Activity Report 2019

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
A2.1.3. - Object-oriented programming
A2.1.4. - Functional programming
A2.1.12. - Dynamic languages
A2.2. - Compilation
A2.2.1. - Static analysis
A2.2.2. - Memory models
A2.2.3. - Memory management
A2.2.5. - Run-time systems
A2.2.9. - Security by compilation
A2.4. - Formal method for 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
A7.2.2. - Automated Theorem Proving
A7.2.3. - Interactive Theorem Proving

Other Research Topics and Application Domains:

B6.1. - Software industry
B6.1.1. - Software engineering
B6.4. - Internet of things
B6.6. - Embedded systems
B9.10. - Privacy

1. Team, Visitors, External Collaborators

Research Scientists

Thomas Jensen [Team leader, Inria, Senior Researcher, HDR]
Frédéric Besson [Inria, Researcher]
Alan Schmitt [Inria, Senior Researcher, HDR]
Benoît Montagu [Inria, Starting Research position, from Nov 2019]

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]

David Pichardie [Ecole normale supérieure de Rennes, Professor, HDR]

Post-Doctoral Fellows

Stefania Dumbrava [Ecole normale supérieure de Rennes, Post-Doctoral Fellow, until Aug 2019]
Jean Christophe Lechenet [Ecole normale supérieure de Rennes, Post-Doctoral Fellow]
Thomas Rubiano [Univ de Rennes I, Post-Doctoral Fellow]

PhD Students

Guillaume Ambal [Univ de Rennes I, PhD Student, from Sep 2019]
Aurele Barriere [Ecole Normale Supérieure Rennes, PhD Student, from Sep 2019]
Alexandre Dang [Inria, PhD Student, until Sep 2019]
Samy Daoud [Orange Labs, PhD Student, granted by CIFRE]
Timothée Haudebourg [Univ de Rennes I, PhD Student]
Rémi Hutin [Ecole normale supérieure de Rennes, PhD Student]
Aurèle Barrière [Ecole normale supérieure de Rennes, PhD Student]
Solène Mirliaz [Ecole normale supérieure de Rennes, PhD Student]
Adam Khayam [Inria, PhD Student, from Jul 2019]
Julien Lepiller [Inria, PhD Student]
Louis Noizet [Univ de Rennes 1, PhD Student, from Oct 2019]

Technical staff

Nicolas Barré [Ecole normale supérieure de Rennes, Engineer]
Benoît Montagu [Inria, Engineer, from May 2019 until Oct 2019]
Samuel Risbourg [Inria, Engineer]

Interns and Apprentices

Odile Radet [Inria, from May 2019 until Jul 2019]
Vincent Rebiscoul [Inria, from Sep 2019]
Justine Sauvage [CNRS, from Jun 2019 until Jul 2019]

Visiting Scientist

Nathanael Courant [Ecole Normale Supérieure Paris, from Mar 2019 until Aug 2019]

External Collaborator

Mickael Delahaye [DGA, from Nov 2019]

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, 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. CompcertSSA

KEYWORDS: Optimizing compiler - Formal methods - Proof assistant - SSA

FUNCTIONAL DESCRIPTION: CompcertSSA is built on top of the Compcert verified C compiler, by adding a middle-end based on the SSA form (Static Single Assignment) : conversion to SSA, SSA-based optimizations, and destruction of SSA.

- Participants: Sandrine Blazy, Delphine Demange, Yon Fernandez de Retana, David Pichardie and Leo Stefanesco
- Contact: Delphine Demange
- Publications: Mechanizing conventional SSA for a verified destruction with coalescing - Validating Dominator Trees for a Fast, Verified Dominance Test - A Formally Verified SSA-based Middle-end : Static Single Assignment meets CompCert - Formal Verification of an SSA-based Middle-end for CompCert - Verifying Fast and Sparse SSA-based Optimizations in Coq.
- URL: http://compcertssa.gforge.inria.fr/

3.2. 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

3.3. Javalib

KEYWORDS: Library - Java - Ocaml

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: David Pichardie
- URL: http://sawja.inria.fr/

3.4. 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/

3.5. 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: David Pichardie
- URL: http://sawja.inria.fr/

3.6. Timbuk

KEYWORDS: Automated deduction - Ocaml - Program verification - Tree Automata - Term Rewriting Systems

FUNCTIONAL DESCRIPTION: Timbuk is a tool designed to compute or over-approximate sets of terms reachable by a given term rewriting system. The libray also provides an OCaml toplevel with all usual functions on Bottom-up Nondeterministic Tree Automata.

- Participant: Thomas Genet
- Contact: Thomas Genet
- URL: http://people.irisa.fr/Thomas.Genet/timbuk/index.html

3.7. jsexplain

JSExplain

KEYWORDS: JavaScript - Compilation - Standards - Debug - Interpreter

FUNCTIONAL DESCRIPTION: JSExplain is a reference interpreter for JavaScript that closely follows the specification and that produces execution traces. These traces may be interactively investigated in a browser, with an interface that displays not only the code and the state of the interpreter, but also the code and the state of the interpreted program. Conditional breakpoints may be expressed with respect to both the interpreter and the interpreted program. In that respect, JSExplain is a double-debugger for the specification of JavaScript.

- Partner: Imperial College London
- Contact: Alan Schmitt
- Publication: JSExplain: A Double Debugger for JavaScript
- URL: https://gitlab.inria.fr/star-explain/jsexplain

4. New Results

4.1. Compiling Sandboxes:Formally Verified Software Fault Isolation

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

Software Fault Isolation (SFI) is a security-enhancing pro- gram transformation for instrumenting an untrusted binary module so that it runs inside a dedicated isolated address space, called a sandbox. To ensure that the untrusted module cannot escape its sandbox, existing approaches such as Google's Native Client rely on a binary verifier to check that all memory accesses are within the sandbox. Instead of rely- ing on a posteriori verification, we design, implement and prove correct a program instrumentation phase as part of the formally verified compiler CompCert that enforces a sandboxing security property a priori. This eliminates the need for a binary verifier and, instead, leverages the soundness proof of the compiler to prove the security of the sandbox- ing transformation. The technical contributions are a novel sandboxing transformation that has a well-defined C semantics and which supports arbitrary function pointers, and a formally verified C compiler that im- plements SFI. Experiments show that our formally verified technique is a competitive way of implementing SFI [6].

4.2. Information-Flow Preservation in Compiler Optimisations

Participants: Frédéric Besson, Alexandre Dang, Thomas Jensen.

Correct compilers perform program transformations preserving input/output behaviours of programs. Though mandatory, correctness is not sufficient to prevent program optimisations from introducing information-flow leaks that would make the target program more vulnerable to side-channel at- tacks than the source program. To tackle this problem, we propose a notion of Information-Flow Preserving (IFP) program transformation which ensures that a target program is no more vulnerable to passive side-channel attacks than a source program. To protect against a wide range of attacks, we model an attacker who is granted arbitrary memory accesses for a pre-defined set of observation points. We have proposed a compositional proof principle for proving that a transformation is IFP. Using this principle, we show how a translation validation technique can be used to automatically verify and even close information-flow leaks introduced by standard compiler passes such as dead- store elimination and register allocation. The technique has been experimentally validated on the CompCert C compiler [7].

4.3. Formalization of Higher-Order Process Calculi

Participants: Guillaume Ambal, Alan Schmitt.

Guillaume Amabal and Alan Schmitt, in collaboration with Sergueï Lenglet, have continued exploring how to formalize HOπ in Coq, in particular how to deal with the different kinds of binders used in the calculus. We have extended our previous study that compared locally nameless, De Bruijn indices, and nominal binders with an approach based on higher-order abstract syntax. We have discovered that this approach is not as elegant as in other calculi. A journal version is submitted for publication. The Coq scripts can be found at http://passivation.gforge.inria.fr/hopi/.

4.4. Certified Semantics and Analyses for JavaScript

Participants: Samuel Risbourg, Alan Schmitt.

Alan Schmitt and Samuel Risbourg have continued 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. It was presented to the TC39 committee standardizing JavaScript in December to solicit feedback.

4.5. Skeletal Semantics

Participants: Guillaume Ambal, Nathanael Courant, Thomas Jensen, Adam Khayam, Louis Noizet, Vincent Rebiscoul, Alan Schmitt.

The work on skeletal semantics [5], a modular and formal way to describe semantics or programming languages, has intensified during 2019. We have continued to develop `necro`, a tool to manipulate skeletal semantics and generate interpreters in OCaml, mechanized semantics in Coq, and static analyzers. The code is available online (). Several interns and PhD students are also working on skeletal semantics.

Nathanaël Courant has designed a control-flow analyzer for languages written as skeletal semantics. This work is now extended by Vincent Rebiscoul to certify the analyzer.

Louis Noizet is studying the formalization in Coq of natural semantics from skeletal semantics. To this end, he extended the necro tool to automatically generate a Coq formalization. Louis is also very involved in the maintenance of necro.

Guillaume Ambal is studying the language features that can be captured using skeletal semantics, focusing on concurrency and distribution. In this setting, he is building an approach to automatically derive a small-step semantics form a big-step one.

Adam Khayam is writing a formal semantics of the Hop multitier language, an extension of JavaScript to write web applications. As a first step, he is writing a skeletal semantics of JavaScript to validate that our approach scale for complex and sizable semantics.

4.6. Static analyses for proofs of programs

Participants: Oana Andreescu, Thomas Jensen, Stéphane Lescuyer, Benoît Montagu.

Thomas Jensen together with three industrial research engineers Oana Andreescu, Stéphane Lescuyer, and Benoît Montagu, worked on the development of static analyses that help reduce the manual proof effort that is needed to formally verify programs.

They improved the correlation analysis that Oana Andreescu introduced in her Phd thesis, by designing a novel abstract domain. They verified in Coq its semantic properties, and evaluated their approach on an industrial micro-kernel developed at *Prove&Run*. They showed that the technique could reduce the proof burden by two thirds [1].

4.7. 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 in [4] as an extended version of [12].

5. Partnerships and Cooperations

5.1. National Initiatives

5.1.1. The ANR Scrypt project

Participants: Frédéric Besson, Sandrine Blazy, Thomas Jensen, David Pichardie, Alexandre Dang, Remi Hutin.

Security, Secure compilation

The Scrypt project (ANR-18-CE25-0014) aims at providing secure implementations of crypto-graphic primitives using formal methods and secure compilation techniques. One specific goal is to design secure compilers which preserve the security of the source code against side-channel attacks.

This is a joint project with the Inria team MARELLE, École Polytechnique and AMOSSYS.

5.1.2. 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.

5.1.3. The ANR AJACS project

Participants: 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 March 2019.

5.1.4. The ANR DISCOVER project

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

The DISCOVER project project (2014–09/2019) 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 started in October 2014 and ended on September 2019.

5.1.5. The ANR CISC project

Participants: Frédéric Besson, Thomas Jensen, Alan Schmitt.

The goal of the CISC project is to investigate multitier languages and compilers to build secure IoT applications with private communication. In particular, we aim at extending multitier platforms by a new orchestration language that we call Hiphop.js to synchronize internal and external activities of IoT applications as a whole. Our goal is to define language, semantics, attacker models, and policies for the IoT and investigate automatic implementation of privacy and security policies by multitier compilation of IoT applications. To guarantee such applications are correct, and in particular that the required security and privacy properties are achieved, we propose to certify them using the Coq proof assistant. We plan to implement the CISC results as extensions of the multitier language Hop.js (developed at Inria), based on the JavaScript language to maximize its impact. Using the new platform, we will carry out experimental studies on IoT security.

The project partners include the following Inria teams: Celtique, Collège de France, Indes, and Privatics. The project runs from April 2018 to March 2022.

5.2. European Initiatives

5.2.1. FP7 & H2020 Projects

5.2.1.1. The ERC VESTA project

Participants: David Pichardie, Sandrine Blazy, Nicolas Barré, Stefania Dumbrava, Jean-Christophe Léchenet, Rémi Hutin, Aurèle Barrière, Solène Mirliaz.

The VESTA project aims at proposing guidance and tool-support to the designers of static analysis, in order to build advanced but reliable static analysis tools. We focus on analyzing low-level softwares written in C, leveraging on the CompCert verified compiler. Verasco is a verified static analyser that analyses C programs and follows many of the advanced abstract interpretation techniques developped for Astrée. The outcome of the VESTA project will be a platform that help designing other verified advanced abstract interpreters like Verasco, without starting from a white page. We will apply this technique to develop security analyses for C programs. The platform will be open-source and will help the adoption of abstract interpretation techniques.

This a consolidator ERC awarded to David Pichardie for 5 years. The project started in September 2018.

5.2.1.2. The SPARTA cybersecurity competence network

Participants: Thomas Jensen, Frédéric Besson.

SPARTA is a novel Cybersecurity Competence Network, supported by the EU's H2020 program, with the objective to develop and implement top-tier research and innovation collaborative actions. Guided by concrete challenges forming an ambitious Cybersecurity Research & Innovation Roadmap, SPARTA will set up unique collaboration means, leading the way in building transformative capabilities and forming a world-leading Cybersecurity Competence Network across the EU. The SPARTA consortium assembles 44 actors from 14 EU Member States at the intersection of scientific excellence, technological innovation, and societal sciences in cybersecurity.

Celtique is coordinating the Inria participation in the SPARTA network. The team contributes to the programme on intelligent infrastructures with techniques for building security-enhanced systems code that respects strong information flow constraints. The team is also leading the elaboration of the SPARTA scientific roadmap, in collaboration with TU Munich.

5.2.2. 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. 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: Organisation

6.1.1.1. Member of the Conference Program Committees

- Frédéric Besson: PriSC 2019 Workshop on Principles of Secure Compilation
- Sandrine Blazy: POPL 2020 (Symposium on Principles of Programming Languages), APLAS 2019 (Asian Symposium on Programming Languages and Systems), ITP 2019 (Conference on Interactive Theorem Proving), SpiSA 2019 (Workshop on Instruction Set Architecture Specification).
- Thomas Jensen: SAS 2019 (Static Analysis Symposium), ENTROPY 2019 (Enabling Trust through OS proofs).

6.1.1.2. Reviewer

- Frédéric Besson: TACAS

6.1.2. Journal

6.1.2.1. Reviewer - Reviewing Activities

- Frédéric Besson: Journal of Computer Security (JCS), Transactions on Software Engineering (TSE)
- Alan Schmitt: Theoretical Informatics and Applications (RAIRO ITA)

6.1.3. Invited Talks

- Frédéric Besson: GT Sécurité des Systèmes, des Logiciels et des Réseaux
- Sandrine Blazy: *The Verasco static analyzer*, Effective Verification: Static Analysis Meets Program Logics, Lorentz center, Netherlands, May. 2019
- Sandrine Blazy: *Formal Verification of a Constant-Time Preserving C Compiler*, Verified software workshop, Newton institute, Cambridge, United Kingdom, Sept. 2019
- Alan Schmitt: *Sémantiques Formelles et Certifiées*, 30 ans des JFLA
- Alan Schmitt: *Formal JavaScript*, Journées Nationales du GDR GPL
- Alan Schmitt: *Skeletal Semantics*, GT Langages, Types et Preuves

6.1.4. Leadership within the Scientific Community

- Sandrine Blazy is member of Section 6 of the national committee for scientific research CoNRS.
- Sandrine Blazy is member of IFIP WG 2.11 on program generation and of IFIP WG 1.9/2.15 on verified software.

6.1.5. Scientific Expertise

- Sandrine Blazy evaluated two Belgian FWO Grant research proposals.
- Sandrine Blazy is member of the Scientific Advisory Board of the Flemish Cybersecurity Initiative Flanders Program.
- Sandrine Blazy and David Pichardie were members of the HCERES evaluation committee of the LSV laboratory.
- Thomas Jensen was member of the HCERES evaluation committee of the University Grenoble–Alpes LIG laboratory.
- Thomas Jensen was vice-president of the ANR 2019 selection committee for research proposals in Computing and Communication (CES 25).

6.1.6. Research Administration

- Sandrine Blazy and Thomas Jensen are members of the steering committee of the international Static Analysis Symposium.

6.2. Teaching - Supervision - Juries

6.2.1. Teaching

License : Frédéric Besson, Functional programming, 28h, Insa3, Insa Rennes, France

Licence : Sandrine Blazy, Programmation de confiance, 81h, L3, Université Rennes 1, France

Master : Sandrine Blazy, Semantics of Programming Languages, 20h, M1, Université Rennes 1, France

Master : Sandrine Blazy, Software vulnerabilities, 30h, M2, Université Rennes 1, 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 : Thomas Genet, Spécialité Informatique 1 - Algorithmique et Complexité Expérimentale, 47h, L1, Université Rennes 1, France

Licence : Thomas Genet, Initiation au génie logiciel, 67h, L2, Université Rennes 1, France

Licence : Alan Schmitt, Programmation de Confiance, 30h, L3, Université Rennes 1, France

Licence : David Pichardie, Programmation, 30h, L3, ENS Rennes, France

Master : David Pichardie, Semantics of Programming Languages, 30h, M1, Université Rennes 1, France

Master : David Pichardie, Static analysis, 30h, M1, Université Rennes 1, France

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

Master : Thomas Genet, Enseigner l'informatique au lycée, 15h, M1, Université Rennes 1, France

Master : Thomas Genet, Analyse et conception formelle, 65h, M1, Université Rennes 1, France

Master : Benoît Montagu, Analyse et Conception Formelles, 16h, M1, Université Rennes 1, France

Master : Alan Schmitt, Suivi de Projet, 24h, M1, ENS Rennes, France

Master : Alan Schmitt, Advanced Semantics, 15h, M2, ENS Rennes, France

Master : Alan Schmitt, Preparation of Agregation exam, 20, M2, ENS Rennes, France

License : Thomas Jensen, Logic, 30h, Insa3, Insa Rennes, France

Master : Thomas Jensen, Software Security , 18h, M2, Université Rennes 1, France

6.2.2. Supervision

Phd : Alexandre Dang, Compilation for memory protection, Université Rennes 1, 10/12/2019, Thomas Jensen, Frédéric Besson

Phd : Julien Lepiller, Verifying Software Fault Isolation, Université Rennes 1, 11/12/2019, Thomas Jensen, Frédéric Besson

PhD in progress : Timothée Haudebourg, Verification of Higher-Order Functional Programs using Tree Automata, September 2017, Thomas Genet and Thomas Jensen

PhD in progress : Rémi Hutin, A C compiler ensuring security properties, September 2018, Sandrine Blazy and David Pichardie

PhD in progress : Aurèle Barrière, Formal verification of a JIT compiler, September 2019, Sandrine Blazy and David Pichardie

PhD in progress : Samy Daoud, Modélisation formelle pour le déploiement et la reconfiguration automatique en ligne dans une infrastructure réseau et IT répartie, November 2018, Alan Schmitt

PhD in progress : Adam Khayam, Formal Semantics of Multitier Languages, July 2019, Alan Schmitt

PhD in progress : Guillaume Ambal, Sémantiques Squelettiques pour Calculs de Processus, September 2019, Alan Schmitt

PhD in progress : Louis Noizet, Compilation Certifiée de Sémantiques Squelettiques, October 2019, Alan Schmitt

Master 2 internship : Aurèle Barrière, Formal verification of a JIT compiler, February of June 2019, Sandrine Blazy and David Pichardie

Master 1 internship : Guillaume Barbier, Maxime Bridoux, Jean Jouve, Clément Legrand-Duchesne, Skeletal Semantics for WebAssembly, September 2018 to May 2019 Alan Schmitt

ENS 4th year internship : Nathanaël Courant, CFA Generation from a Skeletal Semantics, September 2019 to July 2019, Thomas Jensen and Alan Schmitt

ENS 4th year internship : Vincent Rebiscoul, Certified CFA Generation, September 2019 to February 2020, Thomas Jensen and Alan Schmitt

ENS 3th year internship : Justine Sauvage, Modèles formels pour la blockchain, May 2019 to July 2019, Thomas Genet and Thomas Jensen

6.2.3. Juries

- Sandrine Blazy, jury member for the selection of CNRS CR and DR (researchers) candidates, February and March 2019, CNRS, Paris, France.
- Sandrine Blazy, jury member (president) for the PhD defense of Simon Lunel, January 2019, Université Rennes 1
- Sandrine Blazy, jury member for the PhD defense of Joseph Lallemand, November 2019, Lorraine University, Nancy
- Sandrine Blazy, jury member for the HDR defense of Yann Régis-Gianas, November 2019, Paris University
- Sandrine Blazy, jury member (reviewer) for the PhD defense of Mathieu Journault, November 2019, Paris Sorbonne University
- Sandrine Blazy, jury member (president) for the PhD defense of Arif Ali Anapparakkal, December 2019, Université Rennes 1
- Sandrine Blazy, jury member (president) for the PhD defense of Julien Lepiller, December 2019, Université Rennes 1
- Sandrine Blazy, jury member of the GDR GPL PhD award committee
- Sandrine Blazy, member of hiring committee for a professor position, Spring 2019, UBO, Brest
- Thomas Genet, jury member (reviewer) for the PhD defense of Lily Gallois, December 2019, Université de Lille
- Alan Schmitt, president of jury for the PhD defense of Florian Dold, February 2019, Université Rennes 1
- Alan Schmitt, jury member (examiner) for the PhD defense of Julien Lopez, September 2019, Université Paris-Saclay
- Alan Schmitt, jury member (reviewer) for the PhD defense of Steven Varoumas, November 2019, Sorbonne Université
- Delphine Demange, jury member (examiner) for the PhD defense of Oscar Luis Vera Pérez, December 2019, Université Rennes 1
- Delphine Demange, member of hiring committee for a Maître de Conférence position, Spring 2019, Lyon 1 / LIP.

6.3. Popularization

6.3.1. Interventions

- Thomas Genet gave a talk “Bug, Virus, Intrusion, Pirates... So many threats and no defense? Yes... maths.” in high schools close to Rennes. 2019.

7. Bibliography

Publications of the year

Articles in International Peer-Reviewed Journals

[1] O. ANDREESCU, T. JENSEN, S. LESCUYER, B. MONTAGU. *Inferring frame conditions with static correlation analysis*, in "Proceedings of the ACM on Programming Languages", January 2019, vol. 3, n° POPL, pp. 1-29 [*DOI :* 10.1145/3290360], https://hal.inria.fr/hal-02413262

[2] 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", April 2019, vol. 62, n° 4, pp. 433-480 [*DOI :* 10.1007/s10817-017-9439-z], https://hal.inria.fr/hal-01656895

[3] F. BESSON, S. BLAZY, P. WILKE. *CompCertS: A Memory-Aware Verified C Compiler using a Pointer as Integer Semantics*, in "Journal of Automated Reasoning", August 2019, vol. 63, n° 2, pp. 369-392 [*DOI :* 10.1007/s10817-018-9496-y], https://hal.inria.fr/hal-02401182

[4] S. BLAZY, D. PICHARDIE, A. TRIEU. *Verifying constant-time implementations by abstract interpretation*, in "Journal of Computer Security", 2019, vol. 27, n° 1, pp. 137–163 [*DOI :* 10.3233/JCS-181136], https://hal.inria.fr/hal-02025047

[5] M. BODIN, P. GARDNER, T. JENSEN, A. SCHMITT. *Skeletal Semantics and their Interpretations*, in "Proceedings of the ACM on Programming Languages", 2019, vol. 44, pp. 1-31, forthcoming [*DOI :* 10.1145/3290357], https://hal.inria.fr/hal-01881863

International Conferences with Proceedings

[6] F. BESSON, S. BLAZY, A. DANG, T. JENSEN, P. WILKE. *Compiling Sandboxes: Formally Verified Software Fault Isolation*, in "ESOP 2019 - 28th European Symposium on Programming", Prague, Czech Republic, LNCS, Springer, April 2019, vol. 11423, pp. 499-524 [*DOI :* 10.1007/978-3-030-17184-1_18], https://hal.inria.fr/hal-02316189

[7] F. BESSON, A. DANG, T. JENSEN. *Information-Flow Preservation in Compiler Optimisations*, in "CSF 2019 - 32nd IEEE Computer Security Foundations Symposium", Hoboken, United States, IEEE, June 2019, pp. 1-13, https://hal.inria.fr/hal-02180303

[8] S. BLAZY. *Teaching Deductive Verification in Why3 to Undergraduate Students*, in "FM Tea (Formal Methods Teaching)", Porto, Portugal, September 2019, pp. 52-66 [*DOI :* 10.1007/978-3-030-32441-4_4], https://hal.inria.fr/hal-02362306

[9] S. BLAZY, R. HUTIN. *Formal Verification of a Program Obfuscation Based on Mixed Boolean-Arithmetic Expressions*, in "CPP 2019 - 8th ACM SIGPLAN International Conference on Certified Programs and Proofs", Cascais, Portugal, ACM, January 2019, pp. 196-208 [*DOI :* 10.1145/3293880.3294103], https://hal.inria.fr/hal-01955773

Conferences without Proceedings

[10] P. TALBOT, D. CACHERA, E. MONFROY, C. TRUCHET. *Octogones entiers pour le problème RCPSP*, in "JFPC 2019 - Journées Francophones de Programmation par Contraintes", Albi, France, June 2019, pp. 1-10, https://hal.archives-ouvertes.fr/hal-02157804

Research Reports

[11] S. MIRLIAZ, D. PICHARDIE. *Flow insensitive relational static analysis*, ENS Rennes ; Université Rennes 1, June 2019, https://hal.archives-ouvertes.fr/hal-02332139

References in notes

[12] 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