Project Proposal

Title: Cryptologie, Arithmétique : Matériel et Logiciel
(Cryptology, Arithmetic: Hardware and Software)

Acronym: CARAMEL

Scientific leader: Pierrick Gaudry

Proposed INRIA theme:
Algorithmics, Programming, Software and Architecture
Algorithms, Certification, and Cryptography

INRIA scientific and technological challenges:
Guaranteeing the reliability and security of software-prevalent systems

Keywords: integer factorization, discrete logarithm, (hyper-)elliptic curve cryptography, computer arithmetic

Contents

1 Team

- **Head of project-team**
  - Pierrick Gaudry (CR1 CNRS)
- **Vice-head of project-team**
  - Paul Zimmermann (DR1 INRIA)
- **Administrative assistant**
  - Emmanuelle Deschamps
- **Staff members**
  - Jérémie Detrey (CR2 INRIA)
  - Emmanuel Thomé (CR1 INRIA)
- **Postdoctoral fellow**
  - Antonio Vera (CNRS, until January 2010)
  - Sylvain Chevillard (INRIA, until December 2010)
- **Technical staff**
  - Lionel Muller (INRIA IJD, starting November 2009)
- **Ph.D. students**
  - Alexander Kruppa (CNRS, defense planned for December 2009 or January 2010)
  - Damien Robert (UHP, defense planned for the end of 2010)
  - Gaëtan Bisson (INPL, defense planned for the end of 2011)
  - Romain Cosset (CNRS-DGA, defense planned for the end of 2011)
  - Nicolas Estibals (UHP, defense planned for the end of 2012)
- **Associate members**
  - Richard Brent (ANU, Canberra, Australia)
  - Marion Videau (UHP, on leave at ANSSI until January 2011)

2 Overall Objectives

A general keyword that could encompass most of our research objectives is *arithmetic*. Indeed, in the CARAMEL proposal, the goal is to push forward the possibilities to compute efficiently with objects having an arithmetic nature. This includes integers, real and complex numbers, polynomials, finite fields, and, last but not least, algebraic curves.

Our main application domains are public-key cryptography and computer algebra systems. Concerning cryptography, we concentrate on the study of the primitives based on the factorization problem or on the discrete-logarithm problem in finite fields or (Jacobians of) algebraic curves. Both the constructive and destructive sides are of interest to this proposal. For applications in computer algebra systems, we are mostly interested in arithmetic building blocks for integers, floating-point numbers, polynomials, and finite fields. Also some higher level functionalities like factoring and discrete-logarithm computation are usually desired in computer algebra systems.

Since we develop our expertise at various levels, from most low-level software or hardware implementation of basic building blocks to complicated high-level algorithms like integer factorization

or point counting, we have remarked that it is often too simple-minded to separate them: we believe that the interactions between low-level and high-level algorithms are of utmost importance for arithmetic applications, yielding important improvements that would not be possible with a vision restricted to low- or high-level algorithms.

We emphasize three main directions for our project:

- Integer factorization and discrete-logarithm computation in finite fields.

  We are in particular interested in the number field sieve algorithm (NFS) that is the best known algorithm for factoring large RSA-like integers, and for solving discrete logarithms in prime finite fields. A sibling algorithm, the function field sieve (FFS) is the best known algorithm for computing discrete logarithms in finite fields of small characteristic.

  In all these cases, we plan to improve on existing algorithms, with a view towards practical considerations and setting new records.

- Algebraic curves and cryptography.

  Our two main research interests on this topic lie in genus 2 cryptography and in the arithmetic of pairings, mostly on the constructive side in both cases. For genus 2 curves, a key algorithmic tool that we plan to develop is the computation of explicit isogenies; this will allow improvements for cryptography-related computations such as point counting in large characteristic, complex-multiplication construction and computation of the ring of endomorphisms.

  For pairings, our principal concern is the optimization of pairing computations, in particular in hardware, or in constrained environments. We will develop automatic tools to help in choosing the most suitable (hyper-)elliptic curve and generating efficient hardware for a given security level and set of constraints.

- Arithmetic.

  Integer, finite-field and polynomial arithmetics are ubiquitous to our research. We consider them not only as tools for other algorithms, but as a research theme *per se*. We are interested in algorithmic advances, in particular for large input sizes where asymptotically fast algorithms become of practical interest. We also keep an important implementation activity, both in hardware and in software.

This project proposal follows the CACAO project-team that ends in December 2009, as a consequence of his leader, Guillaume Hanrot, moving to Lyon.

3 Scientific Foundations

One of the main applications for our project is public-key cryptography. After 20 years of hegemony, the classical public-key algorithms (whose security is based on integer factorization or discrete logarithm in finite fields) are currently being overtaken by elliptic curves. The fundamental reason for this is that the best-known algorithms for factoring integers or for computing discrete logarithms in finite fields have a subexponential complexity, whereas the best known attack for elliptic-curve discrete logarithms has exponential complexity. As a consequence, for a given security level 2^n, the key sizes must grow linearly with n for elliptic curves, whereas they grow like n^3 for RSA-like systems. As a consequence, several governmental agencies, like the NSA or the BSI, now recommend to use elliptic-curve cryptosystems for new products that are not bound to RSA for backward compatibility.

Besides RSA and elliptic curves, there are several alternatives currently under study. There is a recent trend to promote alternate solutions that do not rely on number theory, with the objective of building systems that would resist a quantum computer (in contrast, integer factorization and discrete logarithms in finite fields and elliptic curves have a polynomial-time quantum solution). Among them, we find systems based on hard problems in lattices (NTRU is the most famous),

those based on coding theory (McEliece system and improved versions), and those based on the difficulty to solve multivariate polynomial equations (HFE, for instance). None of them has yet reached the same level of popularity as RSA or elliptic curves for various reasons, including the presence of unsatisfactory features (like a huge public key), or the non-maturity (system still alternating between being fixed one day and broken the next).

Returning to number theory, an alternative to RSA and elliptic curves is to use other curves and in particular genus 2 curves. These so-called hyperelliptic cryptosystems have been proposed[1] in 1989, soon after the elliptic ones, but their deployment is by far more difficult. The first problem was the group law. For elliptic curves, the elements of the group are just the points of the curve. In a hyperelliptic cryptosystem, the elements of the group are points on a 2-dimensional variety associated to the genus 2 curve, called the Jacobian variety. Although there exist polynomial-time methods to represent and compute with them, it took some time before getting a group law that could compete with the elliptic one in terms of speed. Another question that is still not yet fully answered is the computation of the group order, which is important for assessing the security of the associated cryptosystem. This amounts to counting the points of the curve that are defined over the base field or over an extension, and therefore this general question is called point-counting. In the past ten years there have been major improvements on the topic, but there are still cases where we do not have any solution.

Another recent discovery in public-key cryptography is the fact that having an efficient bilinear map that is hard to invert (in a sense that can be made precise) can lead to powerful cryptographic primitives. The only examples we know of such bilinear maps are associated with algebraic curves, and in particular elliptic curves: this is the so-called Weil pairing (or its variant, the Tate pairing). Initially considered as a threat for elliptic-curve cryptography, they have proven to be quite useful from a constructive point of view, and since the beginning of the decade, hundreds of articles have been published, proposing efficient protocols based on pairings. A long-lasting open question, namely the construction of a practical identity-based encryption scheme, has been solved this way. The first standardization of pairing-based cryptography has recently occurred (see ISO/IEC 14888-3 or IEEE P1363.3), and a large deployment is to be expected in the next years.

Despite the raise of elliptic curve cryptography and the variety of more or less mature other alternatives, classical systems (based on factoring or discrete logarithm in finite fields) are still going to be widely used in the next decade, at least, due to resilience: it takes a long time to adopt new standards, and then an even longer time to renew all the software and hardware that is widely deployed.

This context of public-key cryptography motivates us to work on integer factorization, for which we have acquired expertise, both in factoring moderate-sized numbers, using the ECM (Elliptic Curve Method) algorithm, and in factoring large RSA-like numbers, using the number field sieve algorithm. The goal is to follow the transition from RSA to other systems and continuously assess its security to adjust key sizes. We also want to work on the discrete-logarithm problem in finite fields. This second task is not only necessary for assessing the security of classical public-key algorithms, but is also crucial for the security of pairing-based cryptography.

We also plan to investigate and promote the use of pairing-based and genus 2 cryptosystems. For pairings, this is mostly a question of how efficient can such a system be in software, in hardware, and using all the tools from fast implementation to the search for adequate curves. For genus 2, as said earlier, constructing an efficient cryptosystem requires some more fundamental questions to be solved, namely the point-counting problem.

We summarize in the following table the aspects of public-key cryptography that we propose to address.

---

[1] Koblitz, N. Hyperelliptic cryptosystems. *J. Cryptology 1* (1989), 139–150.

public-key primitive	cryptanalysis	design	implementation
RSA	X	–	–
Finite Field DLog	X	–	–
Elliptic Curve DLog	–	–	Soft
Genus 2 DLog	–	X	Soft
Pairings	X	X	Soft/Hard

Another general application for the project is computer algebra systems (CAS), that rely in many places on efficient arithmetic. Nowadays, the objective of a CAS is not only to have more and more features that the user might wish, but also to compute the results fast enough, since in many cases, the CAS are used interactively, and a human is waiting for the computation to complete. To tackle this question, more and more CAS use external libraries, that have been written with speed and reliability as first concern. For instance, most of today's CAS use the GMP library for their computations with big integers. Many of them will also use some external Basic Linear Algebra Subprograms (BLAS) implementation for their needs in numerical linear algebra.

During a typical CAS session, the libraries are called with objects whose sizes vary a lot; therefore being fast on all sizes is important. This encompasses small-sized data, like elements of the finite fields used in cryptographic applications, and larger structures, for which asymptotically fast algorithms are to be used. For instance, the user might want to study an elliptic curve over the rationals, and as a consequence, check its behaviour when reduced modulo many small primes; and then he can search for large torsion points over an extension field, which will involve computing with high-degree polynomials with large integer coefficients.

Writing efficient software for arithmetic as it is used typically in CAS requires the knowledge of many algorithms with their range of applicability, good programming skills in order to spend time only where it should be spent, and finally good knowledge of the target hardware. Indeed, it makes little sense to disregard the specifics of the possible hardware platforms intended, even more so since in the past years, we have seen a paradigm shift in terms of available hardware: so far, it used to be reasonable to consider that an end-user running a CAS would have access to a single-CPU processor. Nowadays, even a basic laptop computer has a multi-core processor and a powerful graphics card, and a workstation with a reconfigurable coprocessor is no longer science-fiction.

In this context, one of our goals will be to investigate and take advantage of these influences and interactions between various available computing resources in order to design better algorithms for basic arithmetic objects. Of course, this is not disconnected from the others goals, since they all rely more or less on integer or polynomial arithmetic.

4 NFS-like algorithms for factoring and discrete logarithm

4.1 State of the art

When considering hard problems on which to base public-key cryptosystems, it is traditional to refer to records, namely the largest key size that has been broken by solving the instance of the hard problem. For the RSA cryptosystems, some challenges have been published. They consist in a series of large integers constructed as products of two large primes. The largest solved challenge[2] is RSA-200, which is an integer with 663 bits. The RSA-768 challenge is currently being attacked by several teams, including CARAMEL; it should be finished during the timespan of CARAMEL, and should take around 3000 CPU-years. The algorithm used for these record factorization computations is the number field sieve, for which several implementations exist: several record-sized computations have been performed until 2000 at CWI using the code developed at the time by Herman te Riele, Peter Montgomery and Arjen Lenstra; more recent records have been

[2] Bahr, F., Boehm, M., Franke, J., and Kleinjung, T. Factorization of `RSA200`. Unpublished electronic mail. Details at `http://www.rsa.com/rsalabs/node.asp?id=2879`.

completed using software developed by Jens Franke and Thorsten Kleinjung, and the current computation uses an updated version of this program. One should note that the two code bases mentioned above are not widely accessible. Free software implementing the number field sieve include Jason Papadopoulos' msieve program, as well as the CADO-NFS software suite developed by CARAMEL.

The general outline of the number field sieve distinguishes several steps. For ease of exposition, we mention the important steps which are relevant areas for current work and possible optimizations.

- Polynomial selection. The "number field" in the number field sieve is defined by a polynomial which can be chosen with some flexibility. Given the input integer which is to be factored, the task of the polynomial selection step is to identify an exceptionally "good" polynomial, yielding an exceptionally high throughput of relations in the relation collection step.
- Sieving (relation collection). The lattice sieving algorithm calls for careful optimization of the behaviour with respect to the CPU cache.
- Cofactorization. The output of the sieving yields in particular a large number of "cofactors" which must be split into several factors in order to identify smooth relations. This has to be fast, and calls for careful factorization algorithms specially crafted for the typical size of cofactors encountered.
- Linear algebra. In the number field sieve, a very large linear system must be solved. The size of the linear systems under consideration (more than 100 gigabytes for the matrix to be solved for the upcoming RSA-768 factorization) calls for distribution across a network.

For the discrete-logarithm problem in finite fields, the situation differs depending on the characteristic of the base field. If it is large, and even if the field is a prime field, the best known algorithm is a variant of the number field sieve, and the record computation[3] is for a field of size 530 bits. For small-characteristic base fields, the best known algorithm is the function field sieve. The record computation in characteristic 2 is of 613 bits[4] and in characteristic 3 it is of 352 bits[5]. The medium size extension degree is also interesting for pairing applications. In that case, a milestone has been settled with a record computation[6] in GF(370801^{30}). No software specifically intended for the discrete-logarithm problem is distributed. Yet the underlying structure of the number field sieve implementations can largely be reused, and for large prime fields the sieving code can be reused almost without modification. However the major difference between integer factorization and discrete-logarithm computations is that the linear system to be solved is defined over a prime field much larger than GF(2).

Recent research on the discrete-logarithm problem in finite fields resulted in improvements for the medium prime and small prime cases. Some proof-of-concept experiments have been performed for the function field sieve, but have not been pushed as far as for NFS.

4.2 Detailed research objectives

Permanent team members involved: Gaudry, Thomé, Zimmermann.

Non-permanent members involved: Kruppa, Vera, Muller.

---

[3] Kleinjung, T. Discrete logarithms in GF(p) – 160 digits, Feb. 2007. Email to the `NMBRTHRY` mailing list. Available at `http://listserv.nodak.edu/archives/nmbrthry.html`.

[4] Joux, A., and Lercier, R. Discrete logarithms in GF(2^{607}) and GF(2^{613}), Sept. 2005. Email to the `NMBRTHRY` mailing list. Available at `http://listserv.nodak.edu/archives/nmbrthry.html`.

[5] Granger, R., Holt, A. J., Page, D., Smart, N. P., and Vercauteren, F. Function field sieve in characteristic three. In *ANTS-VI* (2004), D. Buell, Ed., vol. 3076 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 223–234. 6th Algorithmic Number Theory Symposium, Burlington, VT, June 2004.

[6] Joux, A., and Lercier, R. Discrete logarithms in GF(370801^{30}) – 168 digits – 556 bits, Nov. 2005. Email to the `NMBRTHRY` mailing list. Available at `http://listserv.nodak.edu/archives/nmbrthry.html`.

4.2.1 Short-term objectives

The number field sieve algorithm (NFS) for integer factorization is an important topic for the members of CARAMEL, following the CADO ANR project (Nov. 2006 – Jan. 2010) where software has been developed implementing most of the best known strategies for this algorithm.

During CARAMEL, we will continue to work on NFS for integer factorization. This includes the continuation of the maintenance and development of the software, which is our main platform for testing new ideas for improving the algorithm. We plan to work on the following main steps of the algorithm, and seek further improvements.

- Polynomial selection, following the latest ideas of Kleinjung. Current work in collaboration with P. Montgomery from Microsoft Research focuses on polynomial selection with non-linear polynomials (current algorithms for polynomial selection yield a linear polynomial and an algebraic polynomial of degree typically 5 or 6).
- Relation collection, in particular at the interface between the sieving and the large prime separation steps, where ECM can be used.
- Linear algebra, with a view towards using the best of the computing resources (from the processor level to the grid scale).
- Automatic choice of sensible parameters for the algorithm.

Our implementation of NFS is targeted at both medium-sized integers and record-sized RSA-moduli. The former is important for the computer algebra application, and we wish to keep an eye to the possibility of having our software included in some widely used computer algebra system[1]. The latter is of course important for cryptography. We are currently involved, together with EPFL, CWI, NTT, in the record factorization of RSA-768, that should be finished in early 2010. Cooperation with EPFL and CWI being fruitful, we plan to continue large-scale experiments thereafter, depending on our access to sufficient computing resources.

4.2.2 Long-term objectives

Based on our expertise on NFS for factoring, we will work on the function field sieve algorithm (FFS) for discrete logarithm in finite fields. The algorithm is very similar to NFS, so that many of the improvements that have been developed in the past few years for NFS can be adapted to FFS, but still too different for any of those improvements to have been ported yet. To our knowledge, the polynomial selection algorithms and the lattice sieving *à la* Franke/Kleinjung have not been tried in the FFS context yet. This will not be immediate, and it is not clear how these improvements will behave in practice, but we are confident that we will be able to set new records for discrete logarithms in finite fields of characteristic 2 and 3. For instance, it sounds reasonable to reach the kilobit milestone for discrete logarithms in the next few years. Computationally speaking, the major stumbling block for this task will be the extension of the linear algebra computation to a prime field. The current implementation in the CADO-NFS software suite is extensible to this case. Given the leap in computational difficulty for the linear system solving, it is expected that more effort would have to be put into sieving.

For both NFS and FFS, we will provide a software implementation, but we will also work on hardware aspects. Our main objective is to evaluate the speed-up that can be obtained using an FPGA-based platform in the various steps of the algorithm. This can also be important to evaluate the security of a cryptosystem against an attacker that would build a dedicated ASIC solution.

On the factoring side, we plan to study Coppersmith's modifications of the number field sieve algorithm, which provide the best asymptotic complexity. The value of the cut-off point with the standard number field sieve is far from clear, and it is likely that a careful study of Coppersmith's variations will reveal both advantages and drawbacks of the approach.

[1] The maintainers of the SAGE computer algebra system expressed interest in this perspective.

4.2.3 Milestones

- Solving the RSA-768 challenge.
- Setting new records for discrete-logarithm computation in finite fields of characteristic 2 and 3.
- Polynomial selection with non-linear polynomials for NFS.

5 Algebraic curves and cryptography

5.1 State of the art

Elliptic and hyperelliptic curves alike are deemed suitable for cryptography only if they yield a large subgroup of prime order (generally we require the curve order itself to be prime or almost prime). This property is usually assessed via point counting algorithms which, in the case of genus 2, fall into two categories. If the base field of the curve has small characteristic, we have efficient and practical point counting algorithms at hand. However, in the case of prime fields, the problem is more difficult. The current record for point counting is held by CARAMEL, and reaches the currently recommended 128-bit security level.

An alternative to point counting is to use a Complex-Multiplication (CM) construction, but this makes it difficult to force small coefficients in the curve equation (this slows down the corresponding cryptosystem), and there exist some researchers and agencies that fear a security threat coming from the additional CM structure. On the other hand, CM constructions allow the construction of cryptosystems with specific features, like having an explicit and efficient pairing. For the moment, the CM method in genus 2 is not as advanced as for elliptic curves; the theory[7] is done, but very high class numbers have not yet been reached[8].

As far as pairings are concerned, there are now many software implementation reports based on elliptic curves and a few for genus 2 curves. Most of them are for a security level of 128 bits or less. In hardware, there are a few results in the case of elliptic curves, but getting a security level of 128 bits on an FPGA is still problematic. When a higher security level is sought, it becomes necessary to use non-supersingular curves, and the search for suitable (so-called *pairing-friendly*) curves has been an active area of research in recent years, and is still not close.

5.2 Detailed research objectives

Permanent team members involved: Detrey, Gaudry, Thomé.

Non-permanent members involved: Bisson, Cosset, Robert, Estibals.

Our objectives in the context of algebraic curves and cryptography are related to genus 2 cryptosystems and pairing-based cryptography.

5.2.1 Short-term objectives

In the context of genus 2 cryptography, our first objective will be to build a curve from its ring of endomorphisms. This enables the construction of curves with additional features, like the presence of an efficient pairing, and, furthermore, the number of points can be easily deduced from the ring of endomorphisms. We plan to work on the Complex-Multiplication method (CM) for genus 2 curves, using both a complex analytic approach and a method based on the Chinese Remainder Theorem.

---

[7] Streng, M. Computing Igusa class polynomials, 2009. `http://arxiv.org/abs/0903.4766`.

[8] Weng, A. Constructing hyperelliptic curves of genus 2 suitable for cryptography. *Math. Comp. 72* (2003), 435–458.

The second approach can take advantage of explicit isogenies, which is subject to active research and is one of the main topics of the ANR CHIC project (Sept. 2009 – Aug. 2012), that should provide basic algorithmic solutions in a rather short term.

Another short-term objective is to improve the group law in genus 2. The recent introduction of Edwards coordinates for elliptic curves has provided a nice speed-up, and we will investigate how such coordinates can be adapted to genus 2. This has been tried by several people already, but an interpretation in terms of Theta functions can help revealing a hidden structure that would extend to genus 2.

In the context of pairing-based cryptography, our main objective is to explore the cost vs. speed trade-off for a given security level (e.g. equivalent to AES-128). "Cost" here may be referring to the CPU requirements (micro-architecture, instruction set, number of cores, etc.) in the case of software implementations, or the silicon area or power consumption for hardware implementations. For lower security levels, the best achievable speed seems to come from supersingular elliptic curves in characteristic 2 or 3, but since the so-called embedding degree of such curves is limited, they do not scale so well to higher security levels. We therefore plan to investigate other alternatives such as ordinary curves over prime fields (*i.e.*, pairing-friendly curves) or hyperelliptic curves.

Our short- to medium-term objectives will concentrate on the FPGA target: they have proven to be a low-cost solution for reaching high speeds compared to general-purpose CPUs, and they are also nice prototyping platforms for a potential ASIC implementation. Our approach is to write automatic tools to help the generation of VHDL, thus allowing us to test a large amount of architectural variants for a given set of parameters, but also to let these parameters vary, in order to tune the circuit to a given target security.

5.2.2 Long-term objectives

Our long-term target is to have a fast point-counting algorithm for genus 2 curves over prime fields, in order to facilitate the construction of genus 2 cryptosystems based on random curves, that reach the highest security levels that can be found in standards. The current best known algorithm is a variant of Schoof's algorithm, and it is highly desirable to have an analogue of the Elkies-Atkin improvement for genus 2. A key to this project resides in having explicit isogenies in genus 2, which is an objective per se that we develop in the following.

Explicit isogenies have been the topic of recent progresses. In CARAMEL, our approach is based on the algebraic theory of Theta functions. We will continue in this direction, trying to push it as far as we can. At the same time, we will study the approaches (analytic, geometric) taken by other groups to see how they compare, and most importantly how we can combine them to get the best of both worlds. On the long term, the existence of a polynomial-time algorithm for explicit isogenies will be turned into a practical and efficient solution that can be used as a building block in various applications, including point counting, CM construction and computation of the ring of endomorphisms.

Regarding pairings, following our research and experiments on various hardware solutions for pairing-based cryptography, we will integrate all the tools we have developed in a general software suite dedicated to the conception of hardware operators for pairings, from the choice of the (hyper)elliptic curve that is the most suitable for the target security to the effective implementation of the circuit (given by its VHDL representation, for instance). The software will also include exploration heuristics for various algorithmic, arithmetic and architectural alternatives, selecting the best one according to some used-specified criteria.

Another long-term plan is to study explicit formulae for pairing computation in a general setting. In particular, we plan to look for formulae based on Theta functions, in relation to our research on fast algorithms for the group law.

5.2.3 Milestones

- Setting new records in genus 2 CM construction.

- Setting new records in genus 2 point counting over prime fields.
- Proposing an efficient and flexible hardware implementation of pairing-based cryptographic primitives.

6 Arithmetic

6.1 State of the art

In the context of integer arithmetic, the reference software is the GMP library. It is highly optimized for common platforms. Although it is thread-safe, it can not itself use several threads to perform a single large arithmetic operation. It is also not yet ported to exotic architectures like graphics cards. In the subquadratic range (*i.e.*, for large and very large operands), GMP is not always optimal[2] and some improvements are currently under development. On the theoretical side, a recent major result is the algorithm by Fürer[9] that improves the asymptotic complexity of integer multiplication. It has not yet proven to be of practical interest.

For arbitrary-precision floating-point arithmetic, the MPFR library has now become a reference. With its companion libraries MPC (for complex floating-point numbers) and MPFI (for interval arithmetic) it provides a complete "toolbox" for arbitrary-precision computation.

Polynomial arithmetic, and especially when the coefficients are integers or elements of a finite field, is strongly related to integer arithmetic, since the so-called Kronecker substitution allows to reduce polynomial multiplication to integer multiplication. In the opposite direction, many FFT-based integer multiplication algorithms can be interpreted in terms of polynomial arithmetic. There has been some recent progress by Harvey[10] on this interaction, and this is still an active topic.

6.2 Detailed research objectives

Permanent team members involved: Detrey, Gaudry, Thomé, Zimmermann.

Non-permanent members involved: Chevillard, Estibals, Kruppa.

As said earlier, this theme is rather transversal, since integer, polynomial, and finite-field arithmetic are ubiquitous to most of our activities. It has of course a large implementation component, but also includes some research on algorithmic advances. Therefore, we do not separate into short- and long-term objectives.

6.2.1 Algorithmic advances

An area where we expect algorithmic advances is in asymptotically fast multiplication algorithms based on the Fast Fourier Transform. We propose to continue our work on the Schönhage-Strassen algorithm for multiplying integers—in particular it would be interesting to try to optimize that algorithm for a fixed transform length—and to investigate how ideas taken from Fürer's new algorithm can be made practical. For the moment, it is not clear what could be gained by using floating-point arithmetic (and a complex-number Fourier transform) for small transforms. Indeed, in order to guarantee the result after rounding, only a part of the available bits of the mantissa can be used. On the other hand, a hybrid version is something that is worth trying.

Another direction where algorithms need to be revisited comes from the variety of computing resources that have appeared in the last few years. It is now clear that SIMD instructions that are present on the processors (SSE2 for Intel) can be useful for arithmetic. This should continue

[2] For example, the computation of the Jacobi symbol of two n-bit numbers is quadratic in n.

[9] Fürer, M. Faster integer multiplication. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC), San Diego, California, USA* (2007), D. S. Johnson and U. Feige, Eds., ACM, pp. 57–66.

[10] Harvey, D. Faster polynomial multiplication via multipoint Kronecker substitution. *Journal of Symbolic Computation* (to appear). `http://arxiv.org/abs/0712.4046`.

to be the case, and even become more and more important. In the same spirit, we plan to have a look at the trendy GPU computing topic in the context of arithmetic, where not so much has been done for the moment. Algorithms adapted for such contexts need not only be optimized, they need quite often to be broadly redesigned in order to achieve better performance.

6.2.2 Software and hardware implementation

In terms of implementation, a first task of CARAMEL will be to continue the development and maintenance of libraries for basic arithmetic. The most important ones are MPFR and MPC that are now contributing a lot to the visibility of the team, since, for instance, MPFR is now required for compiling GCC. MPC is now reaching the same level, in particular with the implementation of all functions required by the ISO C99 standard, and has recently become an optional package for the Sage computer algebra system and its integration to GCC is under study. These floating-point libraries will therefore remain an important part of our software activities.

On the finite-field side, we plan to continue the development of the MPFQ library, that is designed to take full advantage of the knowledge of the base field at compile time. We plan to extend functionalities, so that more finite-field types are available, and to provide polynomials and matrices over finite fields.

Besides the above-mentioned libraries, we will contribute software for basic arithmetic functions in externally maintained libraries. In particular, for asymptotically fast integer multiplication, we might contribute to the GMP or MPIR[3] library, and for characteristic 2 polynomial arithmetic we will contribute to NTL via the GF2X package. For the latter package, an important change during that project will be the release of the new Intel chips with the PCLMULQDQ hardware instruction, which will perform a multiplication of two polynomials of degree less than 64 in a few cycles only, against about 50 cycles with the best current software implementation [11].

On the hardware side, we will develop and distribute a library for automatically generating VHDL code for finite field implementation on FPGA platforms.

7 Applications and Technological Transfer

7.1 Cryptology

The first application domain for our research is cryptology. This includes cryptography (constructive side) and cryptanalysis (breaking systems). For the cryptanalysis part, although it has practical implications, we do not expect any transfer in the classical sense of the term: it is more directed to governmental agencies and the end-users who build their trust, based on the cryptanalysis effort.

7.1.1 Cryptography

Our cryptographic contributions are related to multiple facets of the large realm of curve-based cryptology. While it is quite clear that a satisfying range of algorithms exist in order to provide cryptographers with elliptic curves having a suitably hard discrete logarithm (as found in cryptographic standards for instance), one must bear in mind that refinements of the requirements and extensions to curves of higher genus raise several interesting problems. Our work contributes to expanding the cryptographer's capabilities in these areas.

In the context of genus 2 curves, our work aims at two goals. First, improvements on the group law on selected curves yields better speed for the associated cryptosystems. The cryptographic primitives, and then the whole suite of cryptographic protocols built upon such curves would be accelerated. The second goal is the expansion of the set of curves that can be built given a set of desired properties. Using point counting algorithms for arbitrary curves, a curve offering a 128-bit security level, together with nice properties for fast arithmetic, has been computed by CARAMEL,

[3]MPIR is a new library which was forked from GMP 4.2.1 in 2008, and has a different license.

and curves of larger security levels will follow. Another natural target for construction of curves for cryptography is also the suitability of curves for pairings. We expect to be able to compute such curves.

Implementations of curve-based cryptography, both in hardware and software, are a necessary step on the way to assessing cryptographic speed. We plan to provide such implementations. In particular, on the hardware side, one of our goals is the design of a complete cryptographic coprocessor, including all the primitives for curve-based and pairing-based cryptography, providing optimized and configurable efficiency vs area trade-off.

Those two aspects: new systems and better implementations can be of some interest for practical applications, especially in embedded systems, but we did not find yet an industrial partner.

7.1.2 Cryptanalysis

Our research on cryptanalysis is important for the cryptographic industry: by detecting weak instances, and setting new records we contribute to the definition of recommended families of systems together with their key sizes. The user's confidence in a cryptographic primitive is also related to how well the underlying problem is studied by researchers.

In particular, our involvement in computations with "NFS-like" algorithms encompasses of course the task of assessing the computational limits for integer factorization and discrete-logarithm computations. The impact of the former is quite clear as it concerns the RSA algorithm; record-sized computations attract broad interest and determine updates on key-length recommendations. The latter are particularly important for pairing-based cryptography, since, in this context, one naturally encounters discrete-logarithm problems in extension fields of large degree.

7.2 Standardization

7.2.1 Floating-point arithmetic

The IEEE 754 standard was revised in 2008. The main new features are some new formats for decimal computations, and the recommendation of correctly rounded transcendental functions. The new decimal formats should not have an impact on our work, since we either use integer-only arithmetic, or arbitrary-precision binary floating-point arithmetic through the MPFR library.

A new standard (P1788) is currently under construction for interval arithmetic. We are not officially involved in this standard, but we follow the discussions, to check in particular that the proposed standard will also cover arbitrary precision (interval) arithmetic.

7.2.2 Curve-based cryptography

Elliptic-curve cryptography has been standardized for almost 10 years now, in the IEEE P1313 standard. This standard provides key agreement, signature and encryption schemes, based on integer factorization, discrete logarithm in finite fields and discrete logarithm in elliptic curves. There is another standardization effort, called SECG, which is mostly lead by the Certicom company, with the goal to maintain interoperability between different implementations. In particular, the SECG documents give explicit elliptic curves that can be used for cryptography. Similarly, some elliptic curves have been standardized by the US government; the latest version comes from the NSA Suite B that includes only elliptic curves defined over prime fields.

We are not involved in these standardization processes. However, there is a current revision process of the IEEE P1363 standard, and we might contribute, in particular on the section relating to the difficulty of solving underlying hard problems.

In the long term, this standard is a natural place to promote genus 2 curve cryptography, and by the time we consider that they are mature enough, we will look for an industrial partner to help us pushing towards their standardization.

7.2.3 Pairing-based cryptography

Despite their very recent discovery, identity-based cryptosystems—and more generally pairing-based cryptosystems—have already spawned several international standardization efforts.

The first standard, part of ISO/IEC 14888-3, was published in 2006. However, it almost exclusively focuses on protocols and therefore is of little interest to us. On the other hand, the IEEE P1363.3 standard, which is still in preparation, is planned to offer more details as to the considered curves and pairings on which the protocols are based.

Although we are not officially involved in the elaboration of this standard, we have already participated in the review process of its first draft, and are now considering getting more directly involved in this project.

7.3 Computer algebra systems

Some of our software libraries are being used by computer algebra systems. Most of those libraries are free software, with a license that allows proprietary systems to link them. This gives us a maximal visibility, with a large number of users.

7.3.1 Magma

Magma is a very large computational algebra package. It provides a mathematically rigorous environment for computing with algebraic, number-theoretic, combinatoric, and geometric objects. It is developed in Sydney, by the team around John Cannon. It is non-commercial (in the sense that its goal is not to make profit), but is not freely distributed and is not open-source.

Several members of the team have visited Sydney to contribute to the development of Magma, by implementing their algorithms or helping in integrating their software. Our link to Magma exists also via the libraries it uses: it currently links MPFR and MPC for its floating-point calculations, and links GMP-ECM as part of its factorization suite.

7.3.2 Pari-gp

Pari/GP is a computational number theory system that is composed of a C library and an interpreter on top of it. It is developed in Bordeaux, where Karim Belabas from the LFANT project-team is the main maintainer. Its license is GPL. Although we do not directly contribute to this package, we have good contact with the developers and in the future, MPFR and MPC could be included.

7.3.3 Sage

Sage is a fairly large scale, recent computer algebra system written in python. Sage aggregates a large amount of existing free software, aiming at the goal of selecting the fastest free software package for each given task. The motto of Sage is that instead of "reinventing the wheel" all the time, Sage is "building the car". To date, Sage links MPFR and GMP-ECM. Plans exist to link MPC, GF2X, and CADO-NFS into Sage.

8 Software and hardware developments

8.1 Software

8.1.1 MPFR / MPC

MPFR and MPC are two libraries for arbitrary precision floating-point and complex floating-point computations respectively. They now have a good visibility: MPFR is required for compiling GCC since version 4.3 of GCC, and the same should hold for MPC in the future. MPFR implements all

functions required by ISO C99, and so does the development version of MPC. MPFR is included in the Sage computer algebra system.

MPFR is co-developed with the Arénaire project-team (Lyon), while MPC is co-developed with the LFANT project-team (Bordeaux). Maintenance will be continued but the investment of CARAMEL in further development of new features will be limited in the next few years (except if there is a need for a new functionality for a given application); the main improvements will concern the efficiency of the implementation, either by using new functionalities provided by GMP, or by implementing faster algorithms.

8.1.2 GMP-ECM

GMP-ECM is a program for factoring integers using the Elliptic Curve Method; it has good visibility among factoring people. It is used by Magma and Sage.

The code is mature and stable, but improvements are still frequently added, coming from better choices of curves (elliptic- or genus-2-curve families), together with their group law, better integer and polynomial arithmetic, and better stage 2 algorithms. The current development version includes the HECM algorithm developed by Romain Cosset.

8.1.3 CADO-NFS

CADO-NFS is a set of programs for factoring integers using the number field sieve algorithm. The software suite is the product of the CADO ANR project, and offers a complete number field sieve implementation. Snapshots have been made available in Spring 2009, and the source code repository is now public. Several improvements will be brought to CADO-NFS, both in terms of unification of the different programs, and also performance-wise. Hopefully CADO-NFS will become the reference implementation for the number field sieve.

8.1.4 MPFQ

MPFQ is a code generation library for finite fields. It produces highly optimized code for computations that occur is cryptography, namely with moderate-size finite fields. In particular, MPFQ has been used for providing record-speed implementations of cryptosystems using curves of genus 1 and 2, both on prime and binary finite fields (some of these records have been superseded in special cases since then). MPFQ has been developed mostly for internal usage, but is freely distributed and will continue to be developed and maintained. Some computer algebra systems have expressed interest in such a library, but no concrete project has grown yet.

8.1.5 GF2X

GF2X is a software library implementing a variety of algorithms for multiplying univariate polynomials over GF(2). GF2X offers presently a very primitive API, exporting only the multiplication routine. Yet it is the fastest available code for this operation, offering low-level optimizations, mid-range algorithms such as the Karatsuba, Toom-3 and Toom-4 methods, as well as eventually an FFT algorithm. GF2X can be used as an add-on to the popular NTL library, as of NTL version 5.5.

8.1.6 Countg2

Countg2 is the name of the software that we have developed for counting points on genus 2 curves over prime fields, and that was used to set records. It is based on the NTL library and is a successor of NTLJac2 that had been distributed under a GPL licence. Countg2 has been heavily modified during a large-scale record computation in Spring 2009. It is now in an unstable state. Therefore some cleaning will be done and the code will be released thereafter. It will be a natural place to implement the new algorithms we plan to develop for genus 2 point counting.

8.2 Hardware

Currently less mature than software, we plan to fully integrate hardware development into the area of expertise of CARAMEL. This will also encourage us to explore intermediate solutions, such as many-core architectures (modern GPUs or Intel's Larrabee) and FPGA-augmented microcontrollers.

A direct application of hardware to our problematics would be the study of the relevance and practicality of *ad-hoc* hardware accelerators for NFS-like algorithms such as FFS. Indeed, being much better suited to arithmetic over fields of small characteristic than CPUs, the hardware target is a viable candidate for the implementation of the sieving phase of such algorithms. We therefore plan to develop an FPGA-based sieving processor, possibly considering several architectural alternatives in order to cover as widely as possible the area-time tradeoff spectrum. Benchmarking such accelerators against software implementations would allow us to quantify more accurately the advantage an attacker would gain from having access to such computing resources.

In a second direction, building up on the expertise acquired in the domain of hardware implementation of pairing-based cryptosystems, we plan to develop and release a suite of automatic tools able to generate full coprocessors for pairing-based cryptography, according to given security, resource and computation time constraints. These tools, backed by a complete library of pairing coprocessors, would allow any designer to automatically generate the accelerator best suited to her needs without her having to delve into the involved mathematics behind pairings.

Third, since the two previous objectives are to rely heavily upon finite-field arithmetic, we also plan to develop an extensive library of hardware operators targeted at finite fields. Both small ($\mathbb{F}_{2^m}$ and $\mathbb{F}_{3^m}$) and large characteristics are to be studied in this context. Such a library would actually provide the basic building blocks for a much wider range of applications than sieving in FFS and pairing-based cryptography. Ensuring widespread distribution and maintenance of this library would therefore be full objectives in themselves for CARAMEL.

9 Positioning within the Scientific Community

9.1 NFS-like algorithms

This topic has attracted a constant interest since the discovery of the number field sieve algorithm in the 90's. For factoring and discrete logarithms over prime fields, the most active teams in the past years have been EPFL (A. Lenstra), Bonn (J. Franke and T. Kleinjung), NTT (K. Aoki), and CWI (P. L. Montgomery and H. te Riele). Usually these people work together to set new records, and we have joined them for the still-to-come RSA-768 challenge resolution.

There are also isolated people contributing factoring software: C. Monico and J. Papadopoulos.

Since our implication in the CADO ANR project, we have affirmed our position within this factoring community by writing and publishing a complete and independent factoring software, and by organizing a workshop dedicated to integer factorization at LORIA. Our visibility on the topic will increase at the end of the ongoing RSA-768 computation, in which we are the main contributor in terms of computed relations.

For computing discrete logarithms in non-prime finite fields, there is far less competition, at least in terms of record setting: most of them have been performed by IRMAR (R. Lercier) and Leuven (F. Vercauteren). A decade ago, E. Thomé from our group also worked on the topic and set a new record, but right now, our visibility on the topic has decreased.

We are in good terms with the specialists of this topic and could collaborate with them in the future.

9.2 Algebraic curves in cryptography

On the topic of curves and pairing in cryptography, we have many competitors. In France, we have a starting collaboration via our common ANR CHIC project with IRMAR (Rennes) and

IML (Marseille). We can also cite TANC and LFANT, which are project-teams with which we collaborate (see below). Other French research groups include Toulouse (J.-M. Couveignes) and Paris 7 (J.-F. Mestre) on the mathematical side, and Grenoble (P. Elbaz-Vincent) and Caen (F. Laguillaumie) with which we could collaborate in the future.

With most of the foreign competitors (D. Bernstein, T. Lange, S. Galbraith, F. Vercauteren, F. Hess, M. Scott, K. Kedlaya, A. Sutherland) we are in good terms, with occasional collaborations, invitations, or shared Ph.D. examination jury.

More specifically, on the topic of genus 2 curves, with point counting, fast arithmetic, and explicit isogenies, we have currently a strong position, that we plan to reinforce via our ANR project CHIC. On the topic of pairings, we are leaders in terms of hardware implementation, which is a rather new and active topic. For other aspects of pairings, we are not as present.

9.3 Arithmetic

In terms of multiprecision integer arithmetic, there are currently two main competing teams: the developers of GMP (around T. Granlund) and MPIR (around B. Hart). The second library actually emerged from a fork of the first one in 2008. In this context, our group has a good reputation for providing efficient and reliable code, and to participate to discussions for improvements. We have avoided taking any strong position for one library or the other, although we try to indirectly incite them to stay compatible.

On the floating-point side, we have a high visibility due to the MPFR library which is the reference for reliable multiprecision floating-point arithmetic. It is used as a building block by other libraries and applications (GCC, MPC, MPFI).

For polynomial arithmetic and finite fields, the main competitor is Shoup's NTL library, which is the reference. We are in good terms with Shoup and when possible, our work is ported as a patch for NTL to improve its performance. For multiplications of polynomials over GF(2), the newest version NTL 5.5 can be configured to use our optimized software library GF2X as an add-on.

9.4 Positioning with respect to other project-teams

The project-teams from which it is the most important to explain our positioning are TANC and LFANT. Indeed, CARAMEL and LFANT are partly formed by former members of TANC and they share the same research domain.

Our two research directions "NFS-like algorithms" and "arithmetic" are those for which CARAMEL is the most involved among the 3 project-teams. It does not mean that LFANT and TANC do not work on the topics but these are not in the main objectives of their projects. We collaborate on integer factorization in the CADO project that finishes in January 2010, and LFANT and CARAMEL collaborate via the MPC software.

The main thematic intersection is within our research direction "Algebraic curves and cryptography". The first visible intersection lies in the CM construction in genus 2, that both LFANT and CARAMEL wants to push forward. We will synchronize ourselves and collaborate for the first steps of the research and then distribute the work according to the main interests of both projects: LFANT has expertise in the theory of modular forms that can be used to design better functions in the genus 2 CM context, whereas our interest is to get very fast implementation using interactions with arithmetic building blocks, and to use explicit isogenies. This last topic is another intersection: TANC and CARAMEL both have projects to improve the state of the art. The approaches taken in TANC and CARAMEL are quite different: geometric for TANC and algebraic theta functions in CARAMEL. We are therefore in frequent contact on the topic, so that we can compare the advances of the two methods and we regularly check how they could be combined.

Other parts of this main direction are more specific to CARAMEL: point counting and fast arithmetic in genus 2. Also pairings, especially in hardware is a far more important part of CARAMEL than in TANC or LFANT.

Other project-teams having cryptography as their main applications include SECRET and CASCADE. For SECRET, they are interested in symmetric cryptography, and primitives based on error correcting codes, which is not related to our proposal. CASCADE is doing mainstream cryptography with little interest in the study of the public key primitives (except maybe those based on lattices). Therefore there is essentially no intersection between the SECRET and CASCADE project-teams and CARAMEL.

With the ARENAIRE project-team, there are potential intersections, due to 3 members coming from CACAO: Damien Stehlé, Guillaume Hanrot and Vincent Lefèvre. Stehlé and Hanrot are working on hard problems in lattices, which is no longer a research topic of CARAMEL, and Lefèvre works on floating-point arithmetic and the MPFR software, which is in collaboration with CARAMEL.

10 National and International Collaborations

10.1 At INRIA

- TANC. We collaborate on the topic of integer factorization, since we are 2 of the 3 partners of the ANR CADO project. On the curve topic, and in particular on the explicit isogeny computation, we follow different approaches for the same problem, and we might collaborate in the future to take advantage of both.
- LFANT. We collaborate with the group via the development of the MPC library. We also have some common interest in curves for cryptography and pairings. Several collaborations have occurred in the past with A. Enge (who was a member of TANC until 2008), and we will probably have others, maybe on the Complex Multiplication method in genus 2.
- ARÉNAIRE. We work together on the development of MPFR and on the topic of efficient hardware for pairing computation. There has been some other collaborations with the group in the past (on worst cases for floating-point arithmetic, for instance), and other similar collaborations will most likely occur in the future.

10.2 National

We have good contact with other groups in France, and have occasional collaborations with them. We collaborate in particular with IRMAR (Rennes) and IML (Marseille) in the context of the ANR project "CHIC" which started in September 2009 (see below).

10.3 International

At the international level, we have two long-term collaborations: the first one with Éric Schost (London, Ontario, Canada) on the topic of genus 2 point counting, and the second one with Richard Brent (Canberra, Australia) about efficient arithmetic, in the context of the ANC (Algorithms, Numbers, Computers) associate team.

We have good contacts with Dan Bernstein and Tanja Lange (Eindhoven, Netherlands), with a co-supervised Ph.D. student.

We collaborate with the groups of Arjen Lenstra (Lausanne, Switzerland), Hermann te Riele and Peter Montgomery (Microsoft Research, Redmond), and Kazumaro Aoki (NTT, Japan), on the resolution of the RSA-768 challenge.

We collaborate with Eiji Okamoto and Jean-Luc Beuchat (Tsukuba, Japan) on the topic of hardware implementation of pairings.

10.4 Participation in French or European Projects

- ANR CADO: November 2006 – January 2010. The purpose of this project funded by the French national agency ANR is to study and implement the number field sieve algorithm for factoring integers. It is a collaboration with the TANC project-team and with the number theory group of the Institut Élie Cartan, the institute of mathematics in Nancy. A postdoc (A. Vera) and a Ph.D. student (A. Kruppa) have been hired with this grant.
- ODL MPTools: September 2007 – August 2009. This project funded by INRIA did improve the MPFR and MPC libraries towards their integration within GCC. This was in collaboration with the ARÉNAIRE project-team (Lyon) and Andreas Enge who is now in the LFANT project-team (Bordeaux). An engineer (Ph. Théveny) has been hired with this grant.
- Associated team ANC: until end of 2011. The purpose of this project funded by INRIA is to join our research efforts with the Computational Mathematics group at ANU (Australian National University, Canberra, Australia), based on a long-term and successful cooperation between Richard Brent and Paul Zimmermann. ANC stands for "Algorithms, Numbers, Computers", and the topic is computer arithmetic in a wide sense.
- ANR CHIC: 2009-2012. This project "Courbes Hyperelliptiques: Isogénies et Comptage", funded by the French national agency ANR is in collaboration with IRMAR at Rennes 1 and IML in Marseille. The main objective is to improve algorithms for isogeny computation with a view towards point counting.

11 Selected Publications from Team Members

Articles

[1] Beuchat, J.-L., Brisebarre, N., Detrey, J., Okamoto, E., Shirase, M., and Takagi, T. Algorithms and arithmetic operators for computing the η_T pairing in characteristic three. *IEEE Trans. Comput. 57*, 11 (Nov. 2008), 1454–1468.

[2] Bostan, A., Gaudry, P., and Schost, É. Linear recurrences with polynomial coefficients and application to integer factorization and Cartier-Manin operator. *SIAM J. Comput. 36* (2007), 1777–1806.

[3] Diem, C., and Thomé, E. Index calculus in class groups of non-hyperelliptic curves of genus three. *J. Cryptology 21*, 4 (Oct. 2008), 593–611.

[4] Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., and Zimmermann, P. MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Trans. Math. Softw. 33*, 2 (2007), article 13.

[5] Gaudry, P., and Lubicz, D. The arithmetic of characteristic 2 Kummer surfaces and of elliptic Kummer lines. *Finite Fields and Their Applications 15* (2009), 246–260.

[6] Gaudry, P., Thomé, E., Thériault, N., and Diem, C. A double large prime variation for small genus hyperelliptic index calculus. *Math. Comp. 76*, 257 (Jan. 2007), 475–492.

[7] Thomé, E. Subquadratic computation of vector generating polynomials and improvement of the block Wiedemann algorithm. *J. Symbolic Comput. 33*, 5 (July 2002), 757–775.

Books

[8] Brent, R. P., and Zimmermann, P. *Modern Computer Arithmetic.* Version 0.3, 2009. In preparation. Current version available at `http://www.loria.fr/~zimmerma/mca/pub226.html`.

Conference Communications

[9] Beuchat, J.-L., Brisebarre, N., Detrey, J., Okamoto, E., and Rodríguez-Henríquez, F. A comparison between hardware accelerators for the modified Tate pairing over $\mathbb{F}_{2^m}$ and $\mathbb{F}_{3^m}$. In *Second International Conference on Pairing-Based Cryptography (Pairing'08)* (Egham, UK, Sept. 2008), S. D. Galbraith and K. G. Paterson, Eds., no. 5209 in Lecture Notes in Computer Science, Springer.

[10] Beuchat, J.-L., Detrey, J., Estibals, N., Okamoto, E., and Rodríguez-Henríquez, F. Hardware accelerator for the Tate pairing in characteristic three based on Karatsuba-Ofman multipliers. In *11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'09)* (Lausanne, Switzerland, Sept. 2009), C. Clavier and K. Gaj, Eds., Lecture Notes in Computer Science, Springer. To appear.

[11] Brent, R., Gaudry, P., Thomé, E., and Zimmermann, P. Faster multiplication in GF(2)[x]. In *Proceedings of the 8th International Symposium on Algorithmic Number Theory (ANTS VIII)* (2008), A. J. van der Poorten and A. Stein, Eds., vol. 5011 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 153–166.

[12] Enge, A., and Gaudry, P. An $L(1/3 + \varepsilon)$ algorithm for the discrete logarithm problem for low degree curves. In *Advances in Cryptology – EUROCRYPT 2007* (2007), M. Naor, Ed., vol. 4515 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 379–393.

[13] Gaudry, P., and Schost, É. Construction of secure random curves of genus 2 over prime fields. In *Advances in Cryptology – EUROCRYPT 2004* (2004), C. Cachin and J. Camenisch, Eds., vol. 3027 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 239–256.

[14] Joux, A., Naccache, D., and Thomé, E. When e-th roots become easier than factoring. In *Advances in Cryptology – ASIACRYPT 2007* (2008), vol. 4833 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 13–28.

[15] Zimmermann, P., and Dodson, B. 20 years of ECM. In *Proceedings of the 7th Algorithmic Number Theory Symposium (ANTS VII)* (Berlin Heidelberg, 2006), F. Hess, S. Pauli, and M. Pohst, Eds., vol. 4076 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 525–542.

12 Short Vitae from Permanent Team Members

Jérémie Detrey (29, INRIA research scientist)

2008– INRIA research scientist (CR2) at INRIA Nancy–Grand Est.

2007–2008 Teaching assistant (postdoc) at B-IT (Bonn, Germany).

2003–2007 Ph.D. thesis at LIP (ÉNS Lyon), under the supervision of F. de Dinechin and J.-M. Muller.

2003 M.Sc. in computer science at ÉNS Lyon.

2000 Admission at École normale supérieure de Lyon.

Research interests: Computer arithmetic, cryptology, hardware implementation.

Ph.D. student:

2009– Co-advisor (with P. Gaudry) of N. Estibals's Ph.D. thesis

Pierrick Gaudry (36, CNRS research scientist)

2008 Habilitation à diriger les recherches, University of Nancy 1 (UHP).

2005– CNRS research scientist (CR1) at LORIA.

2001–2005 CNRS research scientist (CR2) at LIX, École polytechnique.

1998–2001 Ph.D. thesis at LIX (École polytechnique), under the supervision of F. Morain.

1995 M.Sc. in computer science at École polytechnique.

1993 Admission at École normale supérieure de Cachan.

Research interests: Computational number theory, cryptology.

Responsibilities within the scientific community:

2006–2009 Coordinator of the CADO project of the ANR Blanc

2003–2005 Vice-head of the TANC project-team at INRIA Futurs

Ph.D. students:

2009– Co-advisor (with J. Detrey) of N. Estibals's Ph.D. thesis

2008– Co-advisor (with T. Lange, TU Eindhoven) of G. Bisson's Ph.D. thesis

2004– Co-advisor (with F. Morain) of T. Houtmann's Ph.D. thesis

Emmanuel Thomé (32, INRIA research scientist)

2006– INRIA research scientist (CR1) at INRIA Nancy–Grand Est.

2003–2006 INRIA research scientist (CR2) at INRIA Lorraine.

1999–2003 Ph.D. thesis at LIX (École polytechnique), under the supervision of F. Morain.

1997 MSc. in computer science at École polytechnique.

1995 Admission at École normale supérieure.

Research interests: Computational number theory, linear algebra.

Responsibilities within the scientific community:

2009–2012 Partner contact for the CHIC project of the ANR Blanc (project CHIC coordinated in Rennes).

Ph.D. student:

2008– Co-advisor (with G. Hanrot) of R. Cosset's Ph.D. thesis.

Paul Zimmermann (45, INRIA research director)

2001 Habilitation à diriger les recherches, University of Nancy 1 (UHP).

1992– INRIA research scientist at INRIA Nancy–Grand Est.

1987–1991 Ph.D. thesis at INRIA Rocquencourt, under the supervision of Ph. Flajolet.

Research interests: Computational number theory, arbitrary precision arithmetic.

Responsibilities within the scientific community:

2006–2009 Vice-head of the CACAO project-team at INRIA Nancy–Grand Est.

2000–2006 Head of the Nancy part of the Spaces project-team.

1998–2000 Head of the PolKA project-team at INRIA Lorraine.

2005–2007 Elected member of INRIA Evaluation Board.

1999–2001 Elected member of INRIA Evaluation Board.

2001– Member of the Program Committee of the Arith conference.

Ph.D. students:

2007– Advisor of A. Kruppa's Ph.D. thesis.

2003–2006 Advisor of L. Fousse's Ph.D. thesis.

2003–2005 Advisor of D. Stehlé's Ph.D. thesis.

1994–1997 Co-advisor (with P. Lescanne) of F. Bertault's Ph.D. thesis.