RESEARCH CENTRE

Rennes - Bretagne Atlantique

IN PARTNERSHIP WITH:

Université Rennes 1, École normale supérieure de Rennes

2020

ACTIVITY REPORT

Project-Team

CELTIQUE

Software certification with semantic analysis

IN COLLABORATION WITH: Institut de recherche en informatique et systèmes aléatoires (IRISA)

DOMAIN

Algorithmics, Programming, Software and Architecture

THEME

Proofs and Verification

Contents

Project-Team CELTIQUE

Creation of the Project-Team: 2009 July 01

Keywords

Computer sciences and digital sciences

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

A2.4.1. – Analysis

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 members, visitors, external collaborators

Research Scientists

- Thomas Jensen [Team leader, Inria, Senior Researcher, HDR]
- Frédéric Besson [Inria, Researcher]
- Simon Castellan [Inria, Researcher]
- Benoît Montagu [Inria, Starting Research Position]
- Thomas Rubiano [Inria, Starting Research Position, until Oct 2020]
- Alan Schmitt [Inria, Senior Researcher, HDR]

Faculty Members

- Sandrine Blazy [Univ de Rennes I, Professor, HDR]
- David Cachera [École 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 [École normale supérieure de Rennes, Professor, HDR]

Post-Doctoral Fellow

- Jean Christophe Lechenet [École normale supérieure de Rennes, until Aug 2020]

PhD Students

- Guillaume Ambal [Univ de Rennes I]
- Aurele Barriere [École normale supérieure de Rennes]
- Santiago Bautista [École normale supérieure de Rennes, from Sep 2020]
- Samy Daoud [Orange Labs, CIFRE, until Jun 2020]
- Timothée Haudebourg [Univ de Rennes I]
- Rémi Hutin [École normale supérieure de Rennes]
- Adam Khayam [Inria]
- Solene Mirliaz [École normale supérieure de Rennes]
- Louis Noizet [Univ de Rennes I]
- Gautier Raimondi [Inria, from Oct 2020]
- Vincent Rebiscoul [Univ de Rennes I, from Sep 2020]

Technical Staff

- Nicolas Barré [École normale supérieure de Rennes, Engineer, until Feb 2020]
- Samuel Risbourg [Inria, Engineer, until Aug 2020]

Interns and Apprentices

- Santiago Bautista [École normale supérieure de Rennes, from Feb 2020 until Jul 2020]
- Theo Gouzien [Inria, from May 2020 until Jul 2020]
- Alexandre Moine [Inria, from Jun 2020 until Jul 2020]
- Mathieu Poirier [Univ de Rennes I, from May 2020 until Jul 2020]
- Vincent Rebiscoul [Inria, until Jul 2020]

Administrative Assistant

- Stephanie Gosselin Lemaile [Inria]

External Collaborators

- Mickael Delahaye [DGA]
- Emmanuel Fleury [Univ de Bordeaux, from Oct 2020]

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.

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

3 Research program

The Celtique team conducts research in

- mechanised semantics of programming languages,
- semantics-based program analysis,
- certified compilation,
- language-based software security.

4 Application domains

We work with three application domains: Java software for small devices, embedded C programs, and web applications.

5 New software and platforms

5.1 New software

5.1.1 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.

URL: `https://compcertssa.gitlabpages.inria.fr/`

Publications: hal-01378393, hal-01193281, hal-01110783, hal-01097677, hal-01110779

Contacts: Delphine Demange, David Pichardie

Participants: Sandrine Blazy, Delphine Demange, Yon Fernandez de Retana, David Pichardie, Leo Stefanesco

5.1.2 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.

URL: `http://people.irisa.fr/Thomas.Genet/timbuk/index.html`

Contact: Thomas Genet

Participant: Thomas Genet

5.1.3 jsexplain

Name: 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.

URL: `https://gitlab.inria.fr/star-explain/jsexplain`

Publication: hal-01745792

Contact: Alan Schmitt

Partner: Imperial College London

5.1.4 JSCert

Name: Certified JavaScript

Keywords: JavaScript, Coq, Formalisation

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.

URL: http://jscert.org/

Contacts: Martin Bodin, Alan Schmitt

Participants: Alan Schmitt, Martin Bodin

Partner: Imperial College London

5.1.5 necro

Name: necro

Keywords: Semantics, Programming language, Specification language

Functional Description: The goal of the project is to provide a tool to manipulate skeletal semantics, a format to represent the semantics of programming languages.

URL: http://skeletons.inria.fr/necro.html

Contacts: Alan Schmitt, Louis Noizet

5.1.6 Causality

Keywords: Semantics, Interpreter, Concurrency, Programming language

Functional Description: Causality is library to write causal semantics of programming languages, based on a monad for truly concurrent computations. It comes with an implementation of an interpreter for a concurrent variant of OCaml.

Contact: Simon Castellan

5.1.7 itauto

Keyword: Automated theorem proving

Functional Description: itauto is a Coq reflexive tactic for solving intuitionistic propositional logic. The tactic inherits features found in modern SAT solvers: definitional conjunctive normal form, lazy unit propagation and conflict driven backjumping. It is also parametrised by a user-provided tactic that is called on the leaves of the proof search.

URL: https://gitlab.inria.fr/fbesson/itauto

Contact: Frédéric Besson

Participant: Frédéric Besson

6 New results

6.1 Formalization of Higher-Order Process Calculi

Participant 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. A journal version describing this work has been accepted for publication [8]. The Coq scripts can be found at http://passivation.inria.fr/hopi/.

In addition, in collaboration with Małgorzata Biernacka, Dariusz Biernacki, and Sergueï Lenglet, Alan Schmitt has designed an automatic translation of semantics written in structural style into non-deterministic abstract machines. This provides a way to formally define abstract machines for process calculi. This work has been submitted for publication.

6.2 JSExplain

Participant 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://gitlab.inria.fr/star-explain/jsexplain.

6.3 Skeletal Semantics

Participant Guillaume Ambal, Thomas Jensen, Adam Khayam, Louis Noizet, Vincent Rébiscoul, Thomas Rubiano, Alan Schmitt.

The work on skeletal semantics [6], a modular and formal way to describe semantics or programming languages, has intensified during 2020. 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 (https://gitlab.inria.fr/skeletons/necro). Several interns, PhD students, and postdocs are also working on skeletal semantics.

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. This work has been published [16]. Louis is also very involved in the evolution of necro, which now supports higher-order definitions and polymorphism.

Guillaume Ambal is studying the language features that can be captured using skeletal semantics, focusing on concurrency and distribution. In this setting, he has built an approach to automatically derive a small-step semantics form a big-step one, generating a fully automated Coq proof of their equivalence in the process. This work is submitted for publication.

Adam Khayam is writing a formal semantics of JavaScript to validate that our approach scale for complex and sizable semantics while remaining close to the specification and usable for formal proofs. A paper describing this work has been accepted at a peer-reviewed national conference and will be presented in 2021.

Thomas Rubiano has written a skeletal semantics of WebAssembly, a low-level language used for high-performance web applications. The semantics covers both the execution and the validation phases. It is used to generate an OCaml interpreter that passes the tests provided by the specification.

Vincent Rébiscoul has started his PhD working on static analyses for skeletal semantics. He is designing a framework that can automatically derive a control-flow analysis from the definition of a language as a skeletal semantics. The correctness of the analysis is automatically derived from the correctness of its components.

6.4 Relational analysis for higher-order languages

Participant Thomas Jensen, Benoît Montagu.

Devising a sound and precise static analysis of the λ-calculus is a difficult task. One major issue is due to the dynamic aspect of the control flow that is exhibited by higher-order programs. Standard approaches for control-flow analyses (CFA) compute an over-approximation of the closures that can be produced at the different call sites of a program. Such analyses are whole program analyses, that are neither relational nor modular.

Thomas Jensen and Benoît Montagu have developed a relational interpretation of the λ-calculus with sums and pairs. The interpretation can serve as a collecting semantics to develop relational, modular analyses for higher-order languages [11]. They proved in Coq that the interpretation is sound and complete, and defined abstractions that turn the collecting semantics into a computable analysis for the λ-calculus, using the abstract interpretation methodology. The precision of the obtained modular analysis is discussed. It favorably compares to existing CFAs on selected examples.

6.5 A new relational abstract domain for the static analysis of algebraic datatypes with numeric values

Participant Santiago Bautista, Thomas Jensen, Benoît Montagu.

As part of his MSc internship, Santiago Bautista, together with Thomas Jensen and Benoît Montagu, developed a new relational abstract domain [12]. It combines the expressive power of the correlation abstract domain [1] with the expressiveness of an arbitrary domain that expresses numeric relations. This combined expressiveness of the domains is suitable for analysing in a modular way programs written in high-level functional languages, where algebraic datatypes are used pervasively.

6.6 Constant-time verification by compilation

Participant Sandrine Blazy, David Pichardie, Rémi Hutin.

Timing side-channels are arguably one of the main sources of vulnerabilities in cryptographic implementations. One effective mitigation against timing side-channels is to write programs that do not perform secret-dependent branches and memory accesses. This mitigation, known as "cryptographic constant-time", is adopted by several popular cryptographic libraries.

This work focuses on compilation of cryptographic constant-time programs, and more specifically on the following question: is the code generated by a realistic compiler for a constant-time source program itself provably constant-time? Surprisingly, we answer the question positively for a mildly modified version of the CompCert compiler, a formally verified and moderately optimizing compiler for C. Concretely, we modify the CompCert compiler to eliminate sources of potential leakage. Then, we instrument the operational semantics of CompCert intermediate languages so as to be able to capture cryptographic constant-time. Finally, we prove that the modified CompCert compiler preserves constant-time. Our mechanization maximizes reuse of the CompCert correctness proof, through the use of new proof techniques for proving preservation of constant-time. These techniques achieve complementary trade-offs between generality and tractability of proof effort, and are of independent interest.

This work has been published in [9].

6.7 A Fast Verified Liveness Analysis in SSA Form

Participant Sandrine Blazy, David Pichardie, Jean-Christophe Léchenet.

Liveness analysis is a standard compiler analysis, enabling several optimizations such as deadcode elimination. The SSA form is a popular compiler intermediate language allowing for simple and fast optimizations. Boissinot et al. designed a fast liveness analysis by combining the specific properties of SSA with graph-theoretic ideas such as depth-first search and dominance. We formalize their approach in the Coq proof assistant, inside the CompCertSSA verified C compiler. We also compare experimentally this approach on CompCert's benchmarks with respect to the classic data-flow-based liveness analysis, and observe performance gains.

This work has been published in [15]

6.8 Proving Termination of Ethereum's Smart Contracts

Participant Thomas Genet, Thomas Jensen, Justine Sauvage.

We have developed a mechanized formal proof of termination of *any* Smart Contract in Ethereum. Ethereum is a decentralized blockchain technology equipped with so-called Smart Contracts. A contract is a program whose code is public, which can be triggered by any user, and whose actual execution is performed by miners participating in Ethereum. Miners execute the contract on the Ethereum Virtual Machine (EVM) and apply its effect by adding new blocks to the blockchain. A contract that takes too much time to be processed by the miners of the network may result into delays or a denial of service in the Ethereum system. To prevent this scenario, termination of Ethereum's Smart Contracts is ensured using a gas mechanism. Roughly, the EVM consumes gas to process each instruction of a contract and the gas provided to run a contract is limited. This technique could make termination of contracts easy to prove but the way the official definition of the EVM specifies gas usage makes the proof of this property non-trivial. EVM implementations and formal analysis techniques of EVM's Smart Contracts use termination of contracts as an assumption, so having a formal proof of termination of contracts is crucial. We have developed such proof in Isabelle/HOL. Furthermore the proof makes minimal assumption on the concrete gas costs for each operation. This is valuable because the gas cost has already changed several times during the EVM's lifetime and is likely to evolve again since gas pricing of operations is still not fully satisfactory. This has been published in [14] and the formalization is available at `http://people.irisa.fr/Thomas.Genet/EVM/`.

6.9 Proving Structural Properties on Programs using Regular Languages

Participant 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. Recently, we shown how to reason and abstract those informations using a dedicated type system associating regular language types to variables, expressions, etc. of a program. By automatically inferring such types we perform fully automatic verification of safety properties of tree-processing higher-order functional programs. We use term rewriting systems to model the program and its semantics and tree automata to model algebraic data types. We define the regular abstract interpretation of the input term rewriting system where the abstract domain is a set of regular languages. From the regular abstract interpretation we derive a type system where each type is a regular language. We define an inference procedure for this type system which allows us check the validity of safety properties. The inference mechanism is built on an invariant learning procedure based on the tree automata completion algorithm. This invariant learning procedure is regularly-complete and complete in refutation, meaning that if it is possible to give a regular type to a term then we will eventually find it, and if there is no possible type (regular or not) then we will eventually find a counter-example. This has been published in [10] and experiments are available http://people.irisa.fr/Thomas.Genet/timbuk/timbuk4/experiments.html.

6.10 Game Semantics: Easy as Pie

Participant Simon Castellan.

We provide a decomposition of causal game semantics models in two steps: (1) a syntactic translation into a dialect of the π-calculus and (2) a semantic interpretation of the processes in terms of event structures. We apply this methodology to a ML-like language extended with shared-memory concurrency, and provide the first non-angelic interactive model for such a language.

Our process calculus is inspired by the connection between session types and Linear Logic, that we extend to the setting of Differential Linear Logic. The new operators in Differential Logic are crucial to interpret shared-memory concurrency as they can be used to represent races between memory accesses on the same memory cell.

This is joint work with Léo Stefanesco (Collège de France) and Nobuko Yoshida (Imperial College London). We have submitted our results to the LICS2021 conference [19].

6.11 Formalization of Intermittent Systems

Participant Delphine Demange.

Transiently-powered systems featuring non-volatile memory as well as external peripherals enable the development of new low-power sensor applications. However, as programmers, we are ill-equipped to reason about systems where power failures are the norm rather than the exception. A first challenge consists in being able to capture all the volatile state of the application, external peripherals included, to ensure progress. A second, more fundamental, challenge consists in specifying how power failures may interact with peripheral operations.

We have proposed a formal specification of intermittent computing with peripherals and an axiomatic model of interrupt-based checkpointing, as well as its proof of correctness, machine-checked in the Coq proof assistant. Our model caputres several systems proposed in the literature. The correctness proof is structured with several intermediate refinement steps, and uses oracle semantics to deal with the non-determinism of failure scenarios (checkpointing and logging mechanisms).

This work has been accepted for publication at LCTES 2020 [13]. An extended version of this paper has also been submitted to TOPLAS, and is currently under review. In the extended version presents

This is joint work with Gautier Berthou (CITI / INSA Lyon), Pierre-Évariste Dagand (CNRS / LIP6), Rémi Oudin (CNRS / LIP6), and Tanguy Risset (CITI / INSA Lyon).

7 Partnerships and cooperations

7.1 International initiatives

7.1.1 Inria international partners

Informal international partners Alan Schmitt is collaborating with Małgorzata Biernacka and Dariusz Biernacki from the university of Wrocław, Poland, on formal transformations of operational semantics.

7.2 European initiatives

7.2.1 FP7 & H2020 Projects

SPARTA

Title: SPARTA Cybersecurity Competence Network

Coordinator: CEA

Partners:

- CENTRE D'EXCELLENCE EN TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION (Belgium)
- CESNET ZAJMOVE SDRUZENI PRAVNICKYCH OSOB (Czech Republic)
- COMMISSARIAT A L ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES (France)
- CONSIGLIO NAZIONALE DELLE RICERCHE (Italy)
- CONSORZIO INTERUNIVERSITARIO NAZIONALE PER L'INFORMATICA (Italy)
- CONSORZIO NAZIONALE INTERUNIVERSITARIO PER LE TELECOMUNICAZIONI (Italy)
- CZ.NIC, ZSPO (Czech Republic)
- DIREZIONE GENERALE PER LE TECNOLOGIE DELLE COMUNICAZIONI E LA SICUREZZA INFORMATICA - ISTITUTO SUPERIORE DELLE COMUNICAZIONI E DELLE TECNOLOGIE DELL'INFORMAZIONE (Italy)
- FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V. (Germany)
- FUNDACIO EURECAT (Spain)
- FUNDACION CENTRO DE TECNOLOGIAS DE INTERACCION VISUAL Y COMUNICACIONES VICOMTECH (Spain)
- FUNDACION TECNALIA RESEARCH & INNOVATION (Spain)
- GENEROLO JONO ZEMAICIO LIETUVOS KARO AKADEMIJA (Lithuania)
- INDRA SISTEMAS SA (Spain)
- INOV INESC INOVACAO - INSTITUTO DE NOVAS TECNOLOGIAS (Portugal)
- INSTITUT NATIONAL DES SCIENCES APPLIQUEES DE LYON (France)
- INSTITUTO SUPERIOR TECNICO (Portugal)
- ITTI SP ZOO (Poland)
- JOANNEUM RESEARCH FORSCHUNGSGESELLSCHAFT MBH (Austria)
- KAUNO TECHNOLOGIJOS UNIVERSITETAS (Lithuania)
- KENTRO MELETON ASFALEIAS (Greece)
- LEONARDO - SOCIETA PER AZIONI (Italy)
- LIETUVOS KIBERNETINIU NUSIKALTIMU KOMPETENCIJU IR TYRIMU CENTRAS (Lithuania)
- LUXEMBOURG INSTITUTE OF SCIENCE AND TECHNOLOGY (Luxembourg)
- MYKOLO ROMERIO UNIVERSITETAS (Lithuania)
- NATIONAL CENTER FOR SCIENTIFIC RESEARCH "DEMOKRITOS" (Greece)
- NAUKOWA I AKADEMICKA SIEC KOMPUTEROWA - PANSTWOWY INSTYTUT BADAWCZY (Poland)
- SECRETARIAT GENERAL DE LA DEFENSE ET DE LA SECURITE NATIONALE (France)
- STOWARZYSZENIE POLSKA PLATFORMA BEZPIECZENSTWA WEWNETRZNEGO (Poland)

- TARTU ULIKOOL (Estonia)
- TECHNIKON FORSCHUNGS- UND PLANUNGSGESELLSCHAFT MBH (Austria)
- TECHNISCHE UNIVERSITAET MUENCHEN (Germany)
- THALES SIX GTS FRANCE SAS (France)
- UNIVERSITAT KONSTANZ (Germany)
- UNIVERSITE DE NAMUR ASBL (Belgium)
- UNIVERSITE DU LUXEMBOURG (Luxembourg)
- VYSOKE UCENI TECHNICKE V BRNE (Czech Republic)

Inria contact: *Thomas Jensen*

Summary: SPARTA is a 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.

The ERC VESTA project

Participants	Sandrine Blazy, David Pichardie, Remi Hutin, Aurèle Barriere, Solène Mirliaz, Jean-Christophe Léchenet.

Keywords: Security, Secure compilation.

Title: ERC Consolidator grant VESTA

Contact person: David Pichardie

Summary: 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.

7.3 National initiatives

7.3.1 The ANR Scrypt project

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

Keywords: Security, Secure compilation.

The Scrypt project (ANR-18-CE25-0014) aims at providing secure implementations of cryptographic 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.

7.3.2 The ANR CISC project

Participant 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.

8 Dissemination

8.1 Promoting scientific activities

8.1.1 Scientific events: selection

Chair of conference program committees

- David Pichardie, co-chair of SAS 2020

Member of the conference program committees

- Delphine Demange, CC 2020
- Delphine Demange, JFLA 2020
- Delphine Demange, CoqPL 2020
- Alan Schmitt, POPL 2021
- Sandrine Blazy, Alan Schmitt, ESOP 2021
- Sandrine Blazy, POPL 2020, ICFP 2020 (ERC), PriSC 2020, TyDE 2020, GPCE 2020, AFADL 2020
- Benoît Montagu, NSAD 2020

8.1.2 Journal

Reviewer - reviewing activities

- Alan Schmitt, Science of Computer Programming
- Benoît Montagu, Journal of Computer Security
- Besson Frédéric, Software Tools for Technology Transfer

8.1.3 Invited talks

- Alan Schmitt, “Sémantique Formelle de JavaScript, Les Enjeux du Passage à l’Échelle”, Collège de France, January 2020
- Benoît Montagu, “Formal Verification of an Industrial μKernel: An Experience Report on the ProvenCore Project”, Inria Scientific Days on Proof Assistants, October 2020
- Sandrine Blazy, "From Verified Compilation to Secure Compilation: a Semantic Approach", ACM SIGSAC PLAS (Programming Languages and Analysis for Security) workshop, November 2020

8.1.4 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.

8.1.5 Scientific expertise

- Alan Schmitt, evaluation of ARED proposal
- Sandrine Blazy, evaluation of FNR proposal
- Sandrine Blazy was member of the HCERES evaluation committee of the LIP laboratory.

8.1.6 Research administration

- Thomas Jensen is director of the Labex CominLabs track on Trust, Security and Privacy.
- David Pichardie is co-chair of the EUR CyberSchool on cybersecurity (PIA3).

8.2 Teaching - Supervision - Juries

8.2.1 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 : Benoît Montagu, Programmation de Confiance, 20h, L3, 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, 24h, M1, Université Rennes 1, France
- Master : Alan Schmitt, Advanced Semantics, 30h, M2, ENS Rennes, France
- Master : Alan Schmitt, Preparation of Agregation exam, 20, M2, ENS Rennes, France
- License : Thomas Jensen, Logic, 20h, Insa3, Insa Rennes, France
- License : Thomas Jensen, Logic, 9h, Univeristy of Copenhagen, Denmark
- Master : Thomas Jensen, Software Security , 14h, M2, Université Rennes 1, France
- Master : Thomas Jensen, Topics in Programming Languages, 8h, Univeristy of Copenhagen, Denmark
- Master : Simon Castellan, Software Security , 6h, M2, Université Rennes 1, France

8.2.2 Supervision

- PhD : Timothée Haudebourg, Verification of Higher-Order Functional Programs using Tree Automata, defended December 2020, 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 : Solène Mirliaz, Provable Cost Analysis of Pipelined-Optimized Cryptographic Code, September 2019, David Pichardie
- 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
- PhD in progress : Vincent Rébiscoul, Analyses Statiques pour Sémantiques Squelettiques, September 2020, Thomas Jensen and Alan Schmitt
- PhD in progress : Santiago Bautista, Analyses statiques pour la vérification semi-interactive de programmes, September 2020, Thomas Jensen and Benoît Montagu
- PhD in progress : Gautier Raimondi, Secure Compilation against Side-channel attacks, September 2020, Frédéric Besson and Thomas Jensen
- Master 2 internship : Santiago Bautista, Relational numeric domains for static correlation analysis, February to June 2020, Thomas Jensen and Benoît Montagu

- ENS 4th year internship : Vincent Rebiscoul, Certified CFA Generation, September 2019 to July 2020, Thomas Jensen and Alan Schmitt
- Master 1 internship : Théo Gouzien, Monadic Gated SSA : formal semantics and proof of compiler optimization, May 2020 to July 2020, Delphine Demange

8.2.3 Juries

- Alan Schmitt, member of hiring committee for a professor, Spring 2020, Université Paris Saclay
- Sandrine Blazy, member of hiring committee for a professor, Spring 2020, ENS Paris
- Alan Schmitt, jury member (president) for the PhD defense of Adrien Durier, June 2020, Université de Lyon
- Alan Schmitt, jury member (reviewer) for the PhD defense of Mohammed Houssem Eddine Hachmaoui, September 2020, Université Paris Saclay
- Thomas Jensen, jury member (reviewer) for the PhD defense of Benjamin Farinier, U. Grenoble Alpes.
- Thomas Jensen, jury member (reviewer) for the PhD defense of Anthony Ferrand, U. Montpellier.
- Thomas Jensen, jury member (president) for the PhD defense of Nicolas Szlifierski, IMT Atlantique.
- Sandrine Blazy, jury member (reviewer) for the PhD defense of Cécile Baritel-Ruet, U. Côte d'Azur, 10/20.
- Sandrine Blazy, jury member for the PhD defense of Charlie Jacomme, ENS Paris Saclay, 10/20.
- Sandrine Blazy, jury member (reviewer) for the PhD defense of Darius Mercadier, Sorbonne U., 11/20.
- Sandrine Blazy, jury member of the GDR GPL PhD award committee
- Sandrine Blazy, jury member for the selection of CNRS CR and DR (researchers) candidates, Spring 2020, Paris.
- David Pichardie, jury member (president) for the PhD defense of Alexandre Debant, Université Rennes 1.
- David Pichardie, jury member (reviewer) for the PhD defense of Marc Chevalier, ENS Paris.

8.3 Popularization

8.3.1 Internal or external Inria responsibilities

Thomas Jensen is member of the Bureau du Comité des projets at Inria RBA.

8.3.2 Articles and contents

- Alan Schmitt, https://www.lemonde.fr/blog/binaire/2020/03/29/semantique-des-langages-de-programmation/

9 Scientific production

9.1 Major publications

[1] O. Andreescu, T. Jensen, S. Lescuyer and B. Montagu. 'Inferring frame conditions with static correlation analysis'. In: *Proceedings of the ACM on Programming Languages* 3.POPL (Jan. 2019), pp. 1–29. DOI: 10.1145/3290360. URL: https://hal.inria.fr/hal-02413262.

[2] G. Barthe, S. Blazy, B. Grégoire, R. Hutin, V. Laporte, D. Pichardie and A. Trieu. 'Formal verification of a constant-time preserving C compiler'. In: *Proceedings of the ACM on Programming Languages* 4.POPL (Jan. 2020), pp. 1–30. DOI: 10.1145/3371075. URL: https://hal.univ-lorraine.fr/hal-02975012.

[3] G. Barthe, D. Demange and D. Pichardie. 'Formal Verification of an SSA-based Middle-end for CompCert'. In: *ACM Transactions on Programming Languages and Systems (TOPLAS)* 36.1 (2014), p. 35. DOI: 10.1145/2579080. URL: https://hal.inria.fr/hal-01097677.

[4] F. Besson, S. Blazy, A. Dang, T. Jensen and P. Wilke. 'Compiling Sandboxes: Formally Verified Software Fault Isolation'. In: *ESOP 2019 - 28th European Symposium on Programming*. Vol. 11423. LNCS. Prague, Czech Republic: Springer, Apr. 2019, pp. 499–524. DOI: 10.1007/978-3-030-17184-1_18. URL: https://hal.inria.fr/hal-02316189.

[5] M. Bodin, A. Charguéraud, D. Filaretti, P. Gardner, S. Maffeis, D. Naudziuniene, A. Schmitt and G. Smith. 'A Trusted Mechanised JavaScript Specification'. In: *POPL 2014 - 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. San Diego, United States, Jan. 2014. URL: https://hal.inria.fr/hal-00910135.

[6] M. Bodin, P. Gardner, T. Jensen and A. Schmitt. 'Skeletal Semantics and their Interpretations'. In: *Proceedings of the ACM on Programming Languages* 44 (2019), pp. 1–31. DOI: 10.1145/3290357. URL: https://hal.inria.fr/hal-01881863.

[7] J.-H. Jourdan, V. Laporte, S. Blazy, X. Leroy and D. Pichardie. 'A formally-verified C static analyzer'. In: *POPL 2015: 42nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. Mumbai, India: ACM, Jan. 2015, pp. 247–259. DOI: 10.1145/2676726.2676966. URL: https://hal.inria.fr/hal-01078386.

9.2 Publications of the year

International journals

[8] G. Ambal, S. Lenglet and A. Schmitt. 'HOπ in Coq'. In: *Journal of Automated Reasoning* (2020). DOI: 10.1007/s10817-020-09553-0. URL: https://hal.inria.fr/hal-02536463.

[9] G. Barthe, S. Blazy, B. Grégoire, R. Hutin, V. Laporte, D. Pichardie and A. Trieu. 'Formal verification of a constant-time preserving C compiler'. In: *Proceedings of the ACM on Programming Languages* 4.POPL (Jan. 2020), pp. 1–30. DOI: 10.1145/3371075. URL: https://hal.univ-lorraine.fr/hal-02975012.

[10] T. Haudebourg, T. Genet and T. Jensen. 'Regular Language Type Inference with Term Rewriting - extended version'. In: *Proceedings of the ACM on Programming Languages*. International Conference on Functional Programming (ICFP) 4.ICFP (2020), pp. 1–29. DOI: 10.1145/3408994. URL: https://hal.inria.fr/hal-02795484.

[11] B. Montagu and T. Jensen. 'Stable relations and abstract interpretation of higher-order programs'. In: *Proceedings of the ACM on Programming Languages* 4.ICFP (2nd Aug. 2020), pp. 1–30. DOI: 10.1145/3409001. URL: https://hal.inria.fr/hal-02916996.

International peer-reviewed conferences

[12] S. Bautista, T. Jensen and B. Montagu. 'Numeric Domains Meet Algebraic Data Types'. In: 9th International Workshop on Numerical and Symbolic Abstract Domains (NSAD 2020). Virtual, United States, 17th Nov. 2020, pp. 12–16. DOI: 10.1145/3427762.3430178. URL: https://hal.inria.fr/hal-03028476.

[13] G. Berthou, P.-E. Dagand, D. Demange, R. Oudin and T. Risset. 'Intermittent Computing with Peripherals, Formally Verified'. In: *LCTES '20: 21st ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems Proceedings*. LCTES '20 - 21st ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems. London / Virtual, United Kingdom, June 2020, pp. 85–96. DOI: 10.1145/3372799.3394365. URL: https://hal.inria.fr/hal-02556878.

[14] T. Genet, T. Jensen and J. Sauvage. 'Termination of Ethereum's Smart Contracts'. In: SECRYPT 2020 - 17th International Conference on Security and Cryptography. Lieusaint - Paris / Virtual, France, 8th July 2020, pp. 39–51. DOI: 10.5220/0009564100390051. URL: https://hal.inria.fr/hal-03122008.

[15] J.-C. Léchenet, S. Blazy and D. Pichardie. 'A Fast Verified Liveness Analysis in SSA Form'. In: IJCAR 2020- International Joint Conference on Automated Reasoning. Paris, France, 24th June 2020, pp. 324–340. DOI: 10.1007/978-3-030-51054-1_19. URL: https://hal.inria.fr/hal-02904204.

National peer-reviewed Conferences

[16] L. Noizet and A. Schmitt. 'Formalisation de Sémantiques Squelettiques'. In: JLFA 2020 - Journées Francophones des Langages Applicatifs. Gruissan, France, 29th Jan. 2020, pp. 1–14. URL: https://hal.inria.fr/hal-02512485.

Reports & preprints

[17] G. Ambal, A. Schmitt and S. Lenglet. *Automatic Transformation of a Big-Step Skeletal Semantics into Small-Step*. Inria Rennes - Bretagne Atlantique, 23rd Sept. 2020. URL: https://hal.inria.fr/hal-02946930.

[18] S. Castellan and P. Clairambault. *Disentangling Parallelism and Interference in Game Semantics*. 26th Mar. 2021. URL: https://hal.archives-ouvertes.fr/hal-03182043.

[19] S. Castellan, L. Stefanesco and N. Yoshida. *Concurrent Game Semantics: Easy as Pi*. Inria, 2nd Feb. 2020. URL: https://hal.inria.fr/hal-03128187.

[20] T. Genet, T. Jensen and J. Sauvage. *Termination of Ethereum's Smart Contracts*. Univ Rennes, Inria, CNRS, IRISA, 27th Apr. 2020. URL: https://hal.inria.fr/hal-02555738.