Activity Report 2018

Project-Team MARELLE

Mathematics, Reasoning, and Software

RESEARCH CENTER
Sophia Antipolis - Méditerranée

THEME
Proofs and Verification

Table of contents

Project-Team MARELLE

Creation of the Project-Team: 2006 November 01

Keywords:

Computer Science and Digital Science:

A2.1.11. - Proof languages
A2.4.3. - Proofs
A4.5. - Formal methods for security
A5.10.3. - Planning
A7.2. - Logic in Computer Science
A7.2.3. - Interactive Theorem Proving
A7.2.4. - Mechanized Formalization of Mathematics
A8.3. - Geometry, Topology
A8.4. - Computer Algebra
A8.10. - Computer arithmetic

Other Research Topics and Application Domains:

B6.1. - Software industry
B9.5.1. - Computer science
B9.5.2. - Mathematics

1. Team, Visitors, External Collaborators

Research Scientists

Yves Bertot [Team leader, Inria, Senior Researcher, HDR]
Cyril Cohen [Inria, Researcher]
José Grimm [Inria, Researcher]
Benjamin Grégoire [Inria, Researcher]
Laurence Rideau [Inria, Researcher]
Enrico Tassi [Inria, Researcher]
Laurent Théry [Inria, Researcher]

Post-Doctoral Fellow

Frank Florian Steinberg [Inria, until Sep 2018]

PhD Students

Cécile Baritel-Ruet [Ecole Normale Supérieure Cachan]
Sophie Bernard [Univ de Nice - Sophia Antipolis]
Boris Djalal [Inria, until Sep 2018]
Mohamad El Laz [Inria]
Damien Rouhling [Ministère de l'Enseignement Supérieur et de la Recherche]

Technical staff

Maxime Dénès [Inria Foundation, until Nov 2018, Inria since then]

Administrative Assistant

Nathalie Bellesso [Inria]

Visiting Scientists

Sunjay Cauligi [University of California San Diego, from Sep 2018 until Nov 2018]

Joshua Gancher [Cornell University, from Sep 2018 until Nov 2018]
Vincent Laporte [IMDEA Madrid, until Jun 2018]

External Collaborators

Gilles Barthe [IMDEA Madrid, HDR]
Loïc Pottier [Ministère de l'Education Nationale, HDR]

2. Overall Objectives

2.1. Overall Objectives

We want to concentrate on the development of mathematical libraries for theorem proving tools. This objective contributes to two main areas of application: tools for mathematicians and correctness verification tools for software dealing with numerical computation.

In the short term, we aim for mathematical libraries that concern polynomials, algebra, group theory, floating point numbers, real numbers, big integers, probabilities and geometrical objects. In the long run, we think that this will involve any function that may be of use in embedded software for control or robotics (in what is called hybrid systems, systems that contain both software and physical components) and in cryptographical systems. We want to integrate these libraries in theorem proving tools because we believe they will become important tools for mathematical practice and for engineers who need to prove the correctness of their algorithms and software.

We believe that theorem proving tools are good tools to produce highly dependable software, because they provide a framework where algorithms and specifications can be studied uniformly and often provide means to mechanically derive programs that are correct by construction.

We also study the extensibility of interactive theorem proving tools based on decision procedures that free designers from the burden of verifying some of the required properties. We often rely on "satisfiability modulo theory" procedures, which can be connected to theorem proving tools in a way that preserves the trustability of the final results.

3. Research Program

3.1. Type theory and formalization of mathematics

The calculus of inductive constructions is a branch of type theory that serves as a foundation for theorem proving tools, especially the Coq proof assistant. It is powerful enough to formalize complex mathematics, based on algebraic structures and operations. This is especially important as we want to produce proofs of logical properties for these algebraic structures, a goal that is only marginally addressed in most scientific computation systems.

The calculus of inductive constructions also makes it possible to write algorithms as recursive functional programs which manipulate tree-like data structures. A third important characteristic of this calculus is that it is also a language for manipulating proofs. All this makes this calculus a tool of choice for our investigations. However, this language still is the object of improvements and part of our work focusses on these improvements.

3.2. Verification of scientific algorithms

To produce certified algorithms, we use the following approach: instead of attempting to prove properties of an existing program written in a conventional programming language such as C or Java, we produce new programs in the calculus of constructions whose correctness is an immediate consequence of their construction. This has several advantages. First, we work at a high level of abstraction, independently of the target implementation language. Secondly, we concentrate on specific characteristics of the algorithm, and abstract away from the rest (for instance, we abstract away from memory management or data implementation strategies). Therefore, we are able to address more high-level mathematics and to express more general properties without being overwhelmed by implementation details.

However, this approach also presents a few drawbacks. For instance, the calculus of constructions usually imposes that recursive programs should explicitly terminate for all inputs. For some algorithms, we need to use advanced concepts (for instance, well-founded relations) to make the property of termination explicit, and proofs of correctness become especially difficult in this setting.

3.3. Programming language semantics

To bridge the gap between our high-level descriptions of algorithms and conventional programming languages, we investigate the algorithms that are present in programming language implementations, for instance algorithms that are used in a compiler or a static analysis tool. When working on these algorithms, we usually base our work on the semantic description of the programming language. The properties that we attempt to prove for an algorithm are, for example, that an optimization respects the meaning of programs or that the programs produced are free of some unwanted behavior. In practice, we rely on this study of programming language semantics to propose extensions to theorem proving tools or to verify that compilers for conventional programming languages are exempt from bugs.

4. Highlights of the Year

4.1. Highlights of the Year

4.1.1. Awards

The paper by Barthe, Grégoire, and Laporte at *Computer Security Foundations* on cryptographic constant-time was awarded a distinguished paper award.

BEST PAPER AWARD:

[16]
G. BARTHE, B. GRÉGOIRE, V. LAPORTE. *Secure Compilation of Side-Channel Countermeasures: The Case of Cryptographic "Constant-Time"*, in "CSF 2018 - 31st IEEE Computer Security Foundations Symposium", Oxford, United Kingdom, July 2018, https://hal.archives-ouvertes.fr/hal-01959560

5. New Software and Platforms

5.1. Coq

The Coq Proof Assistant
KEYWORDS: Proof - Certification - Formalisation

SCIENTIFIC DESCRIPTION: Coq is an interactive proof assistant based on the Calculus of (Co-)Inductive Constructions, extended with universe polymorphism. This type theory features inductive and co-inductive families, an impredicative sort and a hierarchy of predicative universes, making it a very expressive logic. The calculus allows to formalize both general mathematics and computer programs, ranging from theories of finite structures to abstract algebra and categories to programming language metatheory and compiler verification. Coq is organised as a (relatively small) kernel including efficient conversion tests on which are built a set of higher-level layers: a powerful proof engine and unification algorithm, various tactics/decision procedures, a transactional document model and, at the very top an IDE.
FUNCTIONAL DESCRIPTION: Coq provides both a dependently-typed functional programming language and a logical formalism, which, altogether, support the formalisation of mathematical theories and the specification and certification of properties of programs. Coq also provides a large and extensible set of automatic or semi-automatic proof methods. Coq's programs are extractible to OCaml, Haskell, Scheme, ...

RELEASE FUNCTIONAL DESCRIPTION: Coq version 8.8.2 contains the result of refinements and stabilization of features and deprecations, cleanups of the internals of the system along with a few new features.

Summary of changes:

Kernel: fix a subject reduction failure due to allowing fixpoints on non-recursive values (#407), by Matthieu Sozeau. Handling of evars in the VM (#935) by Pierre-Marie Pédrot.

Notations: many improvements on recursive notations and support for destructuring patterns in the syntax of notations by Hugo Herbelin.

Proof language: tacticals for profiling, timing and checking success or failure of tactics by Jason Gross. The focusing bracket { supports single-numbered goal selectors, e.g. 2:{, (#6551) by Théo Zimmermann.

Vernacular: cleanup of definition commands (#6653) by Vincent Laporte and more uniform handling of the Local flag (#1049), by Maxime Dénès. Experimental Show Extraction command (#6926) by Pierre Letouzey. Coercion now accepts Prop or Type as a source (#6480) by Arthur Charguéraud. Export modifier for options allowing to export the option to modules that Import and not only Require a module (#6923), by Pierre-Marie Pédrot.

Universes: many user-level and API level enhancements: qualified naming and printing, variance annotations for cumulative inductive types, more general constraints and enhancements of the minimization heuristics, interaction with modules by Gaëtan Gilbert, Pierre-Marie Pédrot and Matthieu Sozeau.

Library: Decimal Numbers library (#6599) by Pierre Letouzey and various small improvements.

Documentation: a large community effort resulted in the migration of the reference manual to the Sphinx documentation tool. The new documentation infrastructure (based on Sphinx) is by Clément Pit-Claudel. The migration was coordinated by Maxime Dénès and Paul Steckler, with some help of Théo Zimmermann during the final integration phase. The 14 people who ported the manual are Calvin Beck, Heiko Becker, Yves Bertot, Maxime Dénès, Richard Ford, Pierre Letouzey, Assia Mahboubi, Clément Pit-Claudel, Laurence Rideau, Matthieu Sozeau, Paul Steckler, Enrico Tassi, Laurent Théry, Nikita Zyuzin.

Tools: experimental -mangle-names option to coqtop/coqc for linting proof scripts (#6582), by Jasper Hugunin. Main changes:

Critical soundness bugs were fixed between versions 8.8.0 and 8.8.2, and a PDF version of the reference manual was made available. The Windows installer also includes many more external packages that can be individually selected for installation.

On the implementation side, the dev/doc/changes.md file documents the numerous changes to the implementation and improvements of interfaces. The file provides guidelines on porting a plugin to the new version.

More information can be found in the CHANGES file. Feedback and bug reports are extremely welcome.

Distribution Installers for Windows 32 bits (i686), Windows 64 bits (x8_64) and macOS are available. They come bundled with CoqIDE. Windows binaries now include the Bignums library.

Complete sources of the files installed by the Windows installers are made available, to comply with license requirements.
NEWS OF THE YEAR: Version 8.8.0 was released in April 2018 and version 8.8.2 in September 2018. This is the third release of Coq developed on a time-based development cycle. Its development spanned 6 months from the release of Coq 8.7 and was based on a public road-map. It attracted many external contributions. Code reviews and continuous integration testing were systematically used before integration of new features, with an important focus given to compatibility and performance issues.

The main advances in this version are cleanups and fixes in the many different components of the system, ranging from low level kernel fixes to advances in the support of notations and tacticals for selecting goals. A large community effort was made to move the documentation to the Sphinx format, providing a more accessible online ressource to users.

- Participants: Abhishek Anand, C. J. Bell, Yves Bertot, Frédéric Besson, Tej Chajed, Pierre Courtieu, Maxime Denes, Julien Forest, Emilio Jesús Gallego Arias, Gaëtan Gilbert, Benjamin Grégoire, Jason Gross, Hugo Herbelin, Ralf Jung, Matej Kosik, Sam Pablo Kuper, Xavier Leroy, Pierre Letouzey, Assia Mahboubi, Cyprien Mangin, Érik Martin-Dorel, Olivier Marty, Guillaume Melquiond, Pierre-Marie Pédrot, Benjamin C. Pierce, Lars Rasmusson, Yann Régis-Gianas, Lionel Rieg, Valentin Robert, Thomas Sibut-Pinote, Michael Soegtrop, Matthieu Sozeau, Arnaud Spiwack, Paul Steckler, George Stelle, Pierre-Yves Strub, Enrico Tassi, Hendrik Tews, Laurent Théry, Amin Timany, Vadim Zaliva and Théo Zimmermann
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Matthieu Sozeau
- Publication: The Coq Proof Assistant, version 8.8.0
- URL: http://coq.inria.fr/

5.2. Easycrypt

FUNCTIONAL DESCRIPTION: EasyCrypt is a toolset for reasoning about relational properties of probabilistic computations with adversarial code. Its main application is the construction and verification of game-based cryptographic proofs. EasyCrypt can also be used for reasoning about differential privacy.

- Participants: Benjamin Grégoire, Gilles Barthe and Pierre-Yves Strub
- Contact: Gilles Barthe
- URL: https://www.easycrypt.info/trac/

5.3. ELPI

Embeddable Lambda Prolog Interpreter
KEYWORDS: Constraint Programming - Programming language - Higher-order logic
SCIENTIFIC DESCRIPTION: The programming language has the following features

- Native support for variable binding and substitution, via an Higher Order Abstract Syntax (HOAS) embedding of the object language. The programmer needs not to care about De Bruijn indexes.

- Native support for hypothetical context. When moving under a binder one can attach to the bound variable extra information that is collected when the variable gets out of scope. For example when writing a type-checker the programmer needs not to care about managing the typing context.

- Native support for higher order unification variables, again via HOAS. Unification variables of the meta-language (lambdaProlog) can be reused to represent the unification variables of the object language. The programmer does not need to care about the unification-variable assignment map and cannot assign to a unification variable a term containing variables out of scope, or build a circular assignment.

- Native support for syntactic constraints and their meta-level handling rules. The generative semantics of Prolog can be disabled by turning a goal into a syntactic constraint (suspended goal). A syntactic constraint is resumed as soon as relevant variables gets assigned. Syntactic constraints can be manipulated by constraint handling rules (CHR).

- Native support for backtracking. To ease implementation of search.

- The constraint store is extensible. The host application can declare non-syntactic constraints and use custom constraint solvers to check their consistency.

- Clauses are graftable. The user is free to extend an existing program by inserting/removing clauses, both at runtime (using implication) and at "compilation" time by accumulating files.

Most of these feature come with lambdaProlog. Constraints and propagation rules are novel in ELPI.
FUNCTIONAL DESCRIPTION: ELPI implements a variant of lambdaProlog enriched with Constraint Handling Rules, a programming language well suited to manipulate syntax trees with binders and unification variables.

ELPI is a research project aimed at providing a programming platform for the so called elaborator component of an interactive theorem prover.

ELPI is designed to be embedded into larger applications written in OCaml as an extension language. It comes with an API to drive the interpreter and with an FFI for defining built-in predicates and data types, as well as quotations and similar goodies that come in handy to adapt the language to the host application.

RELEASE FUNCTIONAL DESCRIPTION: First public release
NEWS OF THE YEAR: First public release

- Participant: Claudio Sacerdoti Coen
- Contact: Enrico Tassi
- Publications: ELPI: fast, Embeddable, λProlog Interpreter - Implementing Type Theory in Higher Order Constraint Logic Programming
- URL: https://github.com/lpcic/elpi/

5.4. Math-Components

Mathematical Components library
KEYWORD: Proof assistant
FUNCTIONAL DESCRIPTION: The Mathematical Components library is a set of Coq libraries that cover the prerequiste for the mechanization of the proof of the Odd Order Theorem.

RELEASE FUNCTIONAL DESCRIPTION: The library includes 16 more theory files, covering in particular field and Galois theory, advanced character theory, and a construction of algebraic numbers.

- Participants: Alexey Solovyev, Andrea Asperti, Assia Mahboubi, Cyril Cohen, Enrico Tassi, François Garillot, Georges Gonthier, Ioana Pasca, Jeremy Avigad, Laurence Rideau, Laurent Théry, Russell O'Connor, Sidi Ould Biha, Stéphane Le Roux and Yves Bertot
- Contact: Assia Mahboubi
- URL: http://math-comp.github.io/math-comp/

5.5. Semantics

KEYWORDS: Semantic - Programming language - Coq
FUNCTIONAL DESCRIPTION: A didactical Coq development to introduce various semantics styles. Shows how to derive an interpreter, a verifier, or a program analyser from formal descriptions, and how to prove their consistency.

This is a library for the Coq system, where the description of a toy programming language is presented. The value of this library is that it can be re-used in classrooms to teach programming language semantics or the Coq system. The topics covered include introductory notions to domain theory, pre and post-conditions, abstract interpretation, and the proofs of consistency between all these point of views on the same programming language. Standalone tools for the object programming language can be derived from this development.

- Participants: Christine Paulin and Yves Bertot
- Contact: Yves Bertot
- URL: http://www-sop.inria.fr/members/Yves.Bertot/proofs/semantics_survey.tgz

5.6. Ssreflect

FUNCTIONAL DESCRIPTION: Ssreflect is a tactic language extension to the Coq system, developed by the Mathematical Components team.

- Participants: Assia Mahboubi, Cyril Cohen, Enrico Tassi, Georges Gonthier, Laurence Rideau, Laurent Théry and Yves Bertot
- Contact: Yves Bertot
- URL: http://math-comp.github.io/math-comp/

5.7. AutoGnP

KEYWORDS: Formal methods - Security - Cryptography
FUNCTIONAL DESCRIPTION: autoGnP is an automated tool for analyzing the security of padding-based public-key encryption schemes (i.e. schemes built from trapdoor permutations and hash functions). This years we extended the tool to be able to deal with schemes based on cyclic groups and bilinear maps.

- Participants: Benjamin Grégoire, Gilles Barthe and Pierre-Yves Strub
- Contact: Gilles Barthe
- URL: https://github.com/ZooCrypt/AutoGnP

6. New Results

6.1. Extension language for Coq

Participants: Enrico Tassi, Feruccio Guidi [University of Bologna], Claudio Sacerdoti Coen [University of Bologna].

We continued our work on the design of a language mixing λ-prolog and constraint programming. This year, we redesigned and provided a new implementation of the constraint handling rules, leading to a first public release of the software. We are starting to have users beyond our own team:

- (Inria/Parsifal) MLTS https://github.com/voodoos/mlts
- (Inria/Parsifal) proofcert https://github.com/proofcert/checkers
- (UML.eu) Lang-n-play https://github.com/mcimini/lang-n-play

In an article submitted for publication [24], we showed that Elpi could be used to give a short implementation of Type Theory.

We are also starting a collaboration to construct an elaborator for HOL-Light using Elpi.

6.2. Deriving equality tests

Participant: Enrico Tassi.

In type theory, for most inductive types, it is possible to construct a two-argument boolean function that tests when two terms of the type are equal. When inductive types have constructors containing sub-components from another inductive, this needs to be done in a modular way. This year, we studied how this problem could be solved in a modular way using Elpi. It turns out that the unary parametricity translation can serve as a tool to make the derivation compositional. This is described in a pre-print [25].

6.3. Parametricity proofs

Participants: Cyril Cohen, Abishek Anand [Cornell University], Simon Boulier [Inria Gallinette], Matthieu Sozeau [Inria Pi.r2], Nicolas Tabareau [Inria Gallinette], Robert Y. Lewis [Vrije Universiteit Amsterdam], Johannes Hölzl [CMU, Pittsburgh, USA and Vrije Universiteit, Amsterdam, the Netherlands].

After our previous experiment using Elpi to develop a tool that produces parametricity proofs, we investigated the use of the *Template-Coq* framework to implement this kind of algorithm. This work is described in [11]. A similar experiment has been performed using the Lean theorem prover.

6.4. Proving Expected Sensitivity of Probabilistic Programs

Participants: Benjamin Grégoire, Gilles Barthe [IMDEA], Thomas Espitau [UPMC Paris 6], Justin Hsu [University of Pennsylvania], Pierre-Yves Strub [Ecole Polytechnique].

Program sensitivity, also known as Lipschitz continuity, describes how small changes in a program's input lead to bounded changes in the output. We propose an average notion of program sensitivity for probabilistic programs—expected sensitivity—that averages a distance function over a probabilistic coupling of two output distributions from two similar inputs. This work is described in [8].

6.5. An Assertion-Based Program Logic for Probabilistic Programs

Participants: Benjamin Grégoire, Gilles Barthe [IMDEA], Thomas Espitau [UPMC Paris 6], Marco Gaboardi [University at Buffalo, SUNY], Justin Hsu [University of Pennsylvania], Pierre-Yves Strub [Ecole Polytechnique].

We have developed Ellora, a sound and relatively complete assertion-based program logic, and demonstrate its expressivity by verifying several classical examples of randomized algorithms using an implementation in the EasyCrypt proof assistant. Ellora features new proof rules for loops and adversarial code, and supports richer assertions than existing program logics. We also show that Ellora allows convenient reasoning about complex probabilistic concepts by developing a new program logic for probabilistic independence and distribution law, and then smoothly embedding it into Ellora. This is described in article [14].

6.6. Vectorizing Higher-Order Masking

Participants: Benjamin Grégoire, Kostas Papagiannopoulos [Radboud University], Peter Schwabe [Radboud University], Ko Stoffelen [Radboud University].

The cost of higher-order masking as a countermeasure against side-channel attacks is often considered too high for practical scenarios, as protected implementations become very slow. At Eurocrypt 2017, we have proposed the bounded moment leakage model to study the (theoretical) security of parallel implementations of masking schemes. In this work we show how the NEON vector instructions of larger ARM Cortex-A processors can be exploited to build much faster masked implementations of AES based on the bounded moment model. This work is described in publication [18].

6.7. Masking the GLP Lattice-Based Signature Scheme at Any Order

Participants: Benjamin Grégoire, Gilles Barthe [IMDEA], Sonia Belaïd [CryptoExpert], Thomas Espitau [UPMC Paris 6], Pierre-Alain Fouque [Université Rennes 1], Mélissa Rossi [ENS Paris], Mehdi Tibouchi [NTT].

Recently, numerous physical attacks have been demonstrated against lattice based schemes, often exploiting their unique properties such as the reliance on Gaussian distributions, rejection sampling and FFT-based polynomial multiplication. In this work, we describe the first masked implementation of a lattice-based signature scheme. Since masking Gaussian sampling and other procedures involving contrived probability distribution would be prohibitively inefficient, we focus on the GLP scheme. This work is described in [13].

6.8. Symbolic Proofs for Lattice-Based Cryptography

Participants: Benjamin Grégoire, Gilles Barthe [IMDEA], Xiong Fan [Cornell], Joshua Gancher [Cornell], Charlie Jacomme [LSV], Elaine Shi [Cornell].

Symbolic methods have been used extensively for proving security of cryptographic protocols in the Dolev-Yao model, and more recently for proving security of cryptographic primitives and constructions in the computational model. However, existing methods for proving security of cryptographic constructions in the computational model often require significant expertise and interaction, or are fairly limited in scope and expressivity. In this work we introduce a symbolic approach for proving security of cryptographic constructions based on the Learning With Errors assumption. This work is described in [15].

6.9. Formal Security Proof of CMAC and Its Variants

Participants: Benjamin Grégoire, Cécile Baritel-Ruet, François Dupressoir [University of Surrey], Pierre-Alain Fouque [Université Rennes 1].

The CMAC standard, when initially proposed by Iwata and Kurosawa as OMAC1, was equipped with a complex game-based security proof. Following recent advances in formal verification for game-based security proofs, we have formalized a proof of unforgeability for CMAC in EasyCrypt. This work is described in [12].

6.10. Secure Compilation of Side-Channel Countermeasures: The Case of Cryptographic "Constant-Time"

Participants: Benjamin Grégoire, Gilles Barthe [IMDEA], Vincent Laporte [IMDEA].

Software-based countermeasures provide effective mitigation against side-channel attacks, often with minimal efficiency and deployment overheads. Their effectiveness is often amenable to rigorous analysis: specifically, several popular countermeasures can be formalized as information flow policies, and correct implementation of the countermeasures can be verified with state-of-the-art analysis and verification techniques. However, in absence of further justification, the guarantees only hold for the language (source, target, or intermediate representation) on which the analysis is performed. We consider the problem of preserving side-channel counter-measures by compilation for cryptographic “constant-time”, a popular countermeasure against cache-based timing attacks. We have presented a general method, based on the notion of constant-time-simulation, for proving that a compilation pass preserves the constant-time countermeasure. This work was described in [16]. At the conference, this work received the “distinguished paper” award.

6.11. Hypotheses of Decisional Diffie-Hellmann

Participants: Benjamin Grégoire, Mohamad El Laz, Tamara Rezk [Inria, Indes project team].

In the thesis work of Mohamad El Laz, co-supervised by Benjamin Grégoire and Tamara Rezk (Indes project-team), we studied the cryptographic hypothesis of DDH (Decisional Diffie-Hellman) and implementations that would break this hypothesis. We focused on ElGamal encryption cryptosystem implementations to assess they use the DDH hypothesis correctly. We analyzed a number of implementations including Botan, Belenios and Libgcrypt. The lessons learned from this analysis are that the hypotheses are not always well understood.

In a second stage we considered message encoding methods. We investigated several approaches such as DCDH (Decisional Class Diffie-Hellman) in Encoding-Free ElGamal Encryption.

6.12. Proving the domain management protocol

Participants: José Bacelar Almeida [INESC TEC], Manuel Barbosa [INESC TEC], Gilles Barthe [IMDEA], Benjamin Grégoire, Vitor Pereira [INESC TEC], Bernardo Portela [INESC TEC], Benedikt Schmidt [Google Inc.], François-Xavier Standaert [Université Catholique de Louvain], Pierre-Yves Strub [Ecole Polytechnique].

We have performed a machine-checked proof of security for the domain management protocol of Amazon Web Services KMS (Key Management Service), a critical security service used throughout AWS and by AWS customers. Domain management is at the core of KMS; it governs the long-term keys that anchor the security of encryption services at AWS. Informally, we show that the protocol securely implements a distributed encryption mechanism. Formally, the proof shows that the domain management protocol is indistinguishable from an ideal encryption functionality under standard cryptographic assumptions.

6.13. Formalized graph theory algorithms

Participants: Cyril Cohen, Laurent Théry, Ran Chen [Chinese Academy of Science], Jean-Jacques Lévy [Inria Pi.r2], Stephan Merz [Inria Veridis].

We formalise the correctness proof of Tarjan's algorithm for computing strongly connected components using the Mathematical Component Library. This leads to a comparison of formalisation between various systems described in [22].

6.14. Formal study of a triangulation algorithm

Participant: Yves Bertot.

In work from 2010, a formal description of Delaunay triangulations was presented where the input was a triangulation not satisfying the Delaunay criterion and where the output was a triangulation satisfying this criterion.

In this work, we wish to complete the previous work by describing an algorithm that produces the initial triangulation. We plan this work in several phases, where the first phase only uses simple data-structures, more advanced structures being introduced only later. This work was presented partially in an invited talk at the ICTAC conference [10].

6.15. Formalizing Bourbaki-style mathematics

Participant: José Grimm.

Most of the work described here is inspired by the experiment of giving formal proofs in Coq of the exercises found in Bourbaki's exposition of set theory. However, some of the results go beyond what can be found in Bourbaki.

We implemented a paper of Sierpinski about properties of continuous ordinal functions and limits of such functions.

We implemented a paper on sums of sequences of ordinals, showing that the value obtained (which depends on the order) lies in a finite set. We also showed that this result does not hold when replacing ordinals by order types.

We implemented a paper by Tarski that says if every infinite cartinal is equal to its square, then every set can be well-ordered (this is the axiom of choice). We had to modify our library to make the use of the axiom of choice more explicit.

We continued implementing in Coq the Exercises of Set Theory of Bourbaki. We solved two of them, and proved by a counter example that three of them are false.

6.16. Formal study of double-word arithmetic algorithms

Participants: Laurence Rideau, Jean-Michel Muller [CNRS and ENS Lyon], Valentina Popescu [CNRS and ENS Lyon], Mioara Joldes [CNRS LAAS].

As part of the ANR Fastrelax project, we are formalizing double-word arithmetic algorithms, in particular the sum of a double-word and a floating point number and the sum of two double-word numbers described in the article " Tight and rigourous error bounds for basic building blocks of double-word arithmetic" [27]. The formalization is progressing, moving from addition to multiplication. The progress is slowed down because minor errors in the informal proofs are regularly uncovered, which requires a dialog with the initial authors.

6.17. Proofs of transcendence

Participants: Sophie Bernard, Yves Bertot, Laurence Rideau.

The work on proofs of transcendence that was started the previous year was completed this year by an effort to integrate generic part of the proofs in the Mathematical Components library. A public package for easy re-use by other researchers was also developed.

6.18. Abel's theorem

Participants: Sophie Bernard, Yves Bertot, Cyril Cohen, Laurence Rideau, Assia Mahboubi [Inria Gallinette], Russell O'Connor [McMaster University].

A natural extension of the work on group theory is a proof that polynomials of degree higher than 5 cannot be solved by radicals. This is known as Abel's theorem. We have started an experiment to give a formal proof of this result on top of the Mathematical Components library.

6.19. Formalizing Hermitian Forms

Participants: Cyril Cohen, Laurence Rideau.

We updated the representation and relevant theorems for bilinear, sesquilinear, and hermitian forms in the Mathematical Components library and updated the archived proof of the odd-order theorem (Feit-Thompson) to use the new presentation. This work also includes a proof of the Spectral Theorem.

6.20. Mathematical Components Analysis

Participants: Cyril Cohen, Damien Rouhling, Reynald Affeldt [AIST Japan], Assia Mahboubi [Inria Gallinette], Pierre-Yves Strub [Ecole Polytechnique].

As a synthesis of the lessons learned in the usage of Mathematical Components and Coquelicot, we develop an extension of the Mathematical Components library to cover questions of analysis. This work includes a new tactic called `near` to handle reasoning steps around limits and filters and little-o notation (following Landau's style of asymptotic reasoning). This work is described in [6]. There also contains a new formalization of topoligical structures, Rolle's theorem, the intermediate value theorem, and Heine Borel's theorem. Ongoing work concentrates on a better design of the topological hierarchy and a simplification of the properties expected from real numbers (following a design by A. Mahboubi and P.-Y. Strub).

Some of this work also includes experiments performed with the LEAN theorem prover (developed at Microsoft Research).

6.21. Rigorous Polynomial Approximation

Participants: Florian Steinberg, Laurent Théry.

We have developed a certified library for computing Chebyshev models for formulas composed of polynomials, exponential, logarithm, and trigonometric function. This work is part of the ANR project FastRelax. The code is available at https://github.com/FlorianSteinberg/Cheby

6.22. Formalization of proofs in control theory

Participants: Damien Rouhling, Cyril Cohen.

Damien Rouhling presented his work on formalizing control theory for an inverted pendulum at an international conference in January [19].

The original development was based on Coquelicot. An analysis of the difficulties in formalizing led to the design of Mathematical Components Analysis. The development on control was then ported to this new library. This work was presented at the Coq Workshop in July.

6.23. Formalizing Cylindrical Algebraic Decomposition

Participants: Boris Djalal, Yves Bertot, Cyril Cohen.

Our study of cylindrical algebraic decomposition requires that we find a good representation of semi-algebraic sets. An article on this topic was published [17]. This is also the one of the main topics of Boris Djalal's thesis, which was defended in December.

6.24. A type theory for Algebraic Structures

Participants: Cyril Cohen, Assia Mahboubi, Xavier Montillet.

In collaboration with members of the Inria Gallinette team, we are investigating the properties that a type theory should enjoy to support algebraic structures better than what is currently available.

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

Together with IMDEA Madrid (Spain), INESC TEC (Portugal), the Catholic University of Louvain (Belgium), Google, and Ecole Polytechnique, with have a contract with Amazon Web Services. The financial return for Marelle is 67kEuros.

8. Partnerships and Cooperations

8.1. National Initiatives

8.1.1. ANR

We are currently members of four projects funded by the French national agency for research funding.

- TECAP "Analyse de protocoles, Unir les outils existants", starting on October 1st, 20117, for 60 months, with a grant of 89 kEuros. Other partners are Inria teams PESTO (Inria Nancy grand-est), Ecole Polytechnique, ENS Cachan, IRISA Rennes, and CNRS. The corresponding researcher for this contract is Benjamin Grégoire.
- SafeTLS "La sécurisation de l'Internet du futur avec TLS 1.3" started on October 1st, 2016, for 60 months, with a grant of 147kEuros. Other partners are Université de Rennes 1, and secrétariat Général de la Défense et de la Sécurité Nationale. The corresponding researcher for this contract is Benjamin Grégoire.
- BRUTUS "Chiffrements authentifiés et résistants aux attaques par canaux auxiliaires", started on October 1st, 2014, for 60 months, with a grant of 41 kEuros for Marelle. Other partners are Université de Rennes 1, CNRS, secrétariat Général de la défense et de la sécurité nationale, and Université des Sciences et Technologies de Lille 1. The corresponding researcher for this contract is Benjamin Grégoire.
- FastRelax, "Fast and Reliable Approximations", started on October 1st, 2014, for 60 months, with a grant of 75 kEuros for Marelle. Other partners are Inria Grenoble (ARIC project-team), LAAS-CNRS (Toulouse), Inria Saclay (Toccata and Specfun project-teams), and LIP6-CNRS (Paris). The corresponding researcher for this contract is Laurence Rideau.

8.1.2. FUI

The acronym *FUI* stands for “fonds unique interministériel” and is aimed at research and development projects in pre-industrial phase. The Marelle team is part of one such project.

- VERISICC (formal verification for masking techniques for security against side-channel attacks), This contracts concerns 5 partners: CRYPTOEXPERTS a company from the Paris region (île de France), ANSSI (Agence Nationale de Sécurité des Systèmes d’Information), Oberthur Technologies, University of Luxembourg, and Marelle. A sixth company (Ninjalabs) acts as a sub-contractant. The financial grant for Marelle is 391 kEuros, including 111kEuros that are reserved for the sub-contractant. This project started in October 2018 for a duration of 4 years. The corresponding researcher for this contract is Benjamin Grégoire.

8.2. International Research Visitors

8.2.1. Visits of International Scientists

8.2.1.1. Internships

Joshua Gansher from Cornell and Sunjay Cauligi from the University of California at San Diego visited for three months, as part of their PhD training.

Vincent Laporte from IMDEA Madrid visited for 9 months.

Benoît Viguier from Radboud University, Nijmegen visited for 1 month.

8.2.2. Visits to International Teams

Yves Bertot visited AIST in February in Tsukuba, Japan, ITU Copenhagen in April in Copenhagen, Denmark, and the DeepSpec Summer School in July at Princeton University.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific Events Organisation

9.1.1.1. Member of the Organizing Committees

Yves Bertot is member of steering committee for the conferences ITP, CPP and UITP.

Yves Bertot organized the Coq Implementor’s Workshop in May in Nice, France, where Cyril Cohen, Maxime Dénès, and Enrico Tassi also brought support to newcomers.

Laurence Rideau Organized a meeting of the ANR FastRelax project in June in Sophia Antipolis. There were presentations by Sophie Bernard, Yves Bertot, Cyril Cohen, Damien Rouhling, Laurent Théry during this meeting.

9.1.2. Scientific Events Selection

9.1.2.1. Member of the Conference Program Committees

Benjamin Grégoire was a Program Committee member for CSF 2018 and JFLA 2019. Enrico Tassi was a Program Committee member for CPP 2019, ITP 2018, UITP 2018, F-IDE 2018. Laurent Théry was a Program Committee member for AISC, CPP 2019, ITP 2018, and UITP 2018. Yves Bertot was a Program Committee member for AISC, CICM, FMM, and UITP.

9.1.3. Journal

9.1.3.1. Reviewer - Reviewing Activities

Laurent Théry was a reviewer for *Annals of Mathematics and Artificial Intelligence*, *Journal of Applied Logic*, and *Science of Computer Programming*. Cyril Cohen was a reviewer for *Journal of Automated Reasoning* and *Mathematical Structures in Computer Science*. Enrico Tassi was a reviewer for *ACM Transactions on Computational Logic* and *Journal of Automated Reasoning*. Yves Bertot was a reviewer for *Journal of Automated Reasoning*.

9.1.4. Invited Talks

Cyril Cohen gave an invited talk on formalizing robotics in January in Nijmegen, the Netherlands.

Cyril Cohen gave an invited talk on asymptotic reasoning in June in Pittsburgh, USA.

Cyril Cohen gave an invited talk at the workshop *Lean User Group* in November in Freiburg, Germany.

Benjamin Grégoire gave an invited talk at the "journées nationales du GDR sécurité" (national days of the CNRS research group on security) in May in Paris, France.

Benjamin Grégoire gave an invited tutorial at the CHES conference (Cryptographic Hardware and Embedded Systems) in September in Amsterdam, the Netherlands.

Enrico Tassi gave a four-hour tutorial at the EUTypes Summer School in August in Ohrid, Macedonia (https://sites.google.com/view/2018eutypesschool/home)

Enrico Tassi gave an invited talk at the ML workshop in September in Saint Louis, Missouri, USA on "ELPI: an extension language with binders and unification variables".

Yves Bertot gave an invited talk at the ICTAC conference in October in Stellenbosch, South Africa on "Formal Verification of a Geometry Algorithm: A Quest for Abstract Views and Symmetry in Coq Proofs". He also gave a half-day tutorial on Coq.

9.1.5. Leadership within the Scientific Community

We organized two one-week courses on the Coq system, both tagged as entry-level, on Coq and Coq and the Mathematical Components library.

9.1.6. Scientific Expertise

Yves Bertot was part of the review committee for the French *Haut Commissariat pour l'Évaluation de la Recherche et de l'Enseignement Supérieur* for the CNRS laboratory SAMOVAR in Evry, France.

9.1.7. Research Administration

- José Grimm is a member of the local committee for hygiene and work safety.
- Yves Bertot was a member of the "Bureau du comité des projets" until June.
- Benjamin Grégoire is a member of the committee on computer tool usage (CUMI) for the Sophia-Antipolis Méditerranée Inria center.
- Laurence Rideau was a member of the hiring committee for researchers in Sophia Antipolis.

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Doctorat: Enrico Tassi organized an advanced school on Coq and the Mathematical Components library, where Laurence Rideau, Cyril Cohen, Laurent Théry, and Yves Bertot gave lectures and supervised laboratory sessions. This school took place in December and had about 20 attendants.

Licence: Sophie Bernard gave 54 hours of lectures on probabilities at University of Nice Sophia Antipolis.

Licence: Damien Rouhling taught about 60 hours at University Nice Sophia Antipolis: differential calculus, Fourier analysis, and C programming (First year students).

Master: Yves Bertot organized a school on Coq in January, Boris Djalal and Damien Rouhling supervised the lab sessions.

Master: Laurent Théry taught 3 hours on "introduction to computer verified proof" at Ecole des Mines de Paris,

Licence: Boris Djalal taught 4 hours of computer science for first year students in a "classe préparatoire aux grandes écoles".

Licence: Cécile Baritel-Ruet taught 30 hours of computer science for first year students at Université de Nice, and some lectures on computer science history.

Licence: Cyril Cohen prepares students for oral examination in a "classe préparatoire aux grandes écoles".

9.2.2. Supervision

- Yves Bertot and Cyril Cohen supervised Boris Djalal, whose doctoral thesis was defended on December 3rd.
- Yves Bertot and Cyril Cohen supervise the doctoral thesis of Damien rouhling.
- Yves Bertot and Laurence Rideau supervise the doctoral thesis of Sophie Bernard.
- Yves Bertot and Benjamin Grégoire supervise the doctoral thesis of Cécile Baritel-Ruet.

9.2.3. Juries

Enrico Tassi was a member of the Thesis jury for Andrea Gabrielli, in October at the University of Florence, Italy.

Yves Bertot was a member of the Thesis jury for Guillaume Davy, in December at the University of Toulouse and the Institut Supérieur d'Aéronautique et de l'Espace, France.

9.3. Popularization

9.3.1. Interventions

Cyril Cohen presented the work of the Marelle team at a presentation for students coming from Mediterranean regions: Meddays.

10. Bibliography

Major publications by the team in recent years

[1] G. Barthe, B. Grégoire, S. Heraud, S. Z. Béguelin. *Computer-Aided Security Proofs for the Working Cryptographer*, in "Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings", Lecture Notes in Computer Science, Springer, 2011, vol. 6841, pp. 71-90, Best Paper Award

[2] Y. Bertot, P. Castéran. *Interactive Theorem Proving and Program Development, Coq'Art:the Calculus of Inductive Constructions*, Springer-Verlag, 2004

[3] Y. Bertot, G. Gonthier, S. O. Biha, I. Paşca. *Canonical Big Operators*, in "Proceedings of the 21st International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2008)", Lecture Notes in Computer Science, Springer, August 2008, vol. 5170, pp. 12–16, http://hal.inria.fr/inria-00331193/

[4] G. Gonthier, A. Asperti, J. Avigad, Y. Bertot, C. Cohen, F. Garillot, S. Le Roux, A. Mahboubi, R. O'Connor, S. Ould Biha, I. Paşca, L. Rideau, A. Solovyev, E. Tassi, L. Théry. *A Machine-Checked Proof of the Odd Order Theorem*, in "ITP 2013, 4th Conference on Interactive Theorem Proving", Rennes, France, S. Blazy, C. Paulin, D. Pichardie (editors), LNCS, Springer, 2013, vol. 7998, pp. 163-179 [*DOI :* 10.1007/978-3-642-39634-2_14], http://hal.inria.fr/hal-00816699

[5] G. Gonthier, A. Mahboubi, L. Rideau, E. Tassi, L. Théry. *A Modular Formalisation of Finite Group Theory*, in "Proceedings of the 20th International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2007)", K. Schneider, J. Brandt (editors), LNCS, Springer-Verlag, September 2007, vol. 4732, pp. 86-101, http://hal.inria.fr/inria-00139131

Publications of the year

Articles in International Peer-Reviewed Journals

[6] R. Affeldt, C. Cohen, D. Rouhling. *Formalization Techniques for Asymptotic Reasoning in Classical Analysis*, in "Journal of Formalized Reasoning", October 2018, https://hal.inria.fr/hal-01719918

[7] B. Ahrens, R. Matthes, A. Mörtberg. *From signatures to monads in UniMath*, in "Journal of Automated Reasoning", July 2018, pp. 1-34 [*DOI :* 10.1007/s10817-018-9474-4], https://hal.inria.fr/hal-01410487

[8] G. Barthe, T. Espitau, B. Grégoire, J. Hsu, P.-Y. Strub. *Proving expected sensitivity of probabilistic programs*, in "Proceedings of the ACM on Programming Languages", January 2018, vol. 2, n^{o} POPL, pp. 1-29 [*DOI :* 10.1145/3158145], https://hal.archives-ouvertes.fr/hal-01959322

[9] B. Grégoire, G. Bonnet, F. Pedraza. *Mechanisms of formation of slurry aluminide coatings from Al and Cr microparticles*, in "Surface and Coatings Technology", February 2019, vol. 359, pp. 323-333, https://hal.archives-ouvertes.fr/hal-01980488

Invited Conferences

[10] Y. Bertot. *Formal Verification of a Geometry Algorithm: A Quest for Abstract Views and Symmetry in Coq Proofs*, in "ICTAC 2018 - International Colloquium on Theoretical of Computing", Stellenbosch, South Africa, October 2018, https://arxiv.org/abs/1809.00559 , https://hal.inria.fr/hal-01866271

International Conferences with Proceedings

[11] A. Anand, S. Boulier, C. Cohen, M. Sozeau, N. Tabareau. *Towards Certified Meta-Programming with Typed Template-Coq*, in "ITP 2018 - 9th Conference on Interactive Theorem Proving", Oxford, United Kingdom, LNCS, Springer, July 2018, vol. 10895, pp. 20-39 [*DOI :* 10.1007/978-3-319-94821-8_2], https://hal.archives-ouvertes.fr/hal-01809681

[12] C. Baritel-Ruet, F. Dupressoir, P.-A. Fouque, B. Grégoire. *Formal Security Proof of CMAC and Its Variants*, in "CSF 2018 - 31st EEE Computer Security Foundations Symposium", Oxford, United Kingdom, July 2018, https://hal.archives-ouvertes.fr/hal-01959554

[13] G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, B. Grégoire, M. Rossi, M. Tibouchi. *Masking the GLP Lattice-Based Signature Scheme at Any Order*, in "Eurocrypt 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques", Tel Aviv, Israel, J. B. Nielse, V. Rijme (editors), Lecture Notes in Computer Science, Springer, April 2018, vol. 10821, pp. 354-384 [*DOI :* 10.1007/978-3-319-78375-8_12], https://hal.inria.fr/hal-01900708

[14] G. Barthe, T. Espitau, M. Gaboardi, B. Grégoire, J. Hsu, P.-Y. Strub. *An Assertion-Based Program Logic for Probabilistic Programs*, in "Programming Languages and Systems - 27th European Symposium on Programming, ESOP 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings", Thessaloniki, Greece, Programming Languages and Systems - 27th European Symposium on Programming, ESOP 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings, April 2018, pp. 117-144, https://hal.archives-ouvertes.fr/hal-01959567

[15] G. Barthe, X. Fan, J. Gancher, B. Grégoire, C. Jacomme, E. Shi. *Symbolic Proofs for Lattice-Based Cryptography*, in "CCS 2018 - Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security Canada, October 15-19, 2018", Toronto, Canada, ACM Press, October 2018, vol. 17, pp. 538-555 [*DOI :* 10.1145/3243734.3243825], https://hal.archives-ouvertes.fr/hal-01959391

[16] *Best Paper*
G. Barthe, B. Grégoire, V. Laporte. *Secure Compilation of Side-Channel Countermeasures: The Case of Cryptographic "Constant-Time"*, in "CSF 2018 - 31st IEEE Computer Security Foundations Symposium", Oxford, United Kingdom, July 2018, https://hal.archives-ouvertes.fr/hal-01959560.

[17] B. Djalal. *A Constructive Formalisation of Semi-algebraic Sets and Functions*, in "Certified Programs and Proofs", Los Angeles, California, United States, J. Andronick, A. Felty (editors), January 2018, https://hal.inria.fr/hal-01643919

[18] B. Grégoire, K. Papagiannopoulos, P. Schwabe, K. Stoffelen. *Vectorizing Higher-Order Masking*, in "COSADE 2018 - Constructive Side-Channel Analysis and Secure Design - 9th International Workshop", Singapore, Singapore, April 2018, pp. 23-43, https://hal.archives-ouvertes.fr/hal-01959418

[19] D. Rouhling. *A Formal Proof in Coq of a Control Function for the Inverted Pendulum*, in "CPP 2018 - 7th ACM SIGPLAN International Conference on Certified Programs and Proofs", Los Angeles, United States, January 2018, pp. 1-14 [*DOI :* 10.1145/3167101], https://hal.inria.fr/hal-01639819

Research Reports

[20] J. Grimm. *Implementation of Bourbaki's Elements of Mathematics in Coq: Part Two; Ordered Sets, Cardinals, Integers*, Inria Sophia Antipolis ; Inria, 2018, n^{o} RR-7150, 826 p. , https://hal.inria.fr/inria-00440786

Other Publications

[21] Y. Bertot. *Formal study in Coq of pi computations using arithmetic-geometric means*, April 2018, https://archive.softwareheritage.org/swh:1:rev:b1e197c030e66d588987087a193fc3a88d8bd5ed, Software, https://hal.inria.fr/hal-01767263

[22] R. Chen, C. Cohen, J.-J. Levy, S. Merz, L. Théry. *Formal Proofs of Tarjan's Algorithm in Why3, Coq, and Isabelle*, October 2018, https://arxiv.org/abs/1810.11979 - working paper or preprint, https://hal.inria.fr/hal-01906155

[23] T. Coq Development Team. *The Coq Proof Assistant, version 8.8.0*, April 2018, Software [*DOI :* 10.5281/zenodo.1219885], https://hal.inria.fr/hal-01954564

[24] F. GUIDI, C. SACERDOTI COEN, E. TASSI. *Implementing Type Theory in Higher Order Constraint Logic Programming*, November 2018, working paper or preprint, https://hal.inria.fr/hal-01410567

[25] E. TASSI. *Deriving proved equality tests in Coq-elpi (Stronger induction principles for containers in Coq)*, October 2018, working paper or preprint, https://hal.inria.fr/hal-01897468

[26] E. TASSI. *Elpi: an extension language for Coq (Metaprogramming Coq in the Elpi λProlog dialect)*, January 2018, working paper or preprint, https://hal.inria.fr/hal-01637063

References in notes

[27] M. JOLDES, V. POPESCU, J.-M. MULLER. *Tight and rigourous error bounds for basic building blocks of double-word arithmetic*, July 2016, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01351529