $g_2^2 - v \cdot h_2^2 = 1$.

E.4.2 First fault

But this equation holds because $f_2 \in \mu_{p^d+1}$. Now if an attacker injects a fault of value $e \in \mathbb{F}_{p^d}$ such that the faulty value f_2^* equals

$$f_2^* = f_2 + e \notin \mu_{p^d+1}. \tag{E.4.2}$$

It is possible to write the fault effect as

$$f_2^* = (g_2 + e) + h_2 \cdot w. \tag{E.4.3}$$

And the value $(f_2^*)^{p^d+1}$ can be computed by the attacker since he can measure the value f_3^* and r :

$$(f_2^*)^{p^d+1} = (f_3^*)^r \in \mathbb{F}_{p^d}. \tag{E.4.4}$$

Moreover,

$$\begin{aligned}(f_2^*)^{p^d+1} &= (g_2+e)^2 - v\cdot h_2^2 \\ &= 1 + 2\cdot e\cdot g_2 + e^2.\end{aligned}$$

If the attacker knows the error value e, he can computes

$$g_2 = \frac{(f_3^*)^l - 1 - e^2}{2\cdot e}. \tag{E.4.5}$$

And deduces the two candidates for h_2

$$h_2^+ = \sqrt{\frac{g_2^2-1}{v}}\,;\; h_2^- = -\sqrt{\frac{g_2^2-1}{v}}. \tag{E.4.6}$$

With one fault, the attacker found the intermediary value f_2 by checking the two candidates and comparing $(f_2^+)^{\frac{p^d+1}{r}}$ and $(f_2^-)^{\frac{p^d+1}{r}}$ with f_3.

E.4.3 Second fault

At this step, the attacker knows f_3 the correct result of the pairing computation and the intermediary value f_2. Let $f = g + h \cdot w$, $f^{-1} = g' + h' \cdot w$ and $f_2 = f^{p^d-1}$. Then we note K the ratio

$$K = \frac{g_2 - 1}{v \cdot h_2} = \frac{h'}{g'} = -\frac{h}{g}. \tag{E.4.7}$$

In order to recover f, the attacker creates a new fault $e_2 \in \mathbb{F}_{p^d}$ during the inversion in the exponentiation by exponent $p^d - 1$.

Then

$$f_2 = f^{p^d-1} = \bar{f} \cdot f^{-1} \text{ and } f_2^* = \bar{f} \cdot (f^{-1} + e_2). \tag{E.4.8}$$

Let Δ_{f_2} be the difference $\Delta_{f_2} = f_2^* - f_2 = \bar{f} \cdot e_2$. Since $e_2 \in \mathbb{F}_{p^d}$, Δ_{f_2} can be written $\Delta_{f_2} = \Delta_{g_2} + \Delta_{h_2} \cdot w$ with $\Delta_{g_2} = e_2 \cdot g$ and $\Delta_{h_2} = -e_2 \cdot h$.

As f_2^* is not in μ_{p^d+1} with high probability, the attacker can compute $(f_2^*)^{p^d+1} = (f_3^*)^r \in \mathbb{F}_{p^d}$.

Here

$$\begin{aligned} (f_3^*)^{p^d+1} &= (g_2 + \Delta_{g_2})^2 - v \cdot (h_2 + \Delta_{h_2})^2 \\ &= (g_2 + e_2 \cdot g)^2 - v \cdot (h_2 - e_2 \cdot h)^2. \end{aligned}$$

Using the relation $h = -g \cdot K$, we obtain

$$g^2 \cdot e_2^2 \cdot (1 - v \cdot K^2) + g \cdot 2 \cdot e_2 \cdot (g_2 - v \cdot K \cdot h_2) + 1 - (f_3^*)^r = 0. \tag{E.4.9}$$

This quadratic equation provides two solutions for g, each one giving only one possibility thanks to K. The attacker has two candidates for f if he knows e_2.

If he does not exactly know the fault values but is able to have a limited number of guesses, he can still find f_2 easily. But in order to find f he will have to inject more faults similar to the second one in order to uniquely determine f.

As a conclusion, with a minimum of two separate faults during two executions (plus one correct execution) of the pairing computation, the attacker is able to reverse the final exponentiation.

A notable fact about this fault attack is that it can be achieved with instruction skip faults. As a consequence it is possible to combine it with a fault on the Miller algorithm, if the attacker can inject two faults in the same execution, in order to achieve a full pairing fault attack.

A major disadvantage to this attack, making it easy to counter, is that the attacker must be able to observe $f_3^* = (f_2^*)^{\frac{p^d+1}{r}}$. But often, since $f_2 \in \mu_{p^d+1}$ is called a unitary element, it is possible to speed up the final exponentiation computation by replacing the inversions in the computation of f_3 by conjugations (which is equivalent to an inversion for unitary elements). As a consequence, the attacker cannot observe f_3^* in this case and he cannot realise the attack.

E.5 Fault attack against pairings over Theta functions

The latest improvement in the computation of pairings was the description of efficient pairing computations in a more general case for any algebraic variety ; and

in particular for pairings over Theta functions. In [135], Lubicz and Robert generalize the notion of the Weil and the Tate pairings to any abelian variety. To do so, they made an explicit link between the Weil and the Tate pairings and the intersection pairing on the degree 1 homology of an abelian variety. The result is a general definition of pairings and they explicit the formulas for the case of level 2 and 4 Theta functions in order to obtain the most efficient algorithm, considering time and memory consumption. Their algorithm to compute a pairing is based on a Montgomery Ladder approach. In [69], El Mrabet scrutinizes the pairings over Theta functions and proposes a successful fault attack. We describe here the attack. We will not present the theory about Theta functions, we refer to [135] for a detailed approach. We will only present the algorithm for the computation of pairings over Theta functions and the attack.

The Tate pairing over Theta functions

For efficiency reasons, the pairing that will be implemented is the Tate pairing (or a variant of the Tate pairing) so we only consider the side channel attacks against the Tate pairing.

Let n, $l \in \mathbb{N}$ with n even and assume that $\gcd(n,l) = 1$. Let A be an abelian variety over $\mathbb{C}$ with period matrix Ω. We represent A as a closed subvariety of $\mathbb{P}^{n^g-1}$ by the way of level n Theta functions and suppose that this embedding is defined over K. Let $\widetilde{A}$ be the pullback of A via the natural projection $\kappa : A^{n^g} \to \mathbb{P}^{n^g-1}$. For $P \in A$, let $\widetilde{P}$ be an affine lift of P that is a point of A^{n^g} such that $\kappa(\widetilde{P}) = P$. Important ingredients of the algorithm in [135] are the Riemann addition formulas. Suppose that the Theta null point $\widetilde{0} = (\theta_i(0))_{i\in\mathbb{Z}(\overline{n})}$ is known.

Théorème E.5.1. *[135, Theorem 1] Suppose that n and l are relatively primes. For X, $Y \in A(\overline{K})$, denoted by $\widetilde{X}$, $\widetilde{Y}$, $\widetilde{X+Y}$ any affine lifts of X, Y and $X+Y$. For $i \in \mathbb{Z}(\overline{n})$, let $\widetilde{X}_i$ be the i^{th} coordinate of $\widetilde{X}$. For $\in \mathbb{N}$ and $i \in \mathbb{Z}(\overline{n})$, let*

$$f_T(\widetilde{X},\widetilde{Y},\widetilde{X+Y},\widetilde{0},l,i) = \frac{ScalarMult(\widetilde{X+Y},\widetilde{X},\widetilde{Y},\widetilde{0},l)_i}{ScalarMult(\widetilde{X},\widetilde{X},\widetilde{0},\widetilde{0},l)_i}\frac{\widetilde{0}_i}{\widetilde{Y}_i}.$$

Then, for $P \in A(K)/[l]A(K)$, $Q \in A[l]$, if we suppose that $\widetilde{0}$, $\widetilde{P}$, $\widetilde{Q}$ and $\widetilde{P+Q}$ are affine lifts of 0, P, Q and $P+Q$ with coordinates in K, then we have for $i \in \mathbb{Z}(\overline{n})$,

$$e_T(P,Q)^n = f_T(\widetilde{Q},\widetilde{P},\widetilde{P+Q},\widetilde{0},l,i),$$

with $e_T(.,.)$ representing the Tate pairing whenever the right hand side is well defined.

The algorithm ScalarMult is composed of a doubling algorithm and a differential addition algorithm given in Figure E.1, where a, b, $\mathcal{A}$ and $\mathcal{B}$ are constants depending on the Theta functions.

If we compare the efficiency of the Tate and of the Weil pairings, the former is more efficient than the later at least for the security levels considered today, when pairings are computed using a Miller algorithm. In the case of Theta functions, the algorithmic complexity of the Tate pairing consists in two applications of the function

Doubling Algorithm	**Differential Addition Algorithm**
Input : A point $P = (x_P : z_P)$.	**Input :** Two points $P = (x_P : z_P)$ and $Q = (x_Q, z_Q)$ on E, $R = (x_R : z_R) = P - Q$, with $x_R z_R \neq 0$.
Output : The double $2P = (x_{2P} : z_{2P})$	**Output :** The point $P + Q = (x_{P+Q} : z_{P+Q})$
1. $x_0 = (x_P^2 + z_P^2)^2$	1. $x_0 = (x_P^2 + z_P^2)(x_Q^2 + z_Q^2)$
2. $z_0 = \frac{\mathcal{A}^2}{\mathcal{B}^2}(x_P^2 - z_P^2)^2$	2. $z_0 = \frac{\mathcal{A}^2}{\mathcal{B}^2}(x_P^2 - z_P^2)(x_Q^2 - z_Q^2)$
3. $x_{2P} = x_0 + z_0$	3. $x_{P+Q} = (x_0 + z_0)/x_R$
4. $z_{2P} = \frac{a}{b}(x_0 - z_0)$	4. $z_{P+Q} = (x_0 - z_0)/z_R$
5. Return $(x_{2P} : z_{2P})$	5. Return $(x_{P+Q} : z_{P+Q})$

FIGURE E.1 – Doubling and Differential Addition Algorithms

ScalarMult, while the Weil pairing consists in four applications of this function described in [135]. It is quite evident that the Tate pairing over Theta functions will always be more efficient than the Weil pairing over the Theta functions. So we study only the weaknesses of the Tate pairing against a fault attack. Nevertheless, the attacks described for the Tate pairing can easily be adapted to the Weil pairing. As a consequence the countermeasure proposed here must be considered also for the implementation of the Weil pairing.

The Tate pairing is composed of two applications of ScalarMult. First of all, we focus on a fault attack against one application of ScalarMult and after that we will consider an attack against the Tate pairing. The same argument can provide the result of a fault attack against the Weil pairing, or any optimization of the Tate pairing namely Ate, twisted Ate or optimal pairings. The function ScalarMult is a Montgomery Ladder composed by the Doubling and Differential Addition algorithms at each step. When the secret is the exponent this algorithm is an efficient countermeasure to side channel attacks. In the case of pairing based cryptography, the secret is not the exponent but one of the parameters of the Montgomery Ladder algorithm. Consequently, the analysis considering side channel attacks against the Montgomery Ladder for the classical use in cryptography (efficient exponentiation) is no more available.

One model of fault attacks in pairing based cryptography consists in forcing the algorithm to stop early by reducing the number of iterations and by finding the results of two consecutive iterations τ and $\tau + 1$. The results of these two executions give equations that allow to find the secret. In the case of pairings over Theta functions, the fault attack consists in finding one of the coordinates involved during the computation of ScalarMult$(\widetilde{P+Q}, \widetilde{Q}, \widetilde{P})$. The ScalarMult algorithm is composed by the doubling and differential addition algorithms, the results of ScalarMult are the coordinates of $\widetilde{P + lQ}$. The fault attack consists in reducing the number of iteration of ScalarMult. The fault attack for a pairing over Theta functions is easier than the classical fault attack in pairing based cryptography. We need only one fault and the result of this faulty execution to find the secret involved in the ScalarMult algorithm.

The results of the pairing are the coordinates of the point $\widetilde{P+lQ}$. We can suppose that we obtain one of the two coordinates, for example the coordinate z. With the z coordinate of the result, we are able to recover the secret argument of the pairing computation.

Suppose that we can recover the coordinate z of the point $\widetilde{P+jQ}$, for $j < l$. As the points P and Q are of order l by construction, the result of the pairing itself cannot give us information. That is why we need to provoke a fault reducing the number of iterations of the ScalarMult algorithm.

Let $z_1 = z_{P+jQ}$, where j is a known integer. The equation of z_1 is the following

$$z_1 = \left[(x_j^2 + z_j^2)(x_P^2 + z_P^2) - \frac{\mathcal{A}^2}{\mathcal{B}^2}(x_j^2 - z_j^2)(x_P^2 - z_P^2)\right]\frac{1}{\overline{z}}, \tag{E.5.1}$$

where

- $P = (x_P, z_P) = (\overline{x}, \overline{z})$ (with the notations introduced above)
- $(j-1)Q = (x_j, z_j)$
- $P + jQ = (x_1, z_1)$
- $\mathcal{A}$ and $\mathcal{B}$ are constants.

We first describe the attack of the algorithm ScalarMult, before considering the fault attack against the whole Tate pairing algorithm.

If the secret is the point P

Suppose that the point P is secret. The fault attack provides us z_1, the values A, B, x_j and z_j are public. All together, they verify the equation

$$\boldsymbol{\lambda} z_P = \boldsymbol{\beta}(x_P^2 + z_P^2) + \boldsymbol{\gamma}(x_P^2 - z_P^2),$$

where the data $(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\gamma})$ are known. The coordinates x_P and z_P are the values we are looking for.

The point P is given in projective coordinates, this equality is correct for any representative of the point P, i.e. for any $\alpha \neq 0$ we have

$$\boldsymbol{\lambda}(\alpha z_P) = \boldsymbol{\beta}((\alpha x_P)^2 + (\alpha z_P)^2) + \boldsymbol{\gamma}((\alpha x_P)^2 - (\alpha z_P)^2).$$

As the coordinates of P are such that $x_P z_P \neq 0$, we can consider that $\alpha = \frac{1}{z_P}$ and write the equation $\boldsymbol{\lambda} = \boldsymbol{\beta}((x'_P)^2 + 1) + \boldsymbol{\gamma}((x'_P)^2 - 1)$, which leads to $(x'_P)^2 = \frac{\boldsymbol{\lambda}-\boldsymbol{\beta}+\boldsymbol{\gamma}}{\boldsymbol{\beta}-\boldsymbol{\gamma}}$.

Up to the sign, we find one coordinate of a representative of the point P and from that point we can find the secret.

If the secret is the point Q

The formulae are symmetric in the coordinates of P and jQ. Following the same scheme, we obtain z_1 for j not equal to the order of Q and that gives the coordinates of a representative of jQ knowing j. To find the coordinates of Q, we just have to compute the inverse of $j \bmod (l)$ and after that we can recover the coordinates of the point Q.

The condition to perform the fault attack when Q is secret is to stop the computation before $j = l$, as Q is a point of order l. This is a simplification of the fault attack against the pairing considering Miller algorithm, because we only need one faulty execution of ScalarMult.

Considering the computation of the Tate pairing

Recall that the algorithm to compute the Tate pairing is

$$e_T = \frac{\text{ScalarMult}(\widetilde{P+Q}, \widetilde{Q}, \widetilde{P}, l)_i}{\text{ScalarMult}(\widetilde{Q}, \widetilde{Q}, \widetilde{0}, l)_i} \frac{\widetilde{0}_i}{\widetilde{P}_i}.$$

The attacks described above for ScalarMult can be directly adapted to the Tate pairing (and also to the Weil pairing). For efficiency reasons, the computation of $\text{ScalarMult}(\widetilde{P+Q}, \widetilde{Q}, \widetilde{P}, l)_i$ and $\text{ScalarMult}(\widetilde{Q}, \widetilde{Q}, \widetilde{0}, l)_i$ would certainly be implemented in parallel. As a consequence, the fault attack forces the algorithm to stop after the same number of iterations and the result $\text{ScalarMult}(\widetilde{P+Q}, \widetilde{Q}, \widetilde{P}, j)_i$ and $\text{ScalarMult}(\widetilde{Q}, \widetilde{Q}, \widetilde{0}, j)_i$, for the same integer j. For both cases, the secret being either P or Q, the homogeneity of projective coordinates is a trapdoor that gives information about the secret. Let P be the secret point and Q be public, then the coordinate $\widetilde{P}_i$ is also secret, but the homogeneity of the projective coordinates allows us to consider that for example the z coordinate is set to 1, exactly like in the attack described above. We just have to be careful and set the same coordinate to 1 in both calls to ScalarMult, the z one for example. The Equation (E.5.1) would give a slightly different system but linear and easily solvable. The method is the same if the point Q is secret.

Countermeasure for the fault attack

Considering that the fault attack uses the homogeneity of the coordinates, this latter property cannot be used as a countermeasure. A solution would be to use the bilinearity of the pairing [77]. Indeed, if we compute the Tate pairing between the points P and Q, the bilinearity induces that

$$e_T(P,Q) = e_T\left(\delta P, (\delta^{-1} \mod (l))Q\right),$$

for a non zero integer δ. The cost of this countermeasure consists in two exponentiations over the variety $A(K)$.

E.6 Conclusion

We presented in this paper the vulnerability to fault attacks of pairing algorithms when used in an Identity Based Protocol. The first attack against Duursma and Lee algorithm target the number of iterations. The final exponentiation in this case can be reversed using cryptanalytic equations. The most efficient pairings are constructed on the Tate model : an execution of the Miller algorithm followed by a final exponentiation. The Miller algorithm and the final exponentiation were separately analysed with respect to fault attacks. The Miller algorithm was attacked by a modification of the number of iterations and by the corruption of the *if* condition during the last iteration. The final exponentiation was attacked using two "independent" errors in the computation.

TABLE E.1 – Summary of the presented attacks. $P(n, N)$ is the probability to obtain two consecutive numbers after n picks among N integers (cf Section E.3.1).

Attack name	Target	Attack path	Fault model
Page and Vercauteren [159]	Duursma and Lee algorithm	Loop counter	Data modification
Whelan and Scott [189]	Miller algorithm	Sign change	Bit-flip
El Mrabet [66]	Miller algorithm	Loop counter	Data modification
Bae *et al.* [9]	Miller algorithm	If skip	Instruction skip
Lashermes *et al.* [125]	Final exponentiation	Group change	Data modification
El Mrabet [69]	Pairing on Theta functions	Loop counter	Data modification

For once it would be interesting to validate all those fault attack schemes on practical implementations running on a embedded chip. In [126], the authors demonstrate the feasibility of two models of fault attacks against the Miller algorithm. They consider the controlled add and the loop skip fault models. During the controlled add attack the authors target experimentally one addition at several moments to recover the secret used during the pairing computation. The loop skip model realizes in practice the theoretical attack developed in [189]. These two attacks focus only on the Miller algorithm. Further work is necessary to develop an attack against a whole Tate-like pairing, including the Miller algorithm and the final exponentiation. For example in [30], the authors achieve this with clock glitches. For that purpose they perform two fault injections during one pairing execution. The first fault allows them to exit the Miller loop after the first iteration. The second fault is used to skip entirely the final exponentiation. Skipping the final exponentiation is possible only if the method is not inlined (without compiler optimizations). In the other case a fault attack on this algorithm (such as the one presented in Section E.4) would be needed as precised by the authors.

We also highlight the fact that a more general pairing constructed over an algebraic variety is sensitive to fault attacks. As a conclusion, we can say that the fault attack is a threat against an identity based protocol and consequently any implementation of pairings should be protected against physical attacks. We describe existing countermeasures to fault attacks. For now the strongest countermeasures

are the one related to the bilinearity of pairings, with the drawback of a double pairing computation.

Bibliographie

[1] Tolga Acar, Kristin E. Lauter, Michael Naehrig, and Daniel Shumow. Affine pairings on ARM. In Michel Abdalla and Tanja Lange, editors, Pairing-Based Cryptography - Pairing 2012, volume 7708 of Lecture Notes in Computer Science, pages 203–209. Springer, 2012.

[2] Ross Anderson and Marcus Kuhn. Tamper resistance – a cautionary note. In The Second USENIX Workshop on Electronic Commerce Proceedings, pages 1–11, 1996.

[3] D. Aranha, K. Karabina, P. Longa, C. H. Gebotys, and J López. Faster explicit formulas for computing pairings over ordinary curves. In Advances in Cryptology EUROCRYPT 2011, volume 6632 of LNCS, pages 48–68. Springer, 2011.

[4] Diego F. Aranha, Paulo S.L.M. Barreto, Patrick Longa, and Jefferson E. Ricardini. The realm of the pairings. In Tanja Lange, Kristin Lauter, and Petr Lisonek, editors, Selected Areas in Cryptography – SAC 2013, volume 8282 of Lecture Notes in Computer Science, pages 3–25. Springer Berlin Heidelberg, 2014.

[5] Diego F. Aranha, Jean-Luc Beuchat, Jérémie Detrey, and Nicolas Estibals. Optimal eta pairing on supersingular genus-2 binary hyperelliptic curves. In Orr Dunkelman, editor, CT-RSA, volume 7178 of Lecture Notes in Computer Science, pages 98–115. Springer, 2012.

[6] Diego F. Aranha, Luis J. Dominguez Perez, Amine Mrabet, and Peter Schwabe. Software Implementation, chapter 11. CRC Press, 2016.

[7] C. Arene, T. Lange, M. Naehrig, and C. Ritzenthaler. Faster computation of the tate pairing. Journal of number theory, vol. 131(5), pp. 842-857, 2011.

[8] R. Avanzi and E. Cesena. Trace Zero Varieties over Fields of Characteristic 2 for Cryptographic Applications. In J. Chaumine, J. Hirschfield, and R. Rolland, editors, Proceedings of SAGA 2007, The first Symposium on Algebraic Geometry and its Applications, Number Theory and its Applications, pages 188–215. World Scientific, 2007.

[9] Kiseok Bae, Sangjae Moon, and Jaecheol Ha. Instruction fault attack on the miller algorithm in a pairing-based cryptosystem. In Innovative Mobile and

Internet Services in Ubiquitous Computing (IMIS), 2013 Seventh International Conference on, pages 167–174, July 2013.

[10] J.C. Bajard and N. El Mrabet. Pairing in cryptography : an arithmetic point of view. In Advanced Signal Processing Algorithms, Architectures, and Implementations XVII, part of the SPIE Optics & Photonics 2007 Symposium (Proceedings of SPIE), volume 6697, pages 66970O.1–66970O.11, August 2007.

[11] Ramachandran Balasubramanian and Neal Koblitz. The improbability that an elliptic curve has subexponential discrete log problem under the Menezes–Okamoto–Vanstone algorithm. J. Cryptology, 11(2) :141–145, 1998.

[12] Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. working paper or preprint, June 2017.

[13] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In EUROCRYPT 2014, volume 8441 of Lecture Notes in Computer Science, pages 1–16. Springer, 2014.

[14] A Barenghi, G. Bertoni, L. Breveglieri, and G. Pelosi. A FPGA coprocessor for the cryptographic Tate pairing over $\mathbb{F}_p$. In Information Technology : New Generations, 2008. ITNG 2008. Fifth International Conference on, pages 112–119, April 2008.

[15] Paulo S. L. M. Barreto, Craig Costello, Rafael Misoczki, Michael Naehrig, Geovandro C. C. F. Pereira, and Gustavo Zanon. Subgroup security in pairing-based cryptography. In LATINCRYPT 2015, volume 9230 of Lecture Notes in Computer Science, pages 245–265. Springer-Verlag, 2015.

[16] Paulo S. L. M. Barreto, Steven D. Galbraith, Colm O'Eigeartaigh, and Michael Scott. Efficient pairing computation on supersingular abelian varieties. Des. Codes Cryptography, 42(3) :239–271, 2007.

[17] Paulo S. L. M. Barreto, Hae Yong Kim, Ben Lynn, and Michael Scott. Efficient algorithms for pairing-based cryptosystems. In CRYPTO 2002, volume 2442 of LNCS, pages 354–368. Springer, 2002.

[18] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. On the selection of pairing-friendly groups. In Proc. of SAC 2003, volume 3006 of LNCS, pages 17–25. Springer, 2003.

[19] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Efficient implementation of pairing-based cryptosystems. J. Cryptology, 17(4) :321–334, 2004.

[20] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In Proc. of SAC 2005, volume 3897 of LNCS, pages 319–331. Springer, 2006.

[21] Paul Barrett. Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In Proceedings on Advances in cryptology—CRYPTO '86, pages 311–323, London, UK, UK, 1987. Springer-Verlag.

[22] Dan Bernstein and Tanja Lange. Explicit-formulas database, 2010. `http://www.hyperelliptic.org/EFD/`.

[23] J-L. Beuchat, N. El Mrabet, L. Fuentes-Castaeda, and F. Rodriguez-Henriquez. Mathematical Background, chapter 2. CRC Press, 2016.

[24] J-L. Beuchat, L. J. Dominguez Perez, S. Duquesne, N. El Mrabet, L. Fuentes-Castaneda, and F. Rodriguez-Henriquez. Arithmetic of Finite Fields, chapter 5. CRC Press, 2016.

[25] Karim Bigou and Arnaud Tisserand. Improving Modular Inversion in RNS Using the Plus-Minus Method, pages 233–249. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[26] Karim Bigou and Arnaud Tisserand. Single base modular multiplication for efficient hardware rns implementations of ecc. In Conference on Cryptographic Hardware and Embedded Systems, page 123Ð140, September 2015.

[27] O. Billet and M. Joye. The jacobi model of an elliptic curve and side-channel analysis. AAECC 2003, LNCS, vol. 2643, pp. 34-42, 2003.

[28] Ian F. Blake, V. Kumar Murty, and Guangwu Xu. Refinements of Miller's algorithm for computing the Weil/Tate pairing. J. Algorithms, 58(2) :134–149, 2006.

[29] Ian F. Blake, Gagiel Seroussi, Nigel Smart, and J. W. S. Cassels. Advances in Elliptic Curve Cryptography (London Mathematical Society Lecture Note Series). Cambridge University Press, New York, NY, USA, 2005.

[30] J. Blömer, R. Gomes da Silva, P. Günther, J. Krämer, and J.-P. Seifert. A practical second-order fault attack against a real-world pairing implementation. In Fault Diagnosis and Tolerance in Cryptography (FDTC), 2014 Workshop on, Sept 2014.

[31] Johannes Blomer, Martin Otto, and Jean-Pierre Seifert. Sign change fault attacks on elliptic curve cryptosystems. In Luca Breveglieri, Israel Koren, David Naccache, and Jean-Pierre Seifert, editors, FDTC, volume 4236 of Lecture Notes in Computer Science, pages 36–52. Springer, 2006.

[32] M. Bodrato. Towards Optimal Toom-Cook Multiplication for Univariate and Multivariate Polynomials in Characteristic 2 and 0. In WAIFI 2007, volume 4547 of Lecture Notes in Computer Science, pages 116–133. Springer, 2007.

[33] D. Boneh and M Franklin. Identity-based encryption from the weil pairing. SIAM Journal of Computing, vol.32(3) pp. 586-615, 2003.

[34] D. Boneh and M.K. Franklin. Identity-Based Encryption from the Weil Pairing. In CRYPTO '01 : Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology, pages 213–229, London, UK, 2001. Springer-Verlag.

[35] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In CRYPTO 2001, volume 2139 of Lecture Notes in Computer Science, pages 213–229. Springer, 2001.

[36] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. SIAM J. Comput., 32(3) :586–615, 2003. Extended abstract in proc. of Crypto 2001.

[37] Dan Boneh, Craig Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In CRYPTO 2005, volume 3621 of Lecture Notes in Computer Science, pages 258–275. Springer, 2005.

[38] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. Journal of Cryptology, 17(4) :297–319, Sep 2004.

[39] J. Boxall, N. El Mrabet, F. Laguillaumie, and P. Le Duc. A Variant of Miller's Formula and Algorithm. In Pairing 2010, volume 6487, pages 417 – 434, London, UK, 2010. Springer-Verlag.

[40] Laura Fuentes Castaneda, Edward Knapp, and Francisco Rodriguez Henriquez. Faster hashing to ${\mathbb G}_2$. In Selected Areas in Cryptography - 18th International Workshop, 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers, pages 412–430, 2011.

[41] Ray C. C. Cheung, Sylvain Duquesne, Junfeng Fan, Nicolas Guillermin, Ingrid Verbauwhede, and Gavin Xiaoxu Yao. FPGA implementation of pairings using residue number system and lazy reduction. In Cryptographic Hardware and Embedded Systems - CHES 2011, volume 6917 of LNCS, pages 421–441. Springer, 2011.

[42] Jaewook Chung and M Anwar Hasan. Asymmetric squaring formulae. In 18$^{\text{th}}$ symposium on Computer Arithmetic, Montpellier, France, IEEE conference publications, pages 113–122, 2007.

[43] Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, and Frederik Vercauteren. Handbook of Elliptic and Hyperelliptic Curve Cryptography, Second Edition. Chapman & Hall/CRC, 2nd edition, 2012.

[44] Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, and Frederik Vercauteren. Handbook of Elliptic and Hyperelliptic Curve Cryptography, Second Edition. Chapman & Hall/CRC, 2nd edition, 2012.

[45] C. Costello, H. Hisil, C. Boyd, J.M.G. Nieto, and K.K.H. Wong. Faster pairings on special weierstrass curves. Pairing 2009, LNCS, vol. 5671 pp. 89-101, 2009.

[46] C. Costello, T. Lange, and M. Naehrig. Faster pairing computations on curves with high-degree twists. PKC 2010, LNCS, vol. 6056 pp. 224-242, 2010.

[47] Craig Costello, Huseyin Hisil, Colin Boyd, Juan Manuel Gonzalez Nieto, and Kenneth Koon-Ho Wong. Faster pairings on special Weierstrass curves. In Proc. of Pairing 2009, volume 5671 of LNCS, pages 89–101. Springer, 2009.

[48] David A. Cox. Primes of the form $x^2 + ny^2$. John Wiley and Sons, 1989.

[49] M.P.L. Das and P. Sarkar. Pairing computation on twisted Edwards form elliptic curves. Pairing 2008, LNCS, vol. 5209, pp. 192-210, 2008.

[50] Elke De Mulder, Sidika Bernard Örs, Bart Preneel, and Ingrid Verbauwhede. Differential power and electromagnetic attacks on a FPGA implementation of elliptic curve cryptosystems. Comput. Electr. Eng., 33(5-6) :367–382, 2007.

[51] Amine Dehbaoui, Jean-Max Dutertre, B. Robisson, and Assia Tria. Electromagnetic Transient Faults Injection on a Hardware and a Software Implementations of AES. In FDTC, pages 7–15. IEEE, 2012.

[52] Augusto Jun Devegili, Colm O'Eigeartaigh, Michael Scott, and Ricardo Dahab. Multiplication and squaring on pairing-friendly fields. IACR Cryptology ePrint Archive, page 471, 2006.

[53] Augusto Jun Devegili, Michael Scott, and Ricardo Dahab. Implementing cryptographic pairings on Barreto-Naehrig curves. In Pairings 2007, volume 4575 of LNCS, pages 197–207. Springer, 2007.

[54] S. Duquesne, N. El Mrabet, S. Haloui, and F. Rondepierre. Choosing and generating parameters for pairing implementation on bn curves. Applicable Algebra in Engineering, Communication and Computing, to appear :1–35, 2017.

[55] S. Duquesne and G. Frey. Background on Pairings, volume pp. 115-124. In Handbook of Elliptic and Hyperelliptic curves cryptography ch. 6, 2005.

[56] Sylvain Duquesne, Nadia El Mrabet, and Emmanuel Fouotsa. Efficient computation of pairings on jacobi quartic elliptic curves. J. Mathematical Cryptology, 8(4) :331–362, 2014.

[57] Sylvain Duquesne, Nadia El Mrabet, Safia Haloui, Damien Robert, and Franck Rondepierre. Choosing Parameters, chapter 10. CRC Press, 2016.

[58] Sylvain Duquesne and Emmanuel Fouotsa. Tate pairing computation on jacobi's elliptic curves. In Michel Abdalla and Tanja Lange, editors, Pairing-Based Cryptography - Pairing 2012 - 5th International Conference, Cologne, Germany, May 16-18, 2012, Revised Selected Papers, volume 7708 of Lecture Notes in Computer Science, pages 254–269. Springer, 2012.

[59] Sylvain Duquesne and Loubna Ghammam. Memory-saving computation of the pairing final exponentiation on BN curves. Groups, Complexity, Cryptology, 2015. To appear.

[60] R. Dutta, R. Barua, and P. Sarkar. Pairing-based cryptography : A survey. Cryptology ePrint Archive, Report 2004/064, 2004.

[61] Iwan M. Duursma and Hyang-Sook Lee. Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$. In Chi-Sung Laih, editor, ASIACRYPT, volume 2894 of Lecture Notes in Computer Science, pages 111–123. Springer, 2003.

[62] Guillaume Tomasini (Editor). Pourquoi les maths ? Ellipse, 2015.

[63] N. El Mrabet. What about vulnerability to a fault attack of the Miller algorithm during an identity based protocol ? In Advances in Information Security and Assurance, volume 5576, pages 122–134, London, UK, 2009. Springer-Verlag.

[64] N. El Mrabet, G. Di Natale, and M.L. Flottes. A practical differential power analysis attack against the miller algorithm. In IEEE PRIME 2009 - 5th Conference on Ph.D. Research in Microelectronics and Electronics, Circuits and Systems Magazine, IEEE Xplore, 2009.

[65] N. El Mrabet and M. Joye. Guide to Pairing-Based Cryptography. Chapman & Hall/CRC Cryptography and Network Security Series. CRC Press, 2017.

[66] Nadia El Mrabet. What about Vulnerability to a Fault Attack of the MillerÕs algorithm During an Identity Based Protocol ? In Advances in Information Security and Assurance, volume 5576 of LNCS, pages 122–134. Springer, 2009.

[67] Nadia El Mrabet. Fault attacks against the miller's algorithm in edwards coordinates. In Samir Kumar Bandyopadhyay, Wael Adi, Tai-Hoon Kim, and Yang Xiao, editors, Information Security and Assurance - 4th International Conference, ISA 2010, Miyazaki, Japan, June 23-25, 2010. Proceedings, volume 76 of Communications in Computer and Information Science, pages 72–85. Springer, 2010.

[68] Nadia El Mrabet. Fault attack against miller's algorithm. IACR Cryptology ePrint Archive, 2011 :709, 2011.

[69] Nadia El Mrabet. Side channel attacks against pairing over theta functions. In Traian Muntean, Dimitrios Poulakis, and Robert Rolland, editors, Algebraic Informatics - 5th International Conference, CAI 2013, Porquerolles, France, September 3-6, 2013. Proceedings, volume 8080 of Lecture Notes in Computer Science, pages 132–146. Springer, 2013.

[70] Nadia El Mrabet and Emmanuel Fouotsa. Failure of the point blinding countermeasure against fault attack in pairing-based cryptography. In Said El Hajji, Abderrahmane Nitaj, Claude Carlet, and El Mamoun Souidi, editors, Codes,

Cryptology, and Information Security - First International Conference, C2SI 2015, Rabat, Morocco, May 26-28, 2015, Proceedings - In Honor of Thierry Berger, volume 9084 of Lecture Notes in Computer Science, pages 259–273. Springer, 2015.

[71] Nadia El Mrabet, Jacques J. A. Fournier, Louis Goubin, and Ronan Lashermes. A survey of fault attacks in pairing based cryptography. Cryptography and Communications, 7(1) :185–205, 2015.

[72] Nadia El Mrabet and Nicolas Gama. Efficient multiplication over extension fields. In Ferruh Özbudak and Francisco Rodríguez-Henríquez, editors, Arithmetic of Finite Fields - 4th International Workshop, WAIFI 2012, Bochum, Germany, July 16-19, 2012. Proceedings, volume 7369 of Lecture Notes in Computer Science, pages 136–151. Springer, 2012.

[73] Nadia El Mrabet, Louis Goubin, Sylvain Guilley, Jacques Fournier, Damien Jauvart, Martin Moreau, Pablo Rauzy, and Franck Rondepierre. Physical Attacks, chapter 12. CRC Press, 2016.

[74] Nadia El Mrabet, Aurore Guillevic, and Sorina Ionica. Efficient multiplication in finite field extensions of degree 5. In Abderrahmane Nitaj and David Pointcheval, editors, Progress in Cryptology - AFRICACRYPT 2011 - 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings, volume 6737 of Lecture Notes in Computer Science, pages 188–205. Springer, 2011.

[75] Nadia El Mrabet and Christophe Nègre. Finite field multiplication combining AMNS and DFT approach for pairing cryptography. In Colin Boyd and Juan Manuel González Nieto, editors, Information Security and Privacy, 14th Australasian Conference, ACISP 2009, Brisbane, Australia, July 1-3, 2009, Proceedings, volume 5594 of Lecture Notes in Computer Science, pages 422–436. Springer, 2009.

[76] Nadia El Mrabet and Christophe Nègre. Finite field multiplication combining AMNS and DFT approach for pairing cryptography. In Proc. of ACISP '09, volume 5594 of LNCS, pages 422–436. Springer, 2009.

[77] Nadia El Mrabet, Dan Page, and Frederik Vercauteren. Fault attacks on pairing-based cryptography. In Marc Joye and Michael Tunstall, editors, Fault Analysis in Cryptography, Information Security and Cryptography, pages 221–236. Springer, 2012.

[78] M. Abdelaziz Elaabid, Olivier Meynard, Sylvain Guilley, and Jean-Luc Danger. Combined side-channel attacks. In WISA, volume 6513 of Lecture Notes in Computer Science, pages 175–190. Springer, 2010.

[79] Junfeng Fan, K. Sakiyama, and I. Verbauwhede. Montgomery modular multiplication algorithm on multi-core systems. In Signal Processing Systems, 2007 IEEE Workshop on, pages 261–266, Oct 2007.

[80] Emmanuel Fouotsa, Nadia El Mrabet, and Aminatou Pecha. Optimal ate pairing on elliptic curves with embedding degree 9, 15 and 27. IACR Cryptology ePrint Archive, 2016 :1187, 2016.

[81] D. Freeman. Constructing pairing-friendly elliptic curves with embedding degree 10. In F. Hess, S. Pauli, and M. Pohst, editors, Algorithmic Number Theory Symposium-ANTS-VII, volume 4076 of Lecture Notes in Computer Science, pages 452–465. Springer, 2006.

[82] David Freeman, Michael Scott, and Edlyn Teske. A taxonomy of pairing-friendly elliptic curves. J. Cryptology, 23(2) :224–280, 2010.

[83] G. Frey, M. Muller, and H. Ruck. The tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. IEEE Transactions on Information Theory, vol. 45(5), pp. 1717-1719, 1999.

[84] G. Frey and H.G. Ruck. A remark concerning m-divisibility and the discrete logarithm problem in the divisor class group of groups. In Mathematics of Computation, volume 62, pages 865–874, Boston, MA, USA, 1994. American Mathematical Society.

[85] S.D. Galbraith. Pairings. London Mathematics Society Lecture Note Series - Cambridge University Press, vol. 317, pp. 183-213, 2005.

[86] S.D. Galbraith, J.F. McKee, and P.C. Valença. Ordinary abelian varieties having small embedding degree. Finite Fields and Their Applications, 13(4) :800–814, 2007.

[87] C. C. F. Pereira Geovandro, Marcos A. Simplício Jr., Michael Naehrig, and Paulo S. L. M. Barreto. A family of implementation-friendly BN elliptic curves. Journal of Systems and Software, 84(8) :1319–1326, 2011.

[88] Santosh Ghosh, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury. Fault attack and countermeasures on pairing based cryptography. International Journal of Network Security (IJNS), 12(1) :26–33, 2011.

[89] Christophe Giraud and Vincent Verneuil. Atomicity improvement for elliptic curve scalar multiplication. In Dieter Gollmann, Jean-Louis Lanet, and Julien Iguchi-Cartigny, editors, Smart Card Research and Advanced Application, volume 6035 of Lecture Notes in Computer Science, pages 80–101. Springer Berlin Heidelberg, 2010.

[90] R. Granger, D. Page, and N. Smart. On small characteristic algebraic tori in pairing based cryptography. LMS Journal of Computation and Mathematics, 9 :64–85, 2006.

[91] Robert Granger and Michael Scott. Faster squaring in the cyclotomic subgroup of sixth degree extensions. In Public Key Cryptography - PKC 2010, 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, May 26-28, 2010. Proceedings, pages 209–223, 2010.

[92] Gurleen Grewal, Reza Azarderakhsh, Patrick Longa, Shi Hu, and David Jao. Efficient implementation of bilinear pairings on ARM processors. In Lars R. Knudsen and Huapeng Wu, editors, Selected Areas in Cryptography, volume 7707 of Lecture Notes in Computer Science, pages 149–165. Springer Berlin Heidelberg, 2013.

[93] Aurore Guillevic. Kim-barbulescu variant of the number field sieve to compute discrete logarithms in finite fields. EllipticNews blog, 2016. `https://ellipticnews.wordpress.com/2016/05/02/`.

[94] S. Guilley. http ://www.dpacontest.org/home/.

[95] Philippe Guillot, Gilles Millérioux, Brandon Dravie, and Nadia El Mrabet. Spectral approach for correlation power analysis. In Said El Hajji, Abderrahmane Nitaj, and El Mamoun Souidi, editors, Codes, Cryptology and Information Security - Second International Conference, C2SI 2017, Rabat, Morocco, April 10-12, 2017, Proceedings - In Honor of Claude Carlet, volume 10194 of Lecture Notes in Computer Science, pages 238–253. Springer, 2017.

[96] Donald Habing. The use of lasers to simulate radiation-induced transients in semiconductor devices and circuits. In IEEE Transactions On Nuclear Science, volume 39, pages 1647–1653, 1992.

[97] Safia Haloui and Edlyn Teske. Pairing-Friendly Elliptic Curves in Guide to Pairing-Based Cryptography, chapter 4. CRC Press, 2016.

[98] Arash Hariri and Arash Reyhani-Masoleh. Bit-serial and bit-parallel montgomery multiplication and squaring over gf. IEEE Transactions on Computers, 58(10) :1332–1345, 2009.

[99] D. Harris, R. Krishnamurthy, M. Anders, S. Mathew, and S. Hsu. An improved unified scalable radix-2 montgomery multiplier. In Computer Arithmetic, 2005. ARITH-17 2005. 17th IEEE Symposium on, pages 172–178, June 2005.

[100] Florian Heß. Pairing lattices. In Proc. of Pairing 2008, volume 5209 of LNCS, pages 18–38. Springer, 2008.

[101] Florian Heß, Nigel P. Smart, and Frederik Vercauteren. The Eta pairing revisited. IEEE Transactions on Information Theory, 52(10) :4595–4602, 2006.

[102] H. Hisil, Wong K.K., G. Carter, and E. Dawson. Jacobi quartic curves revisited. ACISP 2009, LNCS, vol. 5594, pp.452-468, 2009.

[103] Miaoqing Huang, K. Gaj, and T. El-Ghazawi. New hardware architectures for montgomery modular multiplication algorithm. Computers, IEEE Transactions on, 60(7) :923–936, July 2011.

[104] Miaoqing Huang, Kris Gaj, Soonhak Kwon, and Tarek El-Ghazawi. An optimized hardware architecture for the montgomery multiplication algorithm. In Ronald Cramer, editor, Public Key Cryptography Ð PKC 2008, volume

4939 of Lecture Notes in Computer Science, pages 214–228. Springer Berlin Heidelberg, 2008.

[105] Kuan i Lee. Algorithm and VLSI architecture design for H.264/AVC Inter Frame Coding. PhD thesis, National Cheng Kung University, Tainan, Taiwan, 2007.

[106] Sorina Ionica and Antoine Joux. Another approach to pairing computation in Edwards coordinates. In Indocrypt 2008, volume 5365 of LNCS, pages 400–413. Springer, 2008.

[107] Sorina Ionica and Antoine Joux. Pairing the volcano. In Proc. of ANTS-IX, volume 6197 of LNCS, pages 201–218. Springer, 2010.

[108] Keiichi Iwamura, Tsutomu Matsumoto, and Hideki Imai. High-speed implementation methods for rsa scheme. In RainerA. Rueppel, editor, Advances in Cryptology Ñ EUROCRYPTÕ 92, volume 658 of Lecture Notes in Computer Science, pages 221–238. Springer Berlin Heidelberg, 1993.

[109] Keiichi Iwamura, Tsutomu Matsumoto, and Hideki Imai. Systolic-arrays for modular exponentiation using montgomery method. In RainerA. Rueppel, editor, Advances in Cryptology Ñ EUROCRYPTÕ 92, volume 658 of Lecture Notes in Computer Science, pages 477–481. Springer Berlin Heidelberg, 1993.

[110] Damien Jauvart, Jacques J. A. Fournier, Nadia El Mrabet, and Louis Goubin. Improving side-channel attacks against pairing-based cryptography. In Frédéric Cuppens, Nora Cuppens, Jean-Louis Lanet, and Axel Legay, editors, Risks and Security of Internet and Systems - 11th International Conference, CRiSIS 2016, Roscoff, France, September 5-7, 2016, Revised Selected Papers, volume 10158 of Lecture Notes in Computer Science, pages 199–213. Springer, 2016.

[111] Damien Jauvart, Jacques J. A. Fournier, Louis Goubin, and Nadia El Mrabet (Erratum en cours). First practical side-channel attack to defeat point randomization in secure implementations of pairing-based cryptography. In Pierangela Samarati, Mohammad S. Obaidat, and Enrique Cabello, editors, Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) - Volume 4 : SECRYPT, Madrid, Spain, July 24-26, 2017., pages 104–115. SciTePress, 2017.

[112] A. Joux. A one round protocol for tripartite Diffie-Hellman. In ANTS-IV : Proceedings of the 4th International Symposium on Algorithmic Number Theory, pages 385–394, London, UK, 2000. Springer-Verlag.

[113] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. J. Cryptology, 17(4) :263–276, 2004. Extended abstract in proc. of ANTS 2000.

[114] Antoine Joux. A new index calculus algorithm with complexity l(1/4+o(1)) in small characteristic. In Tanja Lange, Kristin Lauter, and Petr Lisonek, editors, Selected Areas in Cryptography – SAC 2013, volume 8282 of Lecture Notes in Computer Science, pages 355–379. Springer Berlin Heidelberg, 2014.

[115] Marc Joye and Gregory Neven. Identity-based Cryptography. Cryptology and information security series. IOS Press, 2009.

[116] Koray Karabina. Squaring in cyclotomic subgroups. Math. Comput., 82(281), 2013.

[117] Chong Hee Kim and J-J Quisquater. Faults, injection methods, and fault attacks. Design & Test of Computers, IEEE, 24(6) :544–545, 2007.

[118] Taechan Kim and Razvan Barbulescu. Extended tower number field sieve : A new complexity for the medium prime case. In CRYPTO 2016, volume 9814 of Lecture Notes in Computer Science, pages 543–571. Springer, 2016.

[119] TaeHyun Kim, Tsuyoshi Takagi, Dong-Guk Han, HoWon Kim, and Jongin Lim. Side channel attacks and countermeasures on pairing based cryptosystems over binary fields. In David Pointcheval, Yi Mu, and Kefei Chen, editors, Cryptology and Network Security, volume 4301 of Lecture Notes in Computer Science, pages 168–181. Springer Berlin Heidelberg, 2006.

[120] Donald E. Knuth. The Art of Computer Programming, Volume 2 (3rd Ed.) : Seminumerical Algorithms. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[121] Neal Koblitz. Elliptic curve cryptosystems. Mathematics of Computation, 48(177) :203–209, January 1987.

[122] Neal Koblitz and Alfred Menezes. Pairing-based cryptography at high security levels. In IMA Int. Conf. Cryptography and Coding, volume 3796 of LNCS, pages 13–36. Springer, 2005.

[123] C.K. Koc, Tolga Acar, and Jr. Kaliski, B.S. Analyzing and comparing montgomery multiplication algorithms. Micro, IEEE, 16(3) :26–33, Jun 1996.

[124] H.T. Kung. Why systolic architectures ? Computer, 15(1) :37–46, Jan 1982.

[125] Ronan Lashermes, Jacques Fournier, and Louis Goubin. Inverting the final exponentiation of tate pairings on ordinary elliptic curves using faults. In Guido Bertoni and Jean-Sebastien Coron, editors, Cryptographic Hardware and Embedded Systems - CHES 2013, volume 8086 of Lecture Notes in Computer Science, pages 365–382. Springer Berlin Heidelberg, 2013.

[126] Ronan Lashermes, Marie Paindavoine, Nadia El Mrabet, Jacques J. A. Fournier, and Louis Goubin. Practical validation of several fault attacks against the miller algorithm. In Assia Tria and Dooho Choi, editors, 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2014, Busan, South Korea, September 23, 2014, pages 115–122. IEEE Computer Society, 2014.

[127] Kristin E. Lauter, Peter L. Montgomery, and Michael Naehrig. An analysis of affine coordinates for pairing computation. In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, Pairing-Based Cryptography - Pairing 2010

- 4th International Conference, Yamanaka Hot Spring, Japan, December 2010. Proceedings, volume 6487 of Lecture Notes in Computer Science, pages 1–20. Springer, 2010.

[128] Duc-Phong Le, Nadia El Mrabet, and Chik How Tan. On Near Prime-Order Elliptic Curves with Small Embedding Degrees, pages 140–151. Springer International Publishing, Cham, 2015.

[129] Duc-Phong Le and ChikHow Tan. Speeding up ate pairing computation in affine coordinates. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, Information Security and Cryptology - ICISC 2012, volume 7839 of Lecture Notes in Computer Science, pages 262–277. Springer Berlin Heidelberg, 2013.

[130] Eunjeong Lee, Hyang-Sook Lee, and Cheol-Min Park. Efficient and generalized pairing computation on abelian varieties. Information Theory, IEEE Transactions on, 55(4) :1793–1803, April 2009.

[131] Rudolf Lidl and Harald Niederreiter. Finite Fields. Number vol. 20,ptie. 1 in Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1997.

[132] Chae Hoon Lim and Pil Joong Lee. A key recovery attack on discrete log-based schemes using a prime order subgroup. In CRYPTO 1997, pages 249–263. Springer-Verlag, 1997.

[133] Xibin Lin, Changan Zhao, Fangguo Zhang, and Yanming Wang. Computing the Ate pairing on elliptic curves with embedding degree $k = 9$. IEICE Transactions, 91-A(9) :2387–2393, 2008.

[134] Victor Lomné, Emmanuel Prouff, and Thomas Roche. Behind the scene of side channel attacks. In ASIACRYPT (1), volume 8269 of Lecture Notes in Computer Science, pages 506–525. Springer, 2013.

[135] David Lubicz and Damien Robert. Efficient Pairing Computation with Theta Functions, pages 251–269. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[136] Kooroush Manochehri, Saadat Pourmozafari, and Babak Sadeghian. Montgomery and rns for rsa hardware implementation. Computing and Informatics, 29(5) :849–880, 2012.

[137] Seiichi Matsuda, Naoki Kanayama, Florian Heß, and Eiji Okamoto. Optimised versions of the Ate and twisted Ate pairings. IEICE Transactions, 92-A(7) :1660–1667, 2009.

[138] A. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. In STOC '91 : Proceedings of the twenty-third annual ACM symposium on Theory of computing, pages 80–89, New York, NY, USA, 1991. ACM.

[139] Alfred Menezes, Palash Sarkar, and Shashank Singh. Challenges with assessing the impact of nfs advances on the security of pairing-based cryptography. Cryptology ePrint Archive, Report 2016/1102, 2016. http://eprint.iacr.org/2016/1102.

[140] S.V. Miller. The Weil pairing, and its efficient calculation. Journal of cryptology, vol. 17(4), pp. 235-261, 2004.

[141] V. Miller. Short programs for functions on curves. Unpublished manuscript available at http://crypto.stanford.edu/miller/miller.pdf., vol., 1986.

[142] Victor. Miller. Use of elliptic curves in cryptography. In HughC. Williams, editor, Advances in Cryptology - CRYPTO 85 Proceedings, volume 218 of Lecture Notes in Computer Science, pages 417–426. Springer Berlin Heidelberg, 1986.

[143] Victor S. Miller. The Weil pairing, and its efficient calculation. J. Cryptology, 17(4) :235–261, 2004.

[144] A. Miyaji, M. Nakabayashi, and S. Takano. New explicit conditions of elliptic curve traces for fr-reduction. In IEICE Trans.,Fundamentails, vol. E84-A :1234–1243, 2001.

[145] P. L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. Mathematics of Computation, 48(177) :243–264, 1987.

[146] P. L. Montgomery. Five, six, and seven-term Karatsuba-like formulae. IEEE Transactions on Computers, 54(3) :362–369, 2005.

[147] Peter L. Montgomery. Modular multiplication without trial division. Mathematics of Computation, 44(170) :519–521, 1985.

[148] A. Mrabet, B. Bouallegue, N. El Mrabet, M. Machhout, and S. Mesnager. Implementation of faster miller over barreto-naehrig curves in jacobian cordinates. In 2014 Global Summit on Computer Information Technology (GSCIT), pages 1–6, June 2014.

[149] A. Mrabet, N. El Mrabet B. Bouallegue, S. Mesnager, and M. Machhout. An efficient and scalable modular inversion and division for public key cryptosystems. In IEEE International Conference on Engineering and Management Information Systems ICEMIS 2017, 2017.

[150] A. Mrabet, N. El Mrabet, R. Lashermes, J-B. Rigaud, B. Bouallegue, S. Mesnager, and M. Machhout. A scalable and systolic architectures of montgomery modular multiplication for public key cryptosystems based on dsps. Extended version to appear in Journal of Hardware and Systems Security (accepted in August 2017), 2017.

[151] Amine Mrabet, Nadia El Mrabet, Ronan Lashermes, Jean-Baptiste Rigaud, Belgacem Bouallegue, Sihem Mesnager, and Mohsen Machhout. High-performance elliptic curve cryptography by using the CIOS method for modular multiplication. In Frédéric Cuppens, Nora Cuppens, Jean-Louis Lanet,

and Axel Legay, editors, Risks and Security of Internet and Systems - 11th International Conference, CRiSIS 2016, Roscoff, France, September 5-7, 2016, Revised Selected Papers, volume 10158 of Lecture Notes in Computer Science, pages 185–198. Springer, 2016.

[152] Amine Mrabet, Nadia El Mrabet, Ronan Lashermes, Jean-Baptiste Rigaud, Belgacem Bouallegue, Sihem Mesnager, and Mohsen Machhout. A scalable and systolic architectures of montgomery modular multiplication for public key cryptosystems based on dsps. In Claude Carlet, M. Anwar Hasan, and Vishal Saraswat, editors, Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings, volume 10076 of Lecture Notes in Computer Science, pages 138–156. Springer, 2016.

[153] M. Naehrig, P. Barreto, and P. Schwabe. On compressible pairings and their computation. In S. Vaudenay, editor, Progress in Cryptology AFRICACRYPT 2008, volume 5023, pages 371–388. Springer, 2008.

[154] Michael Naehrig, Ruben Niederhagen, and Peter Schwabe. New software speed records for cryptographic pairings. In LATINCRYPT 2010, volume 6212 of LNCS, pages 109–123. Springer, 2010.

[155] Yasuyuki Nogami, Masataka Akane, Yumi Sakemi, Hidehiro Katou, and Yoshitaka Morikawa. Integer variable chi-based ate pairing. In Pairing-Based Cryptography - Pairing 2008, pages 178–191, 2008.

[156] National Institute of Standard and technology. Key management, 2007.

[157] Sıddıka Berna Ors, Lejla Batina, Bart Preneel, and Joos Vandewalle. Hardware implementation of a montgomery modular multiplier in a systolic array. In Parallel and Distributed Processing Symposium, 2003. Proceedings. International, pages 8–pp. IEEE, 2003.

[158] Erdinc Ozturk, Gunnar Gaubatz, and Berk Sunar. Tate pairing with strong fault resiliency. In Proceedings of the Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC '07, pages 103–111, Washington, DC, USA, 2007. IEEE Computer Society.

[159] Dan Page and Frederik Vercauteren. A Fault Attack on Pairing-Based Cryptography. Computers, IEEE Transactions on, 55(9) :1075 –1080, sept. 2006.

[160] Jea Hoon Park, Gyo Yong Sohn, and Sang Jae Moon. A simplifying method of fault attacks on pairing computation. IEICE transactions on fundamentals of Electronics, Communications and Computer Sciences, E94-A(6) :1473–1475, 2011.

[161] Guilherme Perin, Daniel Gomes Mesquita, and João Baptista Martins. Montgomery modular multiplication on reconfigurable hardware : Systolic versus multiplexed implementation. Int. J. Reconfig. Comput., 2011 :61–610, January 2011.

[162] T. Plantard. Modular arithmetic for cryptography. PhD thesis, LIRMM, Universite Montpellier 2, 2005.

[163] S. Canard (porteur). http ://www.agence-nationale-recherche.fr/ ?Projet=ANR-12-INSE-0014.

[164] Microsoft Research. MSR ECCLib v2.0. 2015.

[165] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21 :120–126, 1978.

[166] Franck Rondepierre. Revisiting atomic patterns for scalar multiplications on elliptic curves. In Aurélien Francillon and Pankaj Rohatgi, editors, Smart Card Research and Advanced Applications, volume 8419 of Lecture Notes in Computer Science, pages 171–186. Springer International Publishing, 2014.

[167] K. Rubin and A. Silverberg. Torus-Based Cryptography. In D. Boneh, editor, Advances in Cryptology CRYPTO 2003, volume 2729, pages 349–365. Springer, 2003.

[168] Michael Scott. On the efficient implementation of pairing-based protocols. In Liqun Chen, editor, IMA Int. Conf., volume 7089 of Lecture Notes in Computer Science, pages 296–308. Springer, 2011.

[169] Michael Scott and Paulo S. Barreto. Generating More MNT Elliptic Curves. Des. Codes Cryptography, 38 :209–217, February 2006.

[170] Michael Scott and Paulo S. L. M. Barreto. Compressed pairings. In Advances in cryptology—CRYPTO 2004, volume 3152 of Lecture Notes in Comput. Sci., pages 140–156. Springer, Berlin, 2004.

[171] Michael Scott, Naomi Benger, Manuel Charlemagne, Luis J. Dominguez Perez, and Ezekiel J. Kachisa. On the final exponentiation for calculating pairings on ordinary elliptic curves. In Pairings 2009, volume 5671 of LNCS, pages 78–88. Springer, 2009.

[172] A. Shamir. Identity-based cryptosystems and signature schemes. In Proceedings of CRYPTO 84 on Advances in cryptology, pages 47–53, New York, NY, USA, 1984. Springer-Verlag New York, Inc.

[173] Masaaki Shirase, Tsuyoshi Takagi, and Eiji Okamoto. An efficient countermeasure against side channel attacks for pairing computation. In Liqun Chen, Yi Mu, and Willy Susilo, editors, Information Security Practice and Experience, volume 4991 of Lecture Notes in Computer Science, pages 290–303. Springer Berlin Heidelberg, 2008.

[174] Victor Shoup. NTL : a library for doing number theory, 2009. `http://www.shoup.net/ntl/`.

[175] J.H. Silvermann. The Arithmetic of elliptic curves, volume 106 of graduate texts in Mathematics. Springer-Verlag, 1986.

[176] M Smache, N El Mrabet, A Tria, JJ GIL-QUIJANO, E Riou, and G Chaput. Modeling a node capture attack in a secured wireless sensor network. In IEEE 3rd World Forum on Internet of Things, 2016.

[177] Martijn Stam and Arjen K. Lenstra. Efficient subgroup exponentiation in quadratic and sixth degree extensions. In Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers, pages 318–332, 2002.

[178] Alexandre F. Tenca and Çetin Kaya Koç. A scalable architecture for montgomery multiplication. In Çetin Kaya Koç and Christof Paar, editors, Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings, volume 1717 of Lecture Notes in Computer Science, pages 94–108. Springer, 1999.

[179] Elena Trichina and Roman Korkikyan. Multi fault laser attacks on protected crt-rsa. In Fault Diagnosis and Tolerance in Cryptography (FDTC), 2010 Workshop on, pages 75–86. IEEE, 2010.

[180] Thomas Unterluggauer and Erich Wenger. Efficient pairings and ECC for embedded systems. In Lejla Batina and Matthew Robshaw, editors, Cryptographic Hardware and Embedded Systems - CHES 2014, volume 8731 of Lecture Notes in Computer Science, pages 298–315. Springer Berlin Heidelberg, 2014.

[181] Thomas Unterluggauer and Erich Wenger. Practical attack on bilinear pairings to disclose the secrets of embedded devices. In Ninth International Conference on Availability, Reliability and Security, ARES 2014, Fribourg, Switzerland, September 8-12, 2014, pages 69–77. IEEE Computer Society, 2014.

[182] Marten van Dijk, Robert Granger, Dan Page, Karl Rubin, Alice Silverberg, Martijn Stam, and David Woodruff. Practical cryptography in high dimensional tori. In Ronald Cramer, editor, Advances in Cryptology – EUROCRYPT 2005 : 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005. Proceedings, pages 234–250, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[183] Frederik Vercauteren. Optimal pairings. IEEE Trans. Inf. Theor., 56(1) :455–461, January 2010.

[184] Mahendra Vucha and Arvind Rajawat. Design and fpga implementation of systolic array architecture for matrix multiplication. International Journal of Computer Applications, 26(3) :0975 Ð 8887, july 2011.

[185] H. Wang, K. Wang, L. Zhang, and Bao. Li. Pairing computation on elliptic curves of jacobi quartic form. Chinese Journal of electronics, vol. 20(4) pp. 655-661, 2011.

[186] L.C. Washington. Elliptic Curves, Number Theory and Cryptography. Discrete Math .Aplli, Chapman and Hall, 2008.

[187] A. Weil. Sur les fonctions algebriques a corps de constantes fini. In Compte rendu academie des sciences Paris 210, pages 592–594, 1940.

[188] Jiang Weng, Yunqi Dou, Chuangui Ma, and Nadia El Mrabet. Fault attacks against the miller algorithm in hessian coordinates. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers, volume 7537 of Lecture Notes in Computer Science, pages 102–112. Springer, 2011.

[189] Claire Whelan and Michael Scott. The Importance of the Final Exponentiation in Pairings when considering Fault Attacks. In Pairing-Based Cryptographyâ Pairing 2007, volume 4575 of LNCS, pages 225–246. Springer, 2007.

[190] Claire Whelan and Mike Scott. Side channel analysis of practical pairing implementations : Which path is more secure ? In PhongQ. Nguyen, editor, Progress in Cryptology - VIETCRYPT 2006, volume 4341 of Lecture Notes in Computer Science, pages 99–114. Springer Berlin Heidelberg, 2006.

[191] Bo Yang, Kaijie Wu, and Ramesh Karri. Scan based side channel attack on dedicated hardware implementation of data encryption standard. In Test Conference 2004, proceedings ITC 2004, pages 339 – 344, 2004.

[192] Loïc Zussa, Jean-Max Dutertre, Jessy Clédière, and Assia Tria. Power supply glitch induced faults on FPGA : an in-depth analysis of the injection mechanism. In 2013 IEEE 19th International On-Line Testing Symposium (IOLTS), Chania, Crete, Greece, July 8-10, 2013, pages 110–115. IEEE, 2013.