Activity Report 2017

Team CELTIQUE

Software Certification with Semantic Analysis

Joint team with Inria Rennes – Bretagne Atlantique

D4 – Language and Software Engineering

Table of contents

Project-Team CELTIQUE

Creation of the Project-Team: 2009 July 01

Keywords:

Computer Science and Digital Science:

A2.1. - Programming Languages
A2.1.1. - Semantics of programming languages
A2.1.2. - Object-oriented programming
A2.1.3. - Functional programming
A2.1.9. - Dynamic languages
A2.2. - Compilation
A2.2.1. - Static analysis
A2.2.2. - Memory models
A2.4. - Verification, reliability, certification
A2.4.1. - Analysis
A2.4.2. - Model-checking
A2.4.3. - Proofs
A4. - Security and privacy
A4.5. - Formal methods for security

Other Research Topics and Application Domains:

B6.1. - Software industry
B6.1.1. - Software engineering
B6.6. - Embedded systems

1. Personnel

Research Scientists

Thomas Jensen [Team leader, Inria, Senior Researcher, HDR]
Frédéric Besson [Inria, Researcher]
Alan Schmitt [Inria, Senior Researcher, HDR]

Faculty Members

Sandrine Blazy [Univ de Rennes I, Professor, HDR]
David Cachera [Ecole normale supérieure de Rennes, Associate Professor, HDR]
Delphine Demange [Univ de Rennes I, Associate Professor]
Thomas Genet [Univ de Rennes I, Associate Professor, HDR]
Serguei Lenglet [Univ de Lorraine, Associate Professor]
David Pichardie [Ecole normale supérieure de Rennes, Professor, HDR]

PhD Students

Pauline Bolignano [Prove & Run, until Feb 2017]
Gurvan Cabon [Inria]
Alexandre Dang [Inria]
Yon Fernandez de Retana [Univ de Rennes I]
Timothée Haudebourg [Univ de Rennes I, from Oct 2017]
Julien Lepiller [Inria]

Florent Saudel [Amossys]
Alix Trieu [Univ de Rennes I]
Yannick Zakowski [Ecole normale supérieure de Rennes, until Nov 2017]

Technical staff

Yannick Zakowski [Univ de Rennes I, from Dec 2017]

Interns

Kevin Le Bon [Inria, from May 2017 until Aug 2017]
David Reboullet [Ecole Normale Supérieure Paris, from Jun 2017 until Aug 2017]
Lionel Zoubritzky [Ecole Normale Supérieure Paris, from Jun 2017 until Jul 2017]

Administrative Assistant

Lydie Mabil [Inria]

Visiting Scientist

Ahmad Salim Al-Sibahi [IT University of Copenhagen, until Jan 2017]

2. Overall Objectives

2.1. Project overview

The overall goal of the CELTIQUE project is to improve the security and reliability of software with semantics-based modeling, analysis and certification techniques. To achieve this goal, the project conducts work on improving semantic description and analysis techniques, as well as work on using proof assistants (most notably Coq) to develop and prove properties of these techniques. We are applying such techniques to a variety of source languages, including Java, C, and JavaScript. We also study how these techniques apply to low-level languages, and how they can be combined with certified compilation. The CompCert certified compiler and its intermediate representations are used for much of our work on semantic modeling and analysis of C and lower-level representations.

The semantic analyses extract approximate but sound descriptions of software behaviour from which a proof of safety or security can be constructed. The analyses of interest include numerical data flow analysis, control flow analysis for higher-order languages, alias and points-to analysis for heap structure manipulation. In particular, we have designed several analyses for information flow control, aimed at computing attacker knowledge and detecting side channels.

We work with three application domains: Java software for small devices (in particular smart cards and mobile telephones), embedded C programs, and web applications.

CELTIQUE is a joint project with the CNRS, the University of Rennes 1 and ENS Rennes.

3. New Software and Platforms

3.1. Jacal

JAvaCard AnaLyseur
KEYWORDS: JavaCard - Certification - Static program analysis - AFSCM
FUNCTIONAL DESCRIPTION: Jacal is a JAvaCard AnaLyseur developed on top of the SAWJA platform. This proprietary software verifies automatically that Javacard programs conform with the security guidelines issued by the AFSCM (Association Française du Sans Contact Mobile). Jacal is based on the theory of abstract interpretation and combines several object-oriented and numeric analyses to automatically infer sophisticated invariants about the program behaviour. The result of the analysis is thereafter harvest to check that it is sufficient to ensure the desired security properties.

- Participants: David Pichardie, Delphine Demange, Frédéric Besson and Thomas Jensen
- Contact: Thomas Jensen

3.2. Javalib

FUNCTIONAL DESCRIPTION: Javalib is an efficient library to parse Java .class files into OCaml data structures, thus enabling the OCaml programmer to extract information from class files, to manipulate and to generate valid .class files.

- Participants: David Pichardie, Frédéric Besson, Laurent Guillo, Laurent Hubert, Nicolas Barré, Pierre Vittet and Tiphaine Turpin
- Contact: Frédéric Besson
- URL: http://sawja.inria.fr/

3.3. JSCert

Certified JavaScript
FUNCTIONAL DESCRIPTION: The JSCert project aims to really understand JavaScript. JSCert itself is a mechanised specification of JavaScript, written in the Coq proof assistant, which closely follows the ECMAScript 5 English standard. JSRef is a reference interpreter for JavaScript in OCaml , which has been proved correct with respect to JSCert and tested with the Test 262 test suite.

- Participants: Alan Schmitt and Martin Bodin
- Partner: Imperial College London
- Contact: Alan Schmitt
- URL: http://jscert.org/

3.4. SAWJA

Static Analysis Workshop for Java
KEYWORDS: Security - Software - Code review - Smart card
SCIENTIFIC DESCRIPTION: Sawja is a library written in OCaml, relying on Javalib to provide a high level representation of Java bytecode programs. It name comes from Static Analysis Workshop for JAva. Whereas Javalib is dedicated to isolated classes, Sawja handles bytecode programs with their class hierarchy and with control flow algorithms.

Moreover, Sawja provides some stackless intermediate representations of code, called JBir and A3Bir. The transformation algorithm, common to these representations, has been formalized and proved to be semantics-preserving.

See also the web page http://sawja.inria.fr/ .

Version: 1.5

Programming language: Ocaml
FUNCTIONAL DESCRIPTION: Sawja is a toolbox for developing static analysis of Java code in bytecode format. Sawja provides advanced algorithms for reconstructing high-level programme representations. The SawjaCard tool dedicated to JavaCard is based on the Sawja infrastructure and automatically validates the security guidelines issued by AFSCM (http://www.afscm.org/). SawjaCard can automate the code audit process and automatic verification of functional properties.

- Participants: David Pichardie, Frédéric Besson and Laurent Guillo
- Partners: CNRS - ENS Cachan
- Contact: Frédéric Besson
- URL: http://sawja.inria.fr/

3.5. Timbuk

KEYWORDS: Demonstration - Ocaml - Vérification de programmes - Tree Automata

FUNCTIONAL DESCRIPTION: Timbuk is a collection of tools for achieving proofs of reachability over Term Rewriting Systems and for manipulating Tree Automata (bottom-up non-deterministic finite tree automata)

RELEASE FUNCTIONAL DESCRIPTION: This version does no longer include the tree automata library but focuses on reachability analysis and equational approximations.

- Participant: Thomas Genet
- Contact: Thomas Genet
- URL: http://www.irisa.fr/celtique/genet/timbuk/

4. New Results

4.1. Higher-Order Process Calculi

Participants: Sergueï Lenglet, Alan Schmitt.

Sergueï Lenglet and Alan Schmitt, in collaboration with researchers at Wrocław university, designed a fully abstract encoding of the λ-calculus into HOcore, a minimal higher-order process calculus. This work has been published at LICS [37]. In parallel, Lenglet and Schmitt have formalized HOπ in Coq and showed that its bisimilarity is compatible using Howe's method. This work has been accepted for publication at CPP 2018 [30].

4.2. Certified Semantics and Analyses for JavaScript

Participants: Gurvan Cabon, Alan Schmitt.

Alan Schmitt has continued his collaboration with Arthur Charguéraud (Inria Nancy) and Thomas Wood (Imperial College London) to develop JSExplain, an interpreter for JavaScript that is as close as possible to the specification. The tool is publicly available at https://github.com/jscert/jsexplain and is being extended to cover the current version of the standard.

In parallel, Gurvan Cabon and Alan Schmitt have developed a framework to automatically derive an information-flow tracking semantics from a pretty-big-step semantics. This work has been published [34] and is being formalized in Coq.

4.3. Certified Concurrent Garbage Collector

Participants: Yannick Zakowski, David Cachera, Delphine Demange, David Pichardie.

Concurrent garbage collection algorithms are an emblematic challenge in the area of concurrent program verification. We addressed this problem by proposing a mechanized proof methodology based on the popular Rely-Guarantee (RG) proof technique. We designed a specific compiler intermediate representation (IR) with strong type guarantees, dedicated support for abstract concurrent data structures, and high-level iterators on runtime internals (objects, roots, fields, thread identifiers...). In addition, we defined an RG program logic supporting an incremental proof methodology where annotations and invariants can be progressively enriched. We have formalized the IR, the proof system, and proved the soundness of the methodology in the Coq proof assistant. Equipped with this IR, we have proved the correctness of a fully concurrent garbage collector where mutators never have to wait for the collector. This work has been published in [32].

In this work, reasoning simultaneously about the garbage collection algorithm and the concrete implementation of the concurrent data-structures it uses would have entailed an undesired and unnecessary complexity. The above proof is therefore conducted with respect to abstract operations which execute atomically. In practice, however, concurrent data-structures uses fine-grained concurrency, for performance reasons. One must therefore prove an observational refinement between the abstract concurrent data-structures and their fined-grained, "linearisable" implementation. To adress this issue, we introduce a methodology inspired by the work of Vafeiadis, and provide the approach with solid semantic foundations. Assuming that fine-grained implementations are proved correct with respect to an RG specification encompassing linearization conditions, we prove, once and for all, that this entails a semantic refinement of their abstraction. This methodology is instantiated to prove correct the main data-structure used in our garbage collector. This work has been published in [33].

4.4. Static analysis of functional programs using tree automata and term rewriting

Participants: Thomas Genet, Thomas Jensen, Timothée Haudebourg.

We develop a specific theory and the related tools for analyzing programs whose semantics is defined using term rewriting systems. The analysis principle is based on regular approximations of infinite sets of terms reachable by rewriting. Regular tree languages are (possibly) infinite languages which can be finitely represented using tree automata. To over-approximate sets of reachable terms, the tools we develop use the Tree Automata Completion (TAC) algorithm to compute a tree automaton recognizing a superset of all reachable terms. This over-approximation is then used to prove properties on the program by showing that some "bad" terms, encoding dangerous or problematic configurations, are not in the superset and thus not reachable. This is a specific form of, so-called, Regular Tree Model Checking. We have already shown that tree automata completion can safely over-approximate the image of any first-order complete and terminating functional program. We have extended this result to the case of higher-order functional programs [40] and obtained very encouraging experimental results http://people.irisa.fr/Thomas.Genet/timbuk/funExperiments/. Besides, we have shown that completion was abble to take the evaluation strategy of the program into account [19]. The next step is to show the completeness of the approach, i.e., that any regular approximation of the image of a function can be found using completion. We already made progress in this direction [39].

4.5. C Semantics and Certified Compilation

Participants: Frédéric Besson, Sandrine Blazy.

The COMPCERT C compiler provides the formal guarantee that the observable behaviour of the compiled code improves on the observable behaviour of the source code. A first limitation of this guarantee is that if the source code goes wrong, i.e. does not have a well-defined behaviour, any compiled code is compliant. Another limitation is that COMPCERT 's notion of observable behaviour is restricted to IO events.

Over the past years, we have refined the semantics underlying COMPCERT so that (unlike COMPCERT but like GCC) the binary representation of pointers can be manipulated much like integers and such that memory is a finite resource. We have now a formally verified C compiler, COMPCERTS, which is essentially the COMPCERT compiler, albeit with a stronger formal guarantee. The semantics preservation theorem applies to a wider class of existing C programs and, therefore, their compiled version benefits from the formal guarantee of COMPCERTS. COMPCERTS preserves not only the observable behaviour of programs but also ensures that the memory consumption is preserved by the compiler. As a result, we have the formal guarantee that the compiled code requires no more memory than the source code. This ensures that the absence of stack-overflows is preserved by compilation.

The whole proof of COMPCERTS represents a significant proof-effort. Details about the formal definition of the semantics and the proof of compiler passes can be found in the following publications [17], [25]

4.6. Constant-time verification by compilation and static analysis

Participants: Sandrine Blazy, David Pichardie, Alix Trieu.

To protect their implementations, cryptographers follow a very strict programming discipline called constant-time programming. They avoid branchings controlled by secret data as an attacker could use timing attacks, which are a broad class of side-channel attacks that measure different execution times of a program in order to infer some of its secret values. Several real-world secure C libraries such as NaCl, mbedTLS, or Open Quantum Safe, follow this discipline. We propose an advanced static analysis, based on state-of-the-art techniques from abstract interpretation, to report time leakage during programming. To that purpose, we analyze source C programs and use full context-sensitive and arithmetic-aware alias analyses to track the tainted flows. We give semantic evidences of the correctness of our approach on a core language. We also present a prototype implementation for C programs that is based on the CompCert compiler toolchain and its companion Verasco static analyzer. We present verification results on various real-world constant-time programs and report on a successful verification of a challenging SHA-256 implementation that was out of scope of previous tool-assisted approaches. This work has been published at ESORICS'17 [27].

The previous technique is well-adapted to verify the constant-time discipline at source level and give feedback to programmers, but the final security property must be established on the executable form of the program. In a joint work with IMDEA Software (Gilles Barthe and Vincent Laporte), we propose an automated methodology for validating on low-level intermediate representations the results of a source-level static analysis. Our methodology relies on two main ingredients: a relative-safety checker, an instance of a relational verifier which proves that a program is *safer* than another, and a transformation of programs into defensive form which verifies the analysis results at runtime. We prove the soundness of the methodology, and provide a formally verified instantiation based on the Verasco verified C static analyzer and the CompCert verified C compiler. This work has been published at CSF'17 [24].

5. Partnerships and Cooperations

5.1. National Initiatives

5.1.1. The ANR AnaStaSec project

Participants: Frédéric Besson, Sandrine Blazy, Thomas Jensen, Alexandre Dang, Julien Lepiller.

Static program analysis, Security, Secure compilation

The AnaStaSec project (2015–2018) aims at ensuring security properties of embedded critical systems using static analysis and security enhancing compiler techniques. The case studies are airborne embedded software with ground communication capabilities. The Celtique project focuses on software fault isolation which is a compiler technology to ensure by construction a strong segregation of tasks.

This is a joint project with the Inria teams ANTIQUE and PROSECCO, CEA-LIST, TrustInSoft, AMOSSYS and Airbus Group.

5.1.2. The ANR Binsec project

Participants: Frédéric Besson, Sandrine Blazy, Pierre Wilke, Julien Lepiller.

Binary code, Static program analysis

The Binsec project (2013–2017) is funded by the call ISN 2012, a program of the Agence Nationale de la Recherche. The goal of the BINSEC project is to develop static analysis techniques and tools for performing automatic security analyses of binary code. We target two main applicative domains: vulnerability analysis and virus detection.

Binsec is a joint project with the Inria CARTE team, CEA LIS, VERIMAG and EADS IW.

5.1.3. The ANR MALTHY project

Participant: David Cachera.

The MALTHY project, funded by ANR in the program INS 2013, aims at advancing the state-of-the-art in real-time and hybrid model checking by applying advanced methods and tools from linear algebra and algebraic geometry. MALTHY is coordinated by VERIMAG, involving CEA-LIST, Inria Rennes (Tamis and Celtique), Inria Saclay (MAXPLUS) and VISEO/Object Direct.

5.1.4. The ANR AJACS project

Participants: Martin Bodin, Gurvan Cabon, Thomas Jensen, Alan Schmitt.

The goal of the AJACS project is to provide strong security and privacy guarantees on the client side for web application scripts. To this end, we propose to define a mechanized semantics of the full JavaScript language, the most widely used language for the Web. We then propose to develop and prove correct analyses for JavaScript programs, in particular information flow analyses that guarantee no secret information is leaked to malicious parties. The definition of sub-languages of JavaScript, with certified compilation techniques targeting them, will allow us to derive more precise analyses. Finally, we propose to design and certify security and privacy enforcement mechanisms for web applications, including the APIs used to program real-world applications.

The project partners include the following Inria teams: Celtique, Indes, Prosecco, and Toccata; it also involves researchers from Imperial College as external collaborators. The project runs from December 2014 to November 2018.

5.1.5. The ANR DISCOVER project

Participants: Sandrine Blazy, Delphine Demange, Thomas Jensen, David Pichardie, Yon Fernandez de Retana, Yannick Zakovski.

The DISCOVER project project aims at leveraging recent foundational work on formal verification and proof assistants to design, implement and verify compilation techniques used for high-level concurrent and managed programming languages. The ultimate goal of DISCOVER is to devise new formalisms and proof techniques able to scale to the mechanized correctness proof of a compiler involving a rich class of optimizations, leading to efficient and scalable applications, written in higher-level languages than those currently handled by cutting-edge verified compilers.

In the light of recent work in optimizations techniques used in production compilers of high-level languages, control-flow-graph based intermediate representations seems too rigid. Indeed, the analyses and optimizations in these compilers work on more abstract representations, where programs are represented with data and control dependencies. The most representative representation is the sea-of-nodes form, used in the Java Hotspot Server Compiler, and which is the rationale behind the highly relaxed definition of the Java memory model. DISCOVER proposes to tackle the problem of verified compilation for shared-memory concurrency with a resolute language-based approach, and to investigate the formalization of adequate program intermediate representations and associated correctness proof techniques.

The project runs from October 2014 to September 2019.

5.2. European Initiatives

5.2.1. Collaborations in European Programs, Except FP7 & H2020

Program:CA COST Action CA15123

Project acronym: EUTYPES

Project title: European research network on types for programming and verification

Duration: 03/2016 to 03/2020

Coordinator: Herman Geuvers (Radboud University Nijmegen, The Netherlands)

Other partners: Austria, Belgium, Czech Republic, Denmark, Estonia, Finland, France, Macedonia, Germany, Hungary, Israel, Italy, Lithuania, Netherlands, Norway, Poland, Portugal, Romania, Serbia, Slovenia, Spain, Sweden, United Kingdom

Abstract: Types are pervasive in programming and information technology. A type defines a formal interface between software components, allowing the automatic verification of their connections, and greatly enhancing the robustness and reliability of computations and communications. In rich dependent type theories, the full functional specification of a program can be expressed as a type. Type systems have rapidly evolved over the past years, becoming more sophisticated, capturing new aspects of the behaviour of programs and the dynamics of their execution.

This COST Action will give a strong impetus to research on type theory and its many applications in computer science, by promoting (1) the synergy between theoretical computer scientists, logicians and mathematicians to develop new foundations for type theory, for example as based on the recent development of "homotopy type theory”, (2) the joint development of type theoretic tools as proof assistants and integrated programming environments, (3) the study of dependent types for programming and its deployment in software development, (4) the study of dependent types for verification and its deployment in software analysis and verification. The action will also tie together these different areas and promote cross-fertilisation.

Sandrine Blazy is Substitute Member of the Managment Committee for France.

5.3. International Initiatives

5.3.1. Inria International Partners

5.3.1.1. Declared Inria International Partners

WEBCERT

Title: Verified Trustworthy web Applications

International Partner (Institution - Laboratory - Researcher):

Imperial College London - Department of Computing - Philippa Gardner

Duration: 2015 - 2019

Start year: 2015

See also: JSCert web page

The WebCert partnership focuses on applying formal methods to the JavaScript language: mechanized specification, development of an executable formal specification, design of a program logic, development of verification tools, and study of secure sub-languages.

6. Dissemination

6.1. Promoting Scientific Activities

6.1.1. Scientific Events Selection

6.1.1.1. Chair of Conference Program Committees

- CoqPL 2017 (International Workshop on Coq for PL) was chaired by Sandrine Blazy and Emilio Jesus Gallego Arias

6.1.1.2. Member of the Conference Program Committees

- TASE 2017 (Symposium on Theoretical Aspects of Software Engineering): Alan Schmitt
- Web Programming 2018: Alan Schmitt
- ProWeb 2018: Alan Schmitt
- CC 2017 (Conference on Compiler Construction) : David Pichardie
- ESORICS 2017 (European Symposium on Research in Computer Security) : David Pichardie
- ESOP 2017 (European Symposium on Programming) : David Pichardie
- CC 2018 (Conference on Compiler Construction) : David Pichardie
- CoqPL 2017 (International Workshop on Coq for PL) : Sandrine Blazy
- AFADL 2017 (Approches Formelles dans l'Assistance au Développement de Logiciels) : Sandrine Blazy
- SRC (Student Research Competition) @ PLDI 2017 : Sandrine Blazy
- VSTTE 2017 (Verified Software: Theories, Tools, and Experiments) : Sandrine Blazy
- GPCE 2017 (Generative Programming: Concepts & Experiences) : Sandrine Blazy
- IFL 2017 (International symposium on Implementation and application of Functional Languages) : Sandrine Blazy
- TFP 2017 (Trends in Functional Programming) : Sandrine Blazy
- CPP 2018 (ACM SIGPLAN Conference on Certified Programs and Proofs) : Sandrine Blazy
- Euro S&P 2018 (IEEE European Symposium on Security and Privacy) : Sandrine Blazy
- TACAS 2017 (Tools and Algorithms for the Construction and Analysis of Software : Thomas Jensen.

- FCS 2017 (Int. workshop on Foundations of Computer Security) : Thomas Jensen.
- SAS 2017 (Static Analysis Symposium) : Thomas Jensen.

6.1.1.3. Reviewer

- POPL 2018 (Symposium on Principles of Programming Languages): Alan Schmitt

6.1.2. Journal

6.1.2.1. Reviewer - Reviewing Activities

- Information & Computation: Alan Schmitt
- Science of Computer Programming: Alan Schmitt
- Discrete Mathematics & Theoretical Computer: Alan Schmitt
- Theoretical Computer Science: Alan Schmitt
- Journal of Logical and Algebraic Methods in Programming: Alan Schmitt
- ACM Transactions on Privacy and Security (TOPS): David Pichardie

6.1.3. Invited Talks

- Delphine Demange: "On-the-Fly Garbage Collection: An Exercise in Compiler Verification". Inria Scientific Days 2017. June 2017.
- Thomas Genet: "SPAN+AVISPA for Verifying Cryptographic Protocols". RESSI (Rendez-vous de la recherche et de l'enseignement de la sécurité des systèmes d'information), Grenoble, May 2017 [42].
- Thomas Genet: "Tree Automata for Reachability in Rewriting". International School on Rewriting, Eindhoven, July 2017. http://www.win.tue.nl/~hzantema/isr.html.
- Thomas Jensen: Formal methods for software security, Forum Méthodes Formelles, Toulouse, France, Jan. 2017 [21].
- Thomas Jensen: Formal methods for software security, Journée inaugurale GDR Sécurité Informatique, Paris, June 2017 [22].
- Thomas Jensen. Hybrid information flow analysis against web tracking.. The 12th International Conference on Risks and Security of Internet and Systems (CRiSIS 2017), Dinard, France, Sept. 2017 [23].

6.1.4. Scientific Expertise

- Sandrine Blazy: expertise of an ERC Advanced Grant research proposal.
- Thomas Jensen is Inria representative in the European Cyber Security Organisation (ECSO) working group in Research and Innovation.

6.1.5. Research Administration

- Sandrine Blazy is member of Section 6 of the national committee for scientific research CoNRS.
- Sandrine Blazy is coordinator of the LTP (Languages, Types, Proofs) group of the French GDR GPL.
- Thomas Jensen is head of the NUMERIC department at Université Bretagne Loire.
- Thomas Jensen is director of the IT Security track and member of the executive board of the Laboratoire d'excellence "CominLabs".

6.2. Teaching - Supervision - Juries

6.2.1. Teaching

Licence : Alan Schmitt, Programmation Fonctionnelle, 36h, L3, Insa Rennes, France

Licence : Delphine Demange, Spécialité Informatique 1 - Algorithmique et Complexité Expérimentale, 36h, L1, Université Rennes 1, France

Licence : Delphine Demange, Spécialité Informatique 2 - Functional and Immutable Programming, 70h, L1, Université Rennes 1, France

Licence : Delphine Demange, Programmation de Confiance, 36h, L3, Université Rennes 1, France

Licence : David Pichardie, Graph algorithms, 24h, L3, ESIR, France

Licence : Sandrine Blazy, Functional programming, 30h, L3, Université Rennes 1, France

Licence: Thomas Genet, Software Engineering, 58h, L2, Université de Rennes 1, France

Licence : Thomas Genet, Spécialité Informatique 1 - Algorithmic and Experimental Complexity, 42h, L1, Université Rennes 1, France

Master : Sandrine Blazy, Méthodes Formelles pour le développement de logiciels sûrs, 53h, M1, Université Rennes 1, France

Master : Alan Schmitt, Méthodes Formelles pour le développement de logiciels sûrs, 25h, M1, Université Rennes 1, France France

Master : Sandrine Blazy, Mechanized Semantics, 15h, M2, Université Rennes 1, France

Master : Sandrine Blazy, Semantics, 24h, M1, Université Rennes 1, France

Master : Sandrine Blazy, Software vulnerabilities, 20h, M2, Université Rennes 1, France

Master : Delphine Demange, Software Security, 9h, M2, Université Rennes 1, France

Master : David Cachera, Semantics, 24h, M1, Université Rennes 1, France

Master : David Cachera, Advanced Semantics, 20h, M2, Université Rennes 1, France

Master : Thomas Genet, Formal Design and Verification, 108h, M1, Université de Rennes 1, France.

Master : Thomas Jensen, Program Analysis and Software Security, 21h, M2, Université Rennes 1, France

6.2.2. Supervision

PhD in progress : Timothée Haudebourg, Lightweight Formal Verification for Functional Programs, 1st october 2017, Thomas Genet and Thomas Jensen

PhD in progress : Alexandre Dang, Security by compilation, 1st september 2016, Frédéric Besson and Thomas Jensen

PhD in progress : Julien Lepiller, Binary analysis for Isolation, 1st september 2016, Frédéric Besson and Thomas Jensen

PhD in progress : Gurvan Cabon, Analyse non locale certifiée en JavaScript grâce à une sémantique annotée, 1st september 2015, Alan Schmitt

PhD in progress : Florent Saudel, Vulnerability discovery, November 2015, Sandrine Blazy, Frédéric Besson and Cédric Berthion (Amossys)

PhD in progress : Alix Trieu, Formally verified compilation and static analysis, January 2016, Sandrine Blazy and David Pichardie

PhD in progress : Yon Fernandez De Retana, Verified Optimising Compiler for high-level languages, 1st september 2015, Delphine Demange and David Pichardie

David Bühler, Structuring an abstract interpreter through value and state abstractions, defended March 2017, Sandrine Blazy and Boris Yakobowski (CEA)

Yannick Zakowski, Verification of a Concurrent Garbage Collector, defended December 2017, David Pichardie and David Cachera.

Pauline Bolignano, Formal models and verification of memory management in a hypervisor, defended May 2017, Thomas Jensen and Vincent Siles (Prove & Run).

Oana Andreescu, Static analysis of functional programs with an application to the frame problem in deductive verification, May 2017, Thomas Jensen and Stéphane Lescuyer (Prove & Run).

6.2.3. Juries

- Alan Schmitt, jury member for the selection of Inria CR (researcher) candidates, March and April 2017, Inria, Rennes, France.
- Thomas Jensen, jury member for the selection of Inria CR (researcher) candidates, March and April 2017, Inria, Rennes, France.
- Sandrine Blazy, jury member for the selection of CNRS CR and DR (researchers) candidates, February and March 2017, CNRS, Paris, France.
- Sandrine Blazy, jury member for the selection of a professor at University of Copenhagen, May 2017, Copenhagen, Denmark.
- Sandrine Blazy, jury member (reviewer) for the PhD defense of Romain Aïssat, January 2017, Paris-Sud University
- Sandrine Blazy, jury member for the PhD defense of Oana Andreescu, May 2017, Université Rennes 1
- Sandrine Blazy, jury member for the PhD defense of Ninon Eyrolles, June 2017, Université Versailles Saint-Quentin
- Sandrine Blazy, jury member (reviewer) for the HDR defense of Alain Giorgetti, December 2017, Université de Franche-Comté
- Sandrine Blazy, jury member for the PhD defense of Jordy Ruiz, December 2017, Université de Toulouse
- Sandrine Blazy, jury member for the PhD defense of Pierre Lestringant, December 2017, Université Rennes 1.
- Sandrine Blazy, jury member of the GDR GPL PhD award committee.
- David Pichardie, external reviewer for the PhD defense of Hendra Gunadi, July 2017, Australian National University, Canberra, Australia.
- David Pichardie, Licenciate discussion leader for the PhD student Marco Vassena, Chalmers University of Technology, Gothenburg, Sweden.
- Delphine Demange, jury member of the Gilles Kahn PhD award committee, December 2017, Inria Paris
- Delphine Demange, jury member for the PhD defense of Pauline Bolignano, May 2017, Université Rennes 1
- Thomas Genet, jury member (reviewer) for the PhD defense of Vivien Pelletier, October 2017, Université d'Orléans, France.
- Thomas Jensen, jury member for the HdR defense of Charlotte Truchet, November 2017, Université de Nantes, France.
- Thomas Jensen, jury member (reviewer) for the PhD defense of Zeineb Zhioua, September 2017, Téleécom ParisTech, France.
- Thomas Jensen, jury member for the PhD defense of Deepak Subramanian, December 2017, CentraleSupélec, France.

6.3. Popularization

Article "JavaScript, un langage à la croissance organique", Alan Schmitt, blog Binaire Le Monde. http://binaire.blog.lemonde.fr/2017/05/12/javascript-un-langage-a-la-croissance-organique/

Article “L’assistant de preuve Coq”, Sandrine Blazy, Pierre Castéran, Hugo Herbelin, Techniques et Sciences de l’ingénieur, août 2017. https://www.techniques-ingenieur.fr/base-documentaire/technologies-de-l-information-th9/programmation-42304210/coq-assistant-de-preuve-h3310/

Talk “Bug, Virus, Intrusion, Pirates... So many threats and no defense? Yes... maths.”, Thomas Genet, given three times in high schools close to Rennes.

7. Bibliography

Major publications by the team in recent years

[1] G. BARTHE, D. DEMANGE, D. PICHARDIE. *Formal Verification of an SSA-based Middle-end for CompCert*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", 2014, 35 p. , https://hal.inria.fr/hal-01097677

[2] F. BESSON, N. BIELOVA, T. JENSEN. *Hybrid Information Flow Monitoring Against Web Tracking*, in "CSF - 2013 IEEE 26th Computer Security Foundations Symposium", New Orleans, United States, 2013 [*DOI :* 10.1109/CSF.2013.23], http://hal.inria.fr/hal-00924138

[3] F. BESSON, T. JENSEN, D. PICHARDIE. *Proof-Carrying Code from Certified Abstract Interpretation to Fixpoint Compression*, in "Theoretical Computer Science", 2006, vol. 364, n^{o} 3, pp. 273–291

[4] M. BODIN, A. CHARGUÉRAUD, D. FILARETTI, P. GARDNER, S. MAFFEIS, D. NAUDZIUNIENE, A. SCHMITT, G. SMITH. *A Trusted Mechanised JavaScript Specification*, in "POPL 2014 - 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", San Diego, United States, November 2013, http://hal.inria.fr/hal-00910135

[5] B. BOYER, T. GENET, T. JENSEN. *Certifying a Tree Automata Completion Checker*, in "4th International Joint Conference, IJCAR 2008", Lectures Notes in Computer Science, Springer-Verlag, 2008, vol. 5195, pp. 347–362

[6] D. CACHERA, T. JENSEN, A. JOBIN, F. KIRCHNER. *Inference of polynomial invariants for imperative programs: a farewell to Gro¨bner bases*, in "Science of Computer Programming", 2014, vol. 93, 21 p. [*DOI :* 10.1016/J.SCICO.2014.02.028], https://hal.inria.fr/hal-00932351

[7] D. CACHERA, T. JENSEN, D. PICHARDIE, V. RUSU. *Extracting a Data Flow Analyser in Constructive Logic*, in "Theoretical Computer Science", 2005, vol. 342, n^{o} 1, pp. 56–78

[8] D. DEMANGE, V. LAPORTE, L. ZHAO, D. PICHARDIE, S. JAGANNATHAN, J. VITEK. *Plan B: A Buffered Memory Model for Java*, in "Proc. of the 40th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2013", Rome, Italy, ACM, 2013, http://hal.inria.fr/hal-00924716

[9] T. GENET, V. RUSU. *Equational Approximations for Tree Automata Completion*, in "Journal of Symbolic Computation", 2010, vol. 45(5):574-597, May 2010, n^{o} 5, pp. 574-597

[10] L. HUBERT, T. JENSEN, V. MONFORT, D. PICHARDIE. *Enforcing Secure Object Initialization in Java*, in "15th European Symposium on Research in Computer Security (ESORICS)", Lecture Notes in Computer Science, Springer, 2010, vol. 6345, pp. 101-115

[11] J.-H. JOURDAN, V. LAPORTE, S. BLAZY, X. LEROY, D. PICHARDIE. *A formally-verified C static analyzer*, in "POPL 2015: 42nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", Mumbai, India, ACM, January 2015, pp. 247-259 [*DOI :* 10.1145/2676726.2676966], https://hal.inria.fr/hal-01078386

Publications of the year

Doctoral Dissertations and Habilitation Theses

[12] O. F. ANDREESCU. *Static Analysis of Functional Programs with an Application to the Frame Problem in Deductive Verification*, Rennes 1, May 2017, https://hal.inria.fr/tel-01677897

[13] O. F. ANDREESCU. *Static analysis of functional programs with an application to the frame problem in deductive verification*, Université Rennes 1, May 2017, https://tel.archives-ouvertes.fr/tel-01682503

[14] P. BOLIGNANO. *Formal models and verification of memory management in a hypervisor*, Université Rennes 1, May 2017, https://tel.archives-ouvertes.fr/tel-01637937

[15] D. BÜHLER. *Structuring an Abstract Interpreter through Value and State Abstractions:EVA, an Evolved Value Analysis for Frama-C*, Université de Rennes 1, March 2017, https://hal.archives-ouvertes.fr/tel-01664726

[16] Y. ZAKOWSKI. *Verification of a Concurrent Garbage Collector*, École Normale Supérieur de Rennes, December 2017, https://hal.inria.fr/tel-01680213

Articles in International Peer-Reviewed Journals

[17] F. BESSON, S. BLAZY, P. WILKE. *A Verified CompCert Front-End for a Memory Model Supporting Pointer Arithmetic and Uninitialised Data*, in "Journal of Automated Reasoning", 2017, pp. 1-48 [*DOI :* 10.1007/s10817-017-9439-z], https://hal.inria.fr/hal-01656895

[18] H. CIRSTEA, S. LENGLET, P.-E. MOREAU. *Faithful (Meta-)Encodings Of Programmable Strategies Into Term Rewriting Systems*, in "Logical Methods in Computer Science", November 2017, vol. 13, n^o^ 4, pp. 1-54, Long version of the corresponding RTA-TLCA 15 paper [*DOI :* 10.23638/LMCS-13(4:16)2017], https://hal.inria.fr/hal-01479030

[19] T. GENET, Y. SALMON. *Reachability Analysis of Innermost Rewriting - extended version*, in "Logical Methods in Computer Science", 2017, https://hal.inria.fr/hal-01532090

[20] F. HONSELL, L. LIQUORI, P. MAKSIMOVIC, I. SCAGNETTO. *LLFP : A Logical Framework for modeling External Evidence, Side Conditions, and Proof Irrelevance using Monads*, in "Logical Methods in Computer Science", February 2017, https://arxiv.org/abs/1702.07214 , https://hal.inria.fr/hal-01146059

Invited Conferences

[21] T. JENSEN. *Formal methods for software security (invited talk)*, in "FMF 2017 - Forum "Méthodes Formelles"", Toulouse, France, January 2017, pp. 1-61, https://hal.inria.fr/hal-01658549

[22] T. JENSEN. *Formal methods for software security (invited talk)*, in "Journées Nationales 2017 Pré-GDR Sécurité Informatique", Paris, France, June 2017, pp. 1-31, https://hal.inria.fr/hal-01658835

[23] T. JENSEN. *Hybrid information flow analysis against web tracking (invited talk)*, in "CRiSIS 2017 - 12th International Conference on Risks and Security of Internet and Systems", Dinard, France, September 2017, pp. 1-33, https://hal.inria.fr/hal-01658896

International Conferences with Proceedings

[24] G. BARTHE, S. BLAZY, V. LAPORTE, D. PICHARDIE, A. TRIEU. *Verified Translation Validation of Static Analyses*, in "Computer Security Foundations Symposium", Santa-Barbara, United States, 30th IEEE Computer Security Foundations Symposium, August 2017, https://hal.inria.fr/hal-01588422

[25] F. BESSON, S. BLAZY, P. WILKE. *CompCertS: A Memory-Aware Verified C Compiler using Pointer as Integer Semantics*, in "ITP 2017 - 8th International Conference on Interactive Theorem Proving", Brasilia, Brazil, ITP 2017: Interactive Theorem Proving, Springer, September 2017, vol. 10499, pp. 81-97 [*DOI :* 10.1007/978-3-319-66107-0_6], https://hal.inria.fr/hal-01656875

[26] M. BIERNACKA, D. BIERNACKI, S. LENGLET, P. POLESIUK, D. POUS, A. SCHMITT. *Fully Abstract Encodings of λ-Calculus in HOcore through Abstract Machines*, in "LICS 2017", Reykjavik, Iceland, Proceedings of LICS 2017, June 2017, To appear, https://hal.inria.fr/hal-01479035

[27] S. BLAZY, D. PICHARDIE, A. TRIEU. *Verifying Constant-Time Implementations by Abstract Interpretation*, in "European Symposium on Research in Computer Security", Oslo, Norway, 22nd European Symposium on Research in Computer Security, September 2017, https://hal.inria.fr/hal-01588444

[28] D. KÄSTNER, J. BARRHO, U. WÜNSCHE, M. SCHLICKLING, B. SCHOMMER, M. SCHMIDT, C. FERDINAND, X. LEROY, S. BLAZY. *CompCert: Practical Experience on Integrating and Qualifying a Formally Verified Optimizing Compiler*, in "ERTS2 2018 - Embedded Real Time Software and Systems", Toulouse, France, 3AF, SEE, SIE, January 2018, https://hal.inria.fr/hal-01643290

[29] D. KÄSTNER, X. LEROY, S. BLAZY, B. SCHOMMER, M. SCHMIDT, C. FERDINAND. *Closing the Gap – The Formally Verified Optimizing Compiler CompCert*, in "SSS'17: Safety-critical Systems Symposium 2017", Bristol, United Kingdom, Developments in System Safety Engineering: Proceedings of the Twenty-fifth Safety-critical Systems Symposium, CreateSpace, February 2017, pp. 163-180, https://hal.inria.fr/hal-01399482

[30] S. LENGLET, A. SCHMITT. *HOπ in Coq*, in "CPP 2018 - The 7th ACM SIGPLAN International Conference on Certified Programs and Proofs", Los Angeles, United States, January 2018, 14 p. [*DOI :* 10.1145/3167083], https://hal.inria.fr/hal-01614987

[31] M. LESLOUS, V. VIET TRIEM TONG, J.-F. LALANDE, T. GENET. *GPFinder: Tracking the Invisible in Android Malware*, in "12th International Conference on Malicious and Unwanted Software", Fajardo, Puerto Rico, IEEE Computer Society, October 2017, https://hal-centralesupelec.archives-ouvertes.fr/hal-01584989

[32] Y. ZAKOWSKI, D. CACHERA, D. DEMANGE, G. PETRI, D. PICHARDIE, S. JAGANNATHAN, J. VITEK. *Verifying a Concurrent Garbage Collector using a Rely-Guarantee Methodology*, in "ITP 2017 - 8th International Conference on Interactive Theorem Proving", Brasília, Brazil, Lecture Notes in Computer Science, Springer, September 2017, vol. 10499, pp. 496-513 [*DOI :* 10.1007/978-3-319-66107-0_31], https://hal.inria.fr/hal-01613389

[33] Y. ZAKOWSKI, D. CACHERA, D. DEMANGE, D. PICHARDIE. *Verified Compilation of Linearizable Data Structures: Mechanizing Rely Guarantee for Semantic Refinement*, in "SAC 2018 - The 33rd ACM/SIGAPP Symposium On Applied Computing", Pau, France, April 2018, pp. 1-10, https://hal.archives-ouvertes.fr/hal-01653620

National Conferences with Proceedings

[34] G. CABON, A. SCHMITT. *Non-Interference through Annotated Multisemantics*, in "28ièmes Journées Francophones des Langages Applicatifs", Gourette, France, January 2017, https://hal.archives-ouvertes.fr/hal-01503094

Conferences without Proceedings

[35] G. CABON, A. SCHMITT. *Annotated multisemantics to prove Non-Interference analyses*, in "PLAS 2017 - ACM SIGSAC Workshop on Programming Languages and Analysis for Security", Dallas, United States, PLAS '17 Proceedings of the 2017 Workshop on Programming Languages and Analysis for Security, ACM, October 2017, pp. 49-62 [*DOI :* 10.1145/3139337.3139344], https://hal.archives-ouvertes.fr/hal-01656404

[36] F. SAUDEL, S. BLAZY, F. BESSON. *Confusion de Type en C++: État de l'Art et Difficultés de Détection*, in "RESSI 2017 - Rendez-vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information", Grenoble/Autrans, France, May 2017, pp. 1-5, https://hal.inria.fr/hal-01656979

Research Reports

[37] M. BIERNACKA, D. BIERNACKI, S. LENGLET, P. POLESIUK, D. POUS, A. SCHMITT. *Fully Abstract Encodings of λ-Calculus in HOcore through Abstract Machines*, Inria, April 2017, n° RR-9052, https://hal.inria.fr/hal-01507625

[38] T. GENET. *A Short Isabelle/HOL Tutorial for the Functional Programmer*, IRISA, 2017, https://hal.inria.fr/hal-01208577

[39] T. GENET. *Automata Completion and Regularity Preservation*, IRISA, Inria Rennes, April 2017, https://hal.archives-ouvertes.fr/hal-01501744

[40] T. GENET, T. HAUDEBOURG, T. JENSEN. *Verifying Higher-Order Functions with Tree Automata: Extended Version*, Irisa, October 2017, pp. 1-20, https://hal.inria.fr/hal-01614380

[41] Y. ZAKOWSKI, D. CACHERA, D. DEMANGE, D. PICHARDIE. *Compilation of Linearizable Data Structures: A Mechanised RG Logic for Semantic Refinement*, ENS Rennes ; IRISA, Inria Rennes ; Université Rennes 1, June 2017, https://hal.archives-ouvertes.fr/hal-01538128

Other Publications

[42] T. GENET. *SPAN+AVISPA for Verifying Cryptographic Protocols*, 2017, This is a video tutorial to learn how to use SPAN+AVISPA to automatically check security properties on cryptographic protocols, https://hal.inria.fr/hal-01532086