Activity Report 2014

Project-Team PROSECCO

Programming securely with cryptography

RESEARCH CENTER
Paris - Rocquencourt

THEME
Security and Confidentiality

Table of contents

Project-Team PROSECCO

Keywords: Programming Languages, Security, Formal Methods, Cryptographic Protocols, Automated Verification

Creation of the Project-Team: 2012 July 01.

1. Members

Research Scientists

Karthikeyan Bhargavan [Team leader, Inria, HdR]
Bruno Blanchet [Inria, Senior Researcher, HdR]
Catalin Hritcu [Inria, Researcher]
Graham Steel [Inria, Researcher, HdR]

Engineers

Gergely Bana [Inria, Post-Doctoral Researcher, granted by FP7 PROSECO- ERC CRYSP project]
Romain Bardou [Inria, until Aug 2014, granted by OSEO]
Benjamin Smyth [Inria, Post-Doctoral Researcher, until Oct 2014, granted by FP7 PROSECO- ERC CRYSP project]
Santiago Zanella-Béguelin [Inria, MSR-Inria Post-Doctoral Researcher]

Post-Doctoral Fellow

Elizabeth Quaglia [Inria, Post-Doctoral Researcher, until Oct 2014]

Administrative Assistant

Anna Bednarik [Inria]

Others

Arthur Azevedo de Amorim [Inria, Summer Intern, from Mar 2014 until Aug 2014]
Kelsey Cairns [Inria, Summer Intern, from Mar 2014 until May 2014]
Nikolaos Giannarakis [Inria, Summer Intern, from Apr 2014 until Sep 2014]
Shubham Jindal [Inria, Summer Intern, from May 2014 until Jul 2014]
Zoi Paraskevopoulou [Inria, Summer Intern, from Apr 2014 until Sep 2014]
Susan Thomson [Inria, Summer Intern, from Jun 2014 until Aug 2014]
Evmorfia-Iro Bartzia [Inria, PhD Student, until Oct 2014]
Benjamin Beurdouche [Ecole Centrale de Paris, Summer Intern]
Antoine Delignat-Lavaud [Inria, PhD Student]
Chantal Keller [Inria, MSR-Inria Post-Doctoral Researcher]
Robert Künnemann [Inria, PhD Student, until Jan 2014]
Miriam Paiola [Inria, PhD Studenr, until May 2014]
Alfredo Pironti [Inria, Starting Research Position, until Dec 2014]
Jean-Karim Zinzindohoué [Min. Ecologie, PhD Student]

2. Overall Objectives

2.1. Programming securely with cryptography

In recent years, an increasing amount of sensitive data is being generated, manipulated, and accessed online, from bank accounts to health records. Both national security and individual privacy have come to rely on the security of web-based software applications. But even a single design flaw or implementation bug in an application may be exploited by a malicious criminal to steal, modify, or forge the private records of innocent users. Such *attacks* are becoming increasingly common and now affect millions of users every year.

The risks of deploying insecure software are too great to tolerate anything less than mathematical proof, but applications have become too large for security experts to examine by hand, and automated verification tools do not scale. Today, there is not a single widely-used web application for which we can give a proof of security, even against a small class of attacks. In fact, design and implementation flaws are still found in widely-distributed and thoroughly-vetted security libraries designed and implemented by experts.

Software security is in crisis. A focused research effort is needed if security programming and analysis techniques are to keep up with the rapid development and deployment of security-critical distributed applications based on new cryptographic protocols and secure hardware devices. The goal of our team PROSECCO is to draw upon our expertise in cryptographic protocols and program verification to make decisive contributions in this direction.

Our vision is that, over its lifetime, PROSECCO will contribute to making the use of formal techniques when programming with cryptography as natural as the use of software debugger. To this end, our long-term goals are to design and implement programming language abstractions, cryptographic models, verification tools, and verified security libraries that developers can use to deploy provably secure distributed applications. Our target applications include cryptographic protocol implementations, hardware-based security APIs, smartphone- and browser-based web applications, and cloud-based web services. In particular, we aim to verify the full application: both the cryptographic core and the high-level application code. We aim to verify implementations, not just models. We aim to account for computational cryptography, not just its symbolic abstraction.

We identify three key focus areas for our research in the short- to medium term.

2.1.1. Symbolic verification of cryptographic applications

Our goal is to develop our own security verification tools for models and implementations of cryptographic protocols and security APIs using symbolic cryptography. Our starting point is the tools we have previously developed: the specialized cryptographic prover ProVerif, the reverse engineering and formal test tool Tookan, and the security type systems F7 and F* for the programming language F#. These tools are already used to verify industrial-strength cryptographic protocol implementations and commercial cryptographic hardware. We plan to extend and combine these approaches to capture more sophisticated attacks on applications consisiting of protocols, software, and hardware, as well as to prove symbolic security properties for such composite systems.

2.1.2. Computational verification of cryptographic applications

We aim to develop our own cryptographic application verification tools that use the computational model of cryptography. The tools include the computational prover CryptoVerif, and the computationally sound type system Computational F7 for applications written in F#. Working together, we plan to extend these tools to analyze, for the first time, cryptographic protocols, security APIs, and their implementations under fully precise cryptographic assumptions. We also plan to pursue links between symbolic and computational verification, such as computational soundness results that enable computational proofs by symbolic techniques.

2.1.3. Provably secure web applications

We plan to develop analysis tools and verified libraries to help programmers build provably secure web applications. The tools will include a static and dynamic verification tools for client- and server-side JavaScript web applications, their verified deployment within HTML5 websites and browser extensions, as well as type-preserving compilers from high-level applications written in F* to JavaScript. In addition, we plan to model new security APIs in browsers and smartphones and develop the first formal semantics for various HTML5 web standards. We plan to combine these tools and models to analyze the security of multi-party web applications, consisting of clients on browsers and smartphones, and servers in the cloud.

3. Research Program

3.1. Symbolic verification of cryptographic applications

Despite decades of experience, designing and implementing cryptographic applications remains dangerously error-prone, even for experts. This is partly because cryptographic security is an inherently hard problem, and partly because automated verification tools require carefully-crafted inputs and are not widely applicable. To take just the example of TLS, a widely-deployed and well-studied cryptographic protocol designed, implemented, and verified by security experts, the lack of a formal proof about all its details has regularly led to the discovery of major attacks (including several in 2014) on both the protocol and its implementations, after many years of unsuspecting use.

As a result, the automated verification for cryptographic applications is an active area of research, with a wide variety of tools being employed for verifying different kinds of applications.

In previous work, the we have developed the following three approaches:

- ProVerif: a symbolic prover for cryptographic protocol models
- Tookan: an attack-finder for PKCS#11 hardware security devices
- F7: a security typechecker for cryptographic applications written in F#

3.1.1. Verifying cryptographic protocols with ProVerif

Given a model of a cryptographic protocol, the problem is to verify that an active attacker, possibly with access to some cryptographic keys but unable to guess other secrets, cannot thwart security goals such as authentication and secrecy [86]; it has motivated a serious research effort on the formal analysis of cryptographic protocols, starting with [84] and eventually leading to effective verification tools, such as our tool ProVerif.

To use ProVerif, one encodes a protocol model in a formal language, called the applied pi-calculus, and ProVerif abstracts it to a set of generalized Horn clauses. This abstraction is a small approximation: it just ignores the number of repetitions of each action, so ProVerif is still very precise, more precise than, say, tree automata-based techniques. The price to pay for this precision is that ProVerif does not always terminate; however, it terminates in most cases in practice, and it always terminates on the interesting class of *tagged protocols* [81]. ProVerif also distinguishes itself from other tools by the variety of cryptographic primitives it can handle, defined by rewrite rules or by some equations, and the variety of security properties it can prove: secrecy [79], [71], correspondences (including authentication) [80], and observational equivalences [78]. Observational equivalence means that an adversary cannot distinguish two processes (protocols); equivalences can be used to formalize a wide range of properties, but they are particularly difficult to prove. Even if the class of equivalences that ProVerif can prove is limited to equivalences between processes that differ only by the terms they contain, these equivalences are useful in practice and ProVerif is the only tool that proves equivalences for an unbounded number of sessions.

Using ProVerif, it is now possible to verify large parts of industrial-strength protocols such as TLS [75], JFK [72], and Web Services Security [77]. against powerful adversaries that can run an unlimited number of protocol sessions, for strong security properties expressed as correspondence queries or equivalence assertions. ProVerif is used by many teams at the international level, and has been used in more 30 research papers (references available at http://proverif.inria.fr/proverif-users.html).

3.1.2. Verifying security APIs using Tookan

Security application programming interfaces (APIs) are interfaces that provide access to functionality while also enforcing a security policy, so that even if a malicious program makes calls to the interface, certain security properties will continue to hold. They are used, for example, by cryptographic devices such as smartcards and Hardware Security Modules (HSMs) to manage keys and provide access to cryptographic functions whilst keeping the keys secure. Like security protocols, their design is security critical and very difficult to get right. Hence formal techniques have been adapted from security protocols to security APIs.

The most widely used standard for cryptographic APIs is RSA PKCS#11, ubiquitous in devices from smartcards to HSMs. A 2003 paper highlighted possible flaws in PKCS#11 [82], results which were extended by formal analysis work using a Dolev-Yao style model of the standard [83]. However at this point it was not clear to what extent these flaws affected real commercial devices, since the standard is underspecified and can be implemented in many different ways. The Tookan tool, developed by Steel in collaboration with Bortolozzo, Centenaro and Focardi, was designed to address this problem. Tookan can reverse engineer the particular configuration of PKCS#11 used by a device under test by sending a carefully designed series of PKCS#11 commands and observing the return codes. These codes are used to instantiate a Dolev-Yao model of the device's API. This model can then be searched using a security protocol model checking tool to find attacks. If an attack is found, Tookan converts the trace from the model checker into the sequence of PKCS#11 queries needed to make the attack and executes the commands directly on the device. Results obtained by Tookan are remarkable: of 18 commercially available PKCS#11 devices tested, 10 were found to be susceptible to at least one attack.

*3.1.3. Verifying cryptographic applications using F7 and F**

Verifying the implementation of a protocol has traditionally been considered much harder than verifying its model. This is mainly because implementations have to consider real-world details of the protocol, such as message formats, that models typically ignore. This leads to a situation that a protocol may have been proved secure in theory, but its implementation may be buggy and insecure. However, with recent advances in both program verification and symbolic protocol verification tools, it has become possible to verify fully functional protocol implementations in the symbolic model.

One approach is to extract a symbolic protocol model from an implementation and then verify the model, say, using ProVerif. This approach has been quite successful, yielding a verified implementation of TLS in F# [75]. However, the generated models are typically quite large and whole-program symbolic verification does not scale very well.

An alternate approach is to develop a verification method directly for implementation code, using well-known program verification techniques such as typechecking. F7 [73] is a refinement typechecker for F#, developed jointly at Microsoft Research Cambridge and Inria. It implements a dependent type-system that allows us to specify security assumptions and goals as first-order logic annotations directly inside the program. It has been used for the modular verification of large web services security protocol implementations [76]. F* [87] is an extension of F7 with higher-order kinds and a certifying typechecker. Both F7 and F* have a growing user community. The cryptographic protocol implementations verified using F7 and F* already represent the largest verified cryptographic applications to our knowledge.

3.2. Computational verification of cryptographic applications

Proofs done by cryptographers in the computational model are mostly manual. Our goal is to provide computer support to build or verify these proofs. In order to reach this goal, we have already designed the automatic tool CryptoVerif, which generates proofs by sequences of games. Much work is still needed in order to develop this approach, so that it is applicable to more protocols. We also plan to design and implement techniques for proving implementations of protocols secure in the computational model, by generating them from CryptoVerif specifications that have been proved secure, or by automatically extracting CryptoVerif models from implementations.

A different approach is to directly verify cryptographic applications in the computational model by typing. A recent work [85] shows how to use refinement typechecking in F7 to prove computational security for protocol implementations. In this method, henceforth referred to as computational F7, typechecking is used as the main step to justify a classic game-hopping proof of computational security. The correctness of this method is based on a probabilistic semantics of F# programs and crucially relies on uses of type abstraction and parametricity to establish strong security properties, such as indistinguishability.

In principle, the two approaches, typechecking and game-based proofs, are complementary. Understanding how to combine these approaches remains an open and active topic of research.

An alternative to direct computation proofs is to identify the cryptographic assumptions under which symbolic proofs, which are typically easier to derive automatically, can be mapped to computational proofs. This line of research is sometimes called computational soundness and the extent of its applicability to real-world cryptographic protocols is an active area of investigation.

3.3. Provably secure web applications

Web applications are fast becoming the dominant programming platform for new software, probably because they offer a quick and easy way for developers to deploy and sell their *app*s to a large number of customers. Third-party web-based apps for Facebook, Apple, and Google, already number in the hundreds of thousands and are likely to grow in number. Many of these applications store and manage private user data, such as health information, credit card data, and GPS locations. To protect this data, applications tend to use an ad hoc combination of cryptographic primitives and protocols. Since designing cryptographic applications is easy to get wrong even for experts, we believe this is an opportune moment to develop security libraries and verification techniques to help web application programmers.

As a typical example, consider commercial password managers, such as LastPass, RoboForm, and 1Password. They are implemented as browser-based web applications that, for a monthly fee, offer to store a user's passwords securely on the web and synchronize them across all of the user's computers and smartphones. The passwords are encrypted using a master password (known only to the user) and stored in the cloud. Hence, no-one except the user should ever be able to read her passwords. When the user visits a web page that has a login form, the password manager asks the user to decrypt her password for this website and automatically fills in the login form. Hence, the user no longer has to remember passwords (except her master password) and all her passwords are available on every computer she uses.

Password managers are available as browser extensions for mainstream browsers such as Firefox, Chrome, and Internet Explorer, and as downloadable apps for Android and Apple phones. So, seen as a distributed application, each password manager application consists of a web service (written in PHP or Java), some number of browser extensions (written in JavaScript), and some smartphone apps (written in Java or Objective C). Each of these components uses a different cryptographic library to encrypt and decrypt password data. How do we verify the correctness of all these components?

We propose three approaches. For client-side web applications and browser extensions written in JavaScript, we propose to build a static and dynamic program analysis framework to verify security invariants. To this end, we have developed two security-oriented type systems for JavaScript, Defensive JavaScript [74],[65] and TS* [62], and used them to guarantee security properties for a number of JavaScript applications. For Android smartphone apps and web services written in Java, we propose to develop annotated JML cryptography libraries that can be used with static analysis tools like ESC/Java to verify the security of application code. For clients and web services written in F# for the .NET platform, we propose to use F* to verify their correctness. We also propose to translate verified F* web applications to JavaScript via a verified compiler that preserves the semantics of F* programs in JavaScript.

4. Application Domains

4.1. Cryptographic Protocol Libraries

Cryptographic protocols such as TLS, SSH, IPSec, and Kerberos are the trusted base on which the security of modern distributed systems is built. Our work enables the analysis and verification of such protocols, both in their design and implementation. Hence, for example, we build and verify models and reference implementations for well-known protocols such as TLS and SSH, as well as analyze their popular implementations such as OpenSSL.

4.2. Hardware-based security APIs

Cryptographic devices such as Hardware Security Modules (HSMs) and smartcards are used to protect long-terms secrets in tamper-proof hardware, so that even attackers who gain physical access to the device cannot obtain its secrets. These devices are used in a variety of scenarios ranging from bank servers to transportation cards (e.g. Navigo). Our work investigates the security of commercial cryptographic hardware and evaluates the APIs they seek to implement.

4.3. Web application security

Web applications use a variety of cryptographic techniques to securely store and exchange sensitive data for their users. For example, a website may serve pages over HTTPS, authenticate users with a single sign-on protocol such as OAuth, encrypt user files on the server-side using XML encryption, and deploy client-side cryptographic mechanisms using a JavaScript cryptographic library. The security of these applications depends on the public key infrastructure (X.509 certificates), web browsers' implementation of HTTPS and the same origin policy (SOP), the semantics of JavaScript, HTML5, and their various associated security standards, as well as the correctness of the specific web application code of interest. We build analysis tools to find bugs in all these artifacts and verification tools that can analyze commercial web applications and evaluate their security against sophisticated web-based attacks.

5. New Software and Platforms

5.1. ProVerif

Participants: Bruno Blanchet [correspondant], Xavier Allamigeon [April–July 2004], Vincent Cheval [Sept. 2011–], Benjamin Smyth [Sept. 2009–Feb. 2010].

PROVERIF (proverif.inria.fr) is an automatic security protocol verifier in the symbolic model (so called Dolev-Yao model). In this model, cryptographic primitives are considered as black boxes. This protocol verifier is based on an abstract representation of the protocol by Horn clauses. Its main features are:

- It can handle many different cryptographic primitives, specified as rewrite rules or as equations.
- It can handle an unbounded number of sessions of the protocol (even in parallel) and an unbounded message space.

The PROVERIF verifier can prove the following properties:

- secrecy (the adversary cannot obtain the secret);
- authentication and more generally correspondence properties, of the form "if an event has been executed, then other events have been executed as well";
- strong secrecy (the adversary does not see the difference when the value of the secret changes);
- equivalences between processes that differ only by terms.

PROVERIF is widely used by the research community on the verification of security protocols (see http://proverif.inria.fr/proverif-users.html for references).

PROVERIF is freely available on the web, at proverif.inria.fr, under the GPL license.

5.2. CryptoVerif

Participants: Bruno Blanchet [correspondant], David Cadé [Sept. 2009–].

CRYPTOVERIF(cryptoverif.inria.fr) is an automatic protocol prover sound in the computational model. In this model, messages are bitstrings and the adversary is a polynomial-time probabilistic Turing machine. CRYPTOVERIF can prove secrecy and correspondences, which include in particular authentication. It provides a generic mechanism for specifying the security assumptions on cryptographic primitives, which can handle in particular symmetric encryption, message authentication codes, public-key encryption, signatures, hash functions, and Diffie-Hellman key agreements.

The generated proofs are proofs by sequences of games, as used by cryptographers. These proofs are valid for a number of sessions polynomial in the security parameter, in the presence of an active adversary. CRYPTOVERIF can also evaluate the probability of success of an attack against the protocol as a function of the probability of breaking each cryptographic primitive and of the number of sessions (exact security).

CRYPTOVERIF has been used in particular for a study of Kerberos in the computational model, and as a back-end for verifying implementations of protocols in F# and C.

CRYPTOVERIF is freely available on the web, at cryptoverif.inria.fr, under the CeCILL license.

5.3. Cryptosense Analyzer

Participants: Graham Steel [correspondant], Romain Bardou.

See also the web page http://cryptosense.com.

Cryptosense Analyzer (formerly known as Tookan) is a security analysis tool for cryptographic devices such as smartcards, security tokens and Hardware Security Modules that support the most widely-used industry standard interface, RSA PKCS#11. Each device implements PKCS#11 in a slightly different way since the standard is quite open, but finding a subset of the standard that results in a secure device, i.e. one where cryptographic keys cannot be revealed in clear, is actually rather tricky. Cryptosense Analyzer analyses a device by first reverse engineering the exact implementation of PKCS#11 in use, then building a logical model of this implementation for a model checker, calling a model checker to search for attacks, and in the case where an attack is found, executing it directly on the device. It has been used to find at least a dozen previously unknown flaws in commercially available devices.

In June 2013 we submitted a patent application (13 55374) on the reverse engineering process. We also concluded a license agreement between Inria PROSECCO and the nascent spin-off company Cryptosense to commercialize the tool.

5.4. miTLS

Participants: Karthikeyan Bhargavan [correspondant], Antoine Delignat-Lavaud, Cedric Fournet [Microsoft Research], Markulf Kohlweiss [Microsoft Research], Alfredo Pironti, Pierre-Yves Strub [IMDEA], Santiago Zanella-Béguelin [Microsoft Research], Jean Karim Zinzindohoue.

miTLS is a verified reference implementation of the TLS security protocol in F#, a dialect of OCaml for the .NET platform. It supports SSL version 3.0 and TLS versions 1.0-1.2 and interoperates with mainstream web browsers and servers. miTLS has been verified for functional correctness and cryptographic security using the refinement typechecker F7.

A paper describing the miTLS library was published at IEEE S&P 2013, and two updates to the software were released in 2013. The software and associated research materials are available from http://mitls.rocq.inria.fr.

5.5. WebSpi

Participants: Karthikeyan Bhargavan [correspondant], Chetan Bansal [Microsoft], Antoine Delignat-Lavaud, Sergio Maffeis [Imperial College London].

WebSpi is a library that aims to make it easy to develop models of web security mechanisms and protocols and verify them using ProVerif. It captures common modeling idioms (such as principals and dynamic compromise) and defines a customizable attacker model using a set of flags. It defines an attacker API that is designed to make it easy to extract concrete attacks from ProVerif counterexamples.

WebSpi has been used to analyze social sign-on and social sharing services offered by prominent social networks, such as Facebook, Twitter, and Google, on the basis of new open standards such as the OAuth 2.0 authorization protocol.

WebSpi has also been used to investigate the security of a number of cryptographi web applications, including password managers, cloud storage providers, an e-voting website and a conference management system.

WebSpi is under development and released as an open source library at http://prosecco.inria.fr/webspi/

5.6. Defensive JavaScript

Participants: Antoine Delignat-Lavaud [correspondant], Karthikeyan Bhargavan, Sergio Maffeis [Imperial College London].

Defensive JavaScript (DJS) is a subset of the JavaScript language that guarantees the behaviour of trusted scripts when loaded in an untrusted web page. Code in this subset runs independently of the rest of the JavaScript environment. When propertly wrapped, DJS code can run safely on untrusted pages and keep secrets such as decryption keys. DJS is especially useful to write security APIs that can be loaded in untrusted pages, for instance an OAuth library such as the one used by "Login with Facebook". It is also useful to write secure host-proof web applications, and more generally for cryptography that happens on the browser.

The DJS type checker and various libraries written in DJS are available from http://www.defensivejs.com.

5.7. F*

Participants: Nikhil Swamy [Microsoft Research], Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cedric Fournet [Microsoft Research], Catalin Hritcu, Chantal Keller, Aseem Rastogi, Pierre-Yves Strub.

F* is a new higher order, effectful programming language (like ML) designed with program verification in mind. Its type system is based on a core that resembles System Fω (hence the name), but is extended with dependent types, refined monadic effects, refinement types, and higher kinds. Together, these features allow expressing precise and compact specifications for programs, including functional correctness properties. The F* type-checker aims to prove that programs meet their specifications using an automated theorem prover (usually Z3) behind the scenes to discharge proof obligations. Programs written in F* can be translated to OCaml, F#, or JavaScript for execution.

A detailed description of F* (circa 2011) appeared in the Journal of Functional Programming [88]. F* has evolved substantially since then. The latest version of F* is written entirely in F*, and bootstraps in OCaml and F#. It is under active development at GitHub: https://github.com/FStarLang and the official webpage is at http://fstar-lang.org.

6. New Results

6.1. Highlights of the Year

This year, we published 17 articles in international peer-reviewed journals and conferences, including papers in prestigious conferences such as POPL (2 papers) and all the top conferences in computer security: IEEE S&P Oakland (2 papers), CRYPTO, ACM CCS, NDSS, and Financial Cryptography. Our papers in these top venues (discussed later in New Results) serve as highlights of our research during the year. In addition to these papers, we published 1 PhD thesis and several technical reports.

We released updates to miTLS, ProVerif, CryptoVerif, and started working on a brand-new version of F*. We discovered serious vulnerabilities in a number of TLS libraries, web browsers, and web servers, resulting in 6 published CVEs, and over a dozen software updates based on our recommendations in widely used software such as Firefox, Chrome, Internet Explorer, Safari, OpenSSL, Java, and Mono.

We organized a winter school “The Joint EasyCrypt-F*-CryptoVerif School 2014” which attracted industrial researchers, academics, and students from around the world. Over 75 students learned to use cryptographic verification tools from instructors at Inria, IMDEA, and Microsoft Research. Two of the tools: CryptoVerif and F* are being developed in collaboration with Inria.

If we were to choose one research theme as our highlight of the year, it would be our activities surrounding Transport Layer Security (TLS):

- At CRYPTO 2014, we published a detailed cryptographic proof of the TLS handshake as implemented in miTLS
- At NDSS 2014, we published a study in the use of X.509 certificates in TLS servers on the web
- At IEEE S&P (Oakland), we published a new attack on the TLS protocol called the *triple handshake*, which affected all TLS libraries and mainstream TLS applications such as web browsers.
- To prevent our attack, we proposed patches to major software libraries as part of responsible disclosure. Our research directly led to security updates for all major web browsers and TLS implementations.
- We also proposed a long-term countermeasure for our attack, the TLS Session Hash extension, which we published as an internet draft and presented at the IETF. This draft is on its way to being a published standard and is already implemented in all major TLS libraries.
- We participated in the design of next version (1.3) of the TLS protocol. We hosted an interim TLS working group meeting in Paris. Our proposals such as the session hash construction are now an integral part of the new design, and we continue consulting on the design and implementation of TLS.

6.2. Verification of Security Protocols in the Symbolic Model

Participants: Bruno Blanchet, Miriam Paiola, Robert Künnemann.

Miriam Paiola wrote and defended her PhD thesis on the verification of security protocols with lists [45].

Robert Künnemann published a paper at the IEEE S&P conference on how to extend symbolic cryptographic protocol verifiers to account for global state [60].

Bruno Blanchet published a tutorial on the protocol verifier ProVerif [66], as a follow-up to his teaching in the FOSAD'13 summer school last year.

The applied pi calculus is a widely used language for modeling security protocols, including as a theoretical basis of ProVerif. However, the seminal paper that describes this language (Abadi and Fournet, POPL'01) does not come with proofs, and detailed proofs for the results in this paper were never published. This year, Martin Abadi, Bruno Blanchet, and Cedric Fournet wrote detailed proofs for the main theorems of this paper. This work was also an opportunity to fix a few minor details in the results and to tune the calculus to improve it and make it closer to the input calculus of ProVerif. We plan to submit this work as a journal paper.

6.3. Verification of Security Protocols in the Computational model

Participants: Bruno Blanchet, David Cadé.

We worked on our computationally-sound protocol verifier CryptoVerif in two directions.

First, this verifier includes a specialized compiler that generates secure implementations of protocols from CryptoVerif specifications. We completed a journal version of the proof that this compiler preserves security, which is to appear in the Journal of Computer Security [48]

Second, Bruno Blanchet extended CryptoVerif with support for equational theories: associativity, commutativity, non-commutative and commutative groups, exclusive or. The goal is to be able to verify protocols that rely on the algebraic properties of groups and exclusive or. The extended tool is available at http://cryptoverif.inria.fr.

6.4. Computationally Complete Symbolic Attacker Models

Participants: Gergei Bana, Hubert Comon-Lundh.

A new approach to computational verification is to define a *computationally complete* symbolic attacker, so that a symbolic proof against this attacker can be shown to imply a computational proof of security. Following this line of inquiry, Gergei Bana and Hubert Comon-Lundh recently published work on proving computational reachability properties using symbolic techniques.

Gergei Bana (along with Hubert Comon-Lundh) published a paper on how to extend this work to prove stronger security properties expressed as equivalences [50]. Hence, the proof techniq can now be used also for properties like anonymity, strong secrecy etc. Besides being able to prove such properties, another advantage of this extension is that modern security properties of cryptographic primitives are also formulated in terms of indistinguishability, which makes it easier to translate the security properties cryptographers define to our language than before.

Using the computationally complete symbolic attacker, writing up a full, computationally sound proof (and identifying new attacks) for the NSL protocol when agents can run both roles, including running sessions with themselves. The proof is first attempted without any assumption other than that the encryption is CCA2 and that honest names are assigned at the beginning (that is, absolutely nothing about parsing: triples may be independent from pairs, pairing the projection of pairs may not give back the original item etc.). Along the way, we identified new attacks absent of some necessary parsing properties that implementations may not satisfy in general. Then with these additional parsing properties added to the properties satisfied by the implementation, we verified the protocol, namely secrecy, authentication and agreement. The project included graphical representation of the proof steps and the attacks. Type-flaw attacks that can be found in the literature have been reproduced this way, but a number of other attacks have also be revealed that cannot be found with the Dolev-Yao technique, and have not been found by other computational techniques either, although they are realistic. This is joint work with Pedro Adao of IST Lisbon. We hope to publish parts of this work to illustrate proving strategies. The current state of the writeup is available at http://prosecco.gforge.inria.fr/personal/gebana/nsl-long-both-roles.pdf

6.5. Authentication Attacks against Transport Layer Security

Participants: Karthikeyan Bhargavan [correspondant], Antoine Delignat-Lavaud, Cedric Fournet [Microsoft Research], Markulf Kohlweiss [Microsoft Research], Alfredo Pironti, Pierre-Yves Strub [IMDEA].

We discovered an important client impersonation attack on the Transport Layer Security protocol called the *triple handshake attack.* The attack is on the standard protocol and hence all compliant implementations were potentially at risk. Hence, we systematically followed responsible disclosure by notifying all major web browsers and TLS implementors, and then working with the TLS working group to design a countermeasure. The research results of this work were published at IEEE S&P [53].

To TLS implementors, we proposed short-term countermeasures that mitigated our attack, leading to security updates to all major web browsers: Google Chrome (CVE-2013-6628), Mozilla Firefox (CVE-2014-1491), Internet Explorer (CVE-2014-1771), Apple Safari (CVE-2014-1295), as well as to non-browser TLS libraries such as Oracle JSSE (CVE-2014-6457) and RSA BSAFE (CVE-2014-4630). For more details, see http://secure-resumption.com

To the TLS working group, we proposed a new cryptographic construction called the *session hash* that fundamentally alters the cryptographic core of TLS. This construction has now been adopted as a protocol extension to TLS 1.2 and has been integrated into the upcoming TLS 1.3. We expect an IETF standard for this construction to be published in early 2015.

While the triple handshake attacks primarily affect client-authentication, server authentication in HTTPS (HTTP over TLS) primarily relies on X.509 public key certificates. Antoine Delignat-Lavaud along with co-authors at Microsoft research published a paper at NDSS 2015 on a large-scale study of the Web PKI: how certificates are issued and used on the web [56]. Our work uncovered many unsafe practices and suggested best practices and new security policies.

Antoine Delignat-Lavaud also showed how the unsafe sharing of certificates across multiple websites could be exploited to fully compromise the same origin policty for websites, using an vulnerability called virtual host confusion. These results were discussed in a talk at BlackHat USA: for details see http://bh.ht.vc. A research paper on these attacks is forthcoming at WWW'2015.

6.6. A Verified Reference Implementation of Transport Layer Security

Participants: Benjamin Beurdouche [correspondant], Karthikeyan Bhargavan [correspondant], Antoine Delignat-Lavaud, Cedric Fournet [Microsoft Research], Markulf Kohlweiss [Microsoft Research], Alfredo Pironti, Pierre-Yves Strub [IMDEA], Santiago Zanella-Béguelin [Microsoft Research], Jean Karim Zinzindohoue.

Following on from previous work in the miTLS project, we published new versions of miTLS (http://mitls.org) that implemented various protocol extensions including the new session hash extension.

At CRYPTO 2014 [55], we published the first detailed cryptographic proof of an implementation of the TLS Handshake. The implementation consists of about 5000 lines of code and is equipped with about 2500 lines of security annotations written in F7, and a 3000 line EasyCrypt proof.

Currently, we are extending and improving this verified implementation to cover commonly used TLS extensions as well as TLS 1.3, the new version of TLS that we are actively involved in designing. We recently hosted a meeting of the TLS working group at Inria in Paris and are active members of the core working group.

In parallel, we have been analyzing other implementations of TLS and testing them against our implementation, both to ensure interoperability and to uncover bugs. Our analyses have led to the discovery of serious state machine vulnerabilities in many TLS implementations including Oracle JSSE, NSS, OpenSSL, SecureTransport, CyaSSL, Mono, and RSA BSAFE. On our recommendations, all these TLS libraries have issued important security updates in 2014.

6.7. Verified implementations of cryptographic primitives

Participants: Evmorfia-Iro Bartzia, Jean Karim Zinzindohoue, Pierre-Yves Strub, Karthikeyan Bhargavan.

Cryptographic libraries underpin the security of all security protocol implementations. A bug in the implementation of one primitive could enable an attacker to break the security of the full protocol. Hence, establishing the formal correctness of an efficient cryptographic mechanism is a much-desired but still open goal. We are investigating two directions of research towards this goal, specifically in the context of elliptic curve libraries.

Evmorfia-Iro Bartzia and Pierre-Yves Strub are building a Coq library that enables the precise proof of elliptic curve algorithms, and the automatic extraction of verified OCaml code that implements these algorithms. Their most recent result is the formal proof of a non-trivial theorem by Picard: the existence of an isomorphism between an elliptic curve and its Picard group of divisors. This work led to the publication "A formal library for Elliptic Curves in the Coq proof Assistant" and was presented at the ITP 2014 conference [51]. We have also been working on a formal proof of correctness of the GLV algorithm for scalar multiplication in Coq, using the above development and the CoqEal methodology. At present, we have an implementation of the algorithm in the OCaml language and a formal development regarding multiexponentiation, endomorphisms, scalar decomposition and coordinates in both affine and projective spaces. This work is still in progress.

Jean Karim Zinzindohoue and Karthikeyan Bhargavan are investigating the direct verification of implementations of the Curve25519 elliptic curve that is emerging as the preferred new curve for a variety of cryptographic standards, including TLS and the W3C web cryptography API. We use standard program verification tools such as the Frama-C/Why3 verification toolkit for a C implementation of Curve25519 and the F* typechecker for an OCaml implementation of the curve. This work is still in progress.

6.8. Dynamic Security Verification and Testing

Participants: Catalin Hritcu, Arthur Azevedo de Amorim, Zoi Paraskevopoulou, Nikolaos Giannarakis.

We investigated two directions in the runtime security verification of software and hardware systems.

Catalin Hritcu, Arthur Azevedo de Amorim, Nick Giannarakis, and their collaborators at University of Pennsylvania and Portland State University published work on *micro-policies* a generic framework for defining tag-based reference monitors on a simple tagged RISC processor. The framework was formalized and verified in the Coq proof assistant and was used to define and verify micro-policies for dynamic sealing, control-flow integrity, compartmentalization, and memory safety. This work resulted in publications at POPL 2014 [63], ASPLOS 2015 [58], and another paper is in submission.

Catalin Hritcu along with his co-authors worked on a testing framework for security and functional correctness. We published a journal paper about testing noninterference [68] and submitted an ANR JCJC grant pre-proposal on the whole project. Catalin Hritcu also worked with an intern Zoe Paraskevopoulou on this topic, who successfully defended her thesis at NTU Athens. We plan to publish a polished version of that in the near future.

6.9. Verified Security for Web Applications

Participants: Karthikeyan Bhargavan [correspondant], Chetan Bansal [Microsoft], Antoine Delignat-Lavaud, Sergio Maffeis [Imperial College London].

Karthikeyan Bhargavan, Antoine Delignat-Lavaud, and co-authors published a tutorial on Defensive JavaScript, a typed subset of JavaScript that is designed to be used for security-critical components such as cryptographic libraries that may be deployed within untrusted web pages. This tutorial was published as a follow-up of Karthikeyan Bhargavan's lectures at the FOSAD'13 summer school [65].

Karthikeyan Bhargavan, Antoine Delignat-Lavaud, and co-authors also published a journal version of their work on the WebSpi web security modeling library [47], one of the few formal models that captures the detailed security assumptions of various web mechanisms.

Karthikeyan Bhargavan along with collaborators at Microsoft Research published a paper at POPL 2014 on TS*: a new gradual type system for a large subset of JavaScript [47]. We showed how to compile and safely deploy well-typed TS* programs as standard JavaScript in websites. Such programs preserve their types even if other code running on the website is malicious. Our work was used as a basis for further work on the TypeScript compiler and typechecker developed at Microsoft.

6.10. Electronic Voting and Auctions

Participants: Benjamin Smyth [correspondant], Elizabeth Quaglia, Adam Mccarthy, David Bernhard.

Benjamin Smyth continued his work on proving privacy properties of electronic voting protocols. Smyth and Bernhard worked on a new formal definition of ballot secrecy that works even if the bulletin board (used to publish votes) is malicious [69].

Benjamin Smyth, Elizabeth Quaglia, and Adam McCarthy observed that existing electronic voting schemes could be used as core building blocks for electronic auction protocols. Using this link, they build two new e-auction protocols Hawk and Aucitas by building on top of the e-voting protocols Helios and Civitas resp. They prove that their protocols enjoy many desired security properties. This result was published at Financial Cryptography 2014 [61].

7. Partnerships and Cooperations

7.1. National Initiatives

7.1.1. ANR

7.1.1.1. ProSe

Title: ProSe: Security protocols : formal model, computational model, and implementations (ANR VERSO 2010.)

Other partners: Inria/Cascade, ENS Cachan-Inria/Secsi, LORIA-Inria/Cassis, Verimag.

Duration: December 2010 - December 2014.

Coordinator: Bruno Blanchet, Inria (France)

Abstract: The goal of the project is to increase the confidence in security protocols, and in order to reach this goal, provide security proofs at three levels: the symbolic level, in which messages are terms; the computational level, in which messages are bitstrings; the implementation level: the program itself.

7.1.1.2. AJACS

Title: AJACS: Analyses of JavaScript Applications: Certification and Security

Other partners: Inria-Rennes/Celtique, Inria-Saclay/Toccata, Inria-Sophia Antipolis/INDES, Imperial College London

Duration: October 2014 - March 2019.

Coordinator: Alan Schmitt, Inria (France)

Abstract: The goal of the AJACS project is to provide strong security and privacy guarantees 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, to develop and prove correct analyses for JavaScript programs, and to design and certify security and privacy enforcement mechanisms.

7.1.2. FUI

7.1.2.1. Pisco

Title: PISCO

Partners: Bull, Cassadian, CEA, CS, Saferiver, Serpikom, Telecom Paristech

Duration: January 2013 - December 2014.

Coordinator: Liliana Calabanti, Bull (France)

Abstract: The goal of the project is to develop a prototype of a new secure applicance based on a virtual machine architecture accessing an HSM. The role of PROSECCO is to contribute to the analysis of security http://www.systematic-paris-region.org/en/projets/pisco

7.2. European Initiatives

7.2.1. FP7 & H2020 Projects

7.2.1.1. CRYSP

Type: FP7

Defi: NC

Instrument: ERC Starting Grant

Objectif: NC

Duration: November 2010 - October 2015

Coordinator: Karthikeyan Bhargavan

Partner: Inria (France)

Inria contact: Valérie Boutheon

Abstract: The goal of this grant is to develop a collaborative specification framework and to build incremental, modular, scalable verification techniques that enable a group of collaborating programmers to build an application and its security proof side-by-side. We propose to validate this framework by developing the first large-scale web application and full-featured cryptographic protocol libraries with formal proofs of security.

7.3. International Initiatives

7.3.1. Inria International Partners

7.3.1.1. Informal International Partners

- Microsoft Research (Cambridge, Redmond): Joint research and development on F*, miTLS, and JavaScript with Cedric Fournet, Markulf Kohlweiss, and Nikhil Swamy
- University of Pennsylvania, Portland State University, Harvard University: Joint research on Micro-Policies: Formally Verified Low-Level Tagging Schemes for Safety and Security
- Imperial College (London): Joint research on web application security with Sergio Maffeis
- University of Venice Ca'Foscari: Joint research on security APIs

7.4. International Research Visitors

7.4.1. Visits of International Scientists

- Nikhil Swamy, Limin Jia, Benjamin Pierce, Cedric Fournet visited our group and gave seminars.
- Matteo Maffei, Dominique Unruh, Gilles Barthe, François Dupressoir, came to teach at the Joint EasyCrypt-F*-CryptoVerif School.

7.4.1.1. Internships

Cairns Kelsey

Date: Mar 2014 - May 2014

Institution: Washington State University (USA)

Paraskevopoulou Zoi

Date: Apr 2014 - Sep 2014

Institution: National Technical University of Athens (Greece)

Giannarakis Nikolaos

Date: Apr 2014 - Sep 2014

Institution: National Technical University of Athens (Greece)

Azevedo De Amorim, Arthur

Date: Mar 2014 - Aug 2014

Institution: University of Pennsylvania (USA)

Jindal Shubham

Date: May 2014 - Jul 2014

Institution: Indian Institute of Technology Delhi (India)

Thomson Susan

Date: Jun 2014 - Aug 2014

Institution: University of Bristol (UK)

8. Dissemination

8.1. Promoting Scientific Activities

8.1.1. Scientific events organisation

8.1.1.1. General chair, scientific chair

- PROSECCO organized and hosted the **The Joint EasyCrypt-F*-CryptoVerif School 2014** from Nov 24-28. The main organizers were Catalin Hritcu, Karthikeyan Bhargavan, and Pierre-Yves Strub. The school attracted over 75 participants and was primarily funded by the ERC Starting Grant CRYSP, with supporting funds from CryptoForma, MSR-Inria, and CryptoSense. The school's website is at https://wiki.inria.fr/prosecco/The_Joint_EasyCrypt-F*-CryptoVerif_School_2014
- PROSECCO hosted the **IETF TLS Working Group Interim Meeting** from October 21-22. The main organizers were Alfredo Pironti and Karthikeyan Bhargavan. The meeting involved a dozen members of industry and academia and discussed the detailed design of the next version of TLS. The meeting's website is at https://wiki.inria.fr/prosecco/IETF_TLS_Interim_October_2014

8.1.2. Scientific events selection

8.1.2.1. Chair of conference program committee

- ASA workshop – July 2014, Vienna, Austria: Graham Steem

8.1.2.2. Member of the conference program committee

- ETAPS HotSpot – April 2014, Grenoble, France: Bruno Blanchet
- IEEE CSF – July 2014, Vienna, Austria: Bruno Blanchet
- ACM CCS – November 2014, Scottsdale AZ, USA: Graham Steel
- ETAPS POST – April 2014, Grenoble, France: Graham Steel
- PPDP – September 2014, Canterbury, UK: Chantal Keller
- ACNS – June 2014, Lausanne, Switzerland: Catalin Hritcu
- FCS-FCC 2014 – July 2014, Vienna, Austria: Catalin Hritcu, Benjamin Smyth
- IEEE CloudCom – December 2014, Singapore: Benjamin Smyth

8.1.2.3. Reviewer

- Members of Prosecco reviewed papers for many major conferences and workshops including ACM CCS, ACM POPL, IEEE S&P, etc.

8.1.3. Journal

8.1.3.1. Member of the editorial board

Associate Editor

– of the *International Journal of Applied Cryptography (IJACT)* – Inderscience Publishers: Bruno Blanchet

8.1.3.2. Reviewer

- Members of Prosecco reviewed papers for many major journals including ACM TOPLAS, ACM TISSEC, Journal of Computer Security, etc.

8.2. Teaching - Supervision - Juries

8.2.1. Teaching

Master: Bruno Blanchet, Cryptographic protocols: formal and computational proofs, 18h equivalent TD, master M2 MPRI, université Paris VII, France

Doctorat: Bruno Blanchet, course on CryptoVerif, joint EasyCrypt-F*-CryptoVerif school, Paris, France

Master: Karthikeyan Bhargavan, Cryptographic protocols: formal and computational proofs, 18h equivalent TD, master M2 MPRI, université Paris VII, France

License: Karthikeyan Bhargavan, INF431, INF421, INF672, INF321, introductory courses at at Ecole Polytechnique, Palaiseau, France

Doctorat: Catalin Hritcu, course on F*, joint EasyCrypt-F*-CryptoVerif school, Paris, France

Doctorat: Graham Steel, 3 teaching hours in Venice Ca' Foscari on Security APIs

Doctorat: Benjamin Smyth, Electronic Voting: Introduction, privacy and verifiability definitions, and examples @ CryptoBG'14, Bulgaria

8.2.2. Supervision

- PhD: Miriam Paiola
  Verification of cryptographic protocols with lists of unbounded length,
  defended on May 28, 2014, supervised by Bruno Blanchet
- PhD in progress: Evmorfia-Iro Bartzia
  Machine-checked program verification for concrete cryptography,
  started October 2011, supervised by Karthikeyan Bhargavan and Pierre-Yves Strub
- PhD in progress: Antoine Delignat-Lavaud
  Verified security for web applications,
  started September 2012, supervised by Karthikeyan Bhargavan
- PhD in progress: Jean Karim Zinzindohoue
  Analyzing cryptographic protocols and their implementations,
  started September 2014, supervised by Karthikeyan Bhargavan

8.2.3. Juries

- Cyrille Wiedling – Ph.D. – Nov. 21, 2014 – Université de Lorraine
  Formal Verification of Advanced Families of Security Protocols: E-Voting and APIs
  Bruno Blanchet (reviewer)
- Guillaume Scerri – Ph.D. – Jan. 29, 2015 – ENS Cachan
  Proofs of security protocols revisited
  Bruno Blanchet (reviewer)
- Bart van Delft – Licentiate – Mar. 14, 2014 – Chalmers University
  Karthikeyan Bhargavan (discussion leader)

8.3. Popularization

8.3.1. Seminars

- Graham Steel: invited talks at FCS-FCC 2014 (Vienna)
- Graham Steel: invited keynote at Grande Region security day (Saarbrucken).
- Bruno Blanchet: invited talk at the Dagstuhl seminar "The synergy between programming languages and cryptography".
- Karthikeyan Bhargavan: invited talk at the IETF TLS Working Group to discuss the Triple Handshake Attack (London)
- Karthikeyan Bhargavan: invited talk at the Dagstuhl seminar "The synergy between programming languages and cryptography".
- Karthikeyan Bhargavan: invited talk at Les Journées Scientifiques Inria in Lille
- Karthikeyan Bhargavan: invited panelist at Security Standardization Research workshop in Surrey UK
- Karthikeyan Bhargavan: invited keynote at Santa's Crypto Workshop 2014 (Prague)
- Antoine Delignat-Lavaud: briefing at BlackHat USA on "The BEAST Wins Again: Why TLS Keeps Failing to Protect HTTP"
- Catalin Hritcu: invited keynote at Grande Region security day (Saarbrucken).
- Catalin Hritcu: seminar at Groupe de travail LTP du GDR GPL

- Catalin Hritcy: seminar at Groupe de travail Théorie des types et réalisabilité

8.3.2. Vulnerabily Reports

- Karthikeyan Bhargavan, Antoine Delignat-Lavaud, and Alfredo Pironti reported the so-called Triple Handshake attacks on TLS implementations leading to security updates to all major web browsers: Google Chrome (CVE-2013-6628), Mozilla Firefox (CVE-2014-1491), Internet Explorer (CVE-2014-1771), Apple Safari (CVE-2014-1295), as well as to non-browser TLS libraries such as Oracle JSSE (CVE-2014-6457) and RSA BSAFE (CVE-2014-4630). For more details, see http://secure-resumption.com
- Antoine Delignat-Lavaud reported virtual host confusion attacks on a number of web servers, leading to security updates to the Akamai content delivery network, Dropbox, Bugzilla, as well as the NGINX web server. His results were presented at BlackHat USA and are summarized at http://bh.ht.vc
- Karthikeyan Bhargavan reported state machine attacks on major TLS libraries, such as OpenSSL (CVE-2014-3572), NSS, JSSE, CyaSSL, and SecureTransport, leading to security updates in all these libraries.

9. Bibliography

Major publications by the team in recent years

[1] M. ABADI, B. BLANCHET. *Analyzing Security Protocols with Secrecy Types and Logic Programs*, in "Journal of the ACM", January 2005, vol. 52, n^{o} 1, pp. 102–146

[2] M. ABADI, B. BLANCHET. *Computer-Assisted Verification of a Protocol for Certified Email*, in "Science of Computer Programming", October 2005, vol. 58, n^{o} 1–2, pp. 3–27, Special issue SAS'03

[3] M. ABADI, B. BLANCHET, H. COMON-LUNDH. *Models and Proofs of Protocol Security: A Progress Report*, in "21st International Conference on Computer Aided Verification (CAV'09)", Grenoble, France, A. BOUAJJANI, O. MALER (editors), Lecture Notes on Computer Science, Springer Verlag, June 2009, vol. 5643, pp. 35–49

[4] M. ABADI, B. BLANCHET, C. FOURNET. *Just Fast Keying in the Pi Calculus*, in "Programming Languages and Systems: Proceedings of the 13th European Symposium on Programming (ESOP'04)", Barcelona, Spain, D. SCHMIDT (editor), Lecture Notes on Computer Science, Springer Verlag, March 2004, vol. 2986, pp. 340–354

[5] M. ABADI, B. BLANCHET, C. FOURNET. *Just Fast Keying in the Pi Calculus*, in "ACM Transactions on Information and System Security (TISSEC)", July 2007, vol. 10, n^{o} 3, pp. 1–59

[6] X. ALLAMIGEON, B. BLANCHET. *Reconstruction of Attacks against Cryptographic Protocols*, in "18th IEEE Computer Security Foundations Workshop (CSFW-18)", Aix-en-Provence, France, IEEE Computer Society, June 2005, pp. 140–154

[7] G. BANA, K. HASEBE, M. OKADA. *Computationally Complete Symbolic Attacker and Key Exchange*, in "ACM Conference on Computer and Communications Security (CCS'13)", Berlin, Germany, ACM, 2013, pp. 1231–1246, http://hal.inria.fr/hal-00918848

[8] C. BANSAL, K. BHARGAVAN, S. MAFFEIS. *Discovering Concrete Attacks on Website Authorization by Formal Analysis*, in "25th IEEE Computer Security Foundations Symposium (CSF'12)", Cambridge, MA, USA, IEEE, June 2012, pp. 247–262, http://moscova.inria.fr/~karthik/pubs/discovering_concrete_attacks_csf12.pdf

[9] R. BARDOU, R. FOCARDI, Y. KAWAMOTO, L. SIMIONATO, G. STEEL, J.-K. TSAY. *Efficient Padding Oracle Attacks on Cryptographic Hardware*, in "CRYPTO", 2012, pp. 608–625

[10] J. BENGTSON, K. BHARGAVAN, C. FOURNET, A. D. GORDON, S. MAFFEIS. *Refinement types for secure implementations*, in "21st IEEE Computer Security Foundations Symposium (CSF'08)", 2008, pp. 17–32

[11] J. BENGTSON, K. BHARGAVAN, C. FOURNET, A. D. GORDON, S. MAFFEIS. *Refinement types for secure implementations*, in "ACM Trans. Program. Lang. Syst.", 2011, vol. 33, n^{o} 2, 8 p.

[12] K. BHARGAVAN, A. DELIGNAT-LAVAUD, S. MAFFEIS. *Language-Based Defenses Against Untrusted Browser Origins*, in "Proceedings of the 22th USENIX Security Symposium", 2013, http://hal.inria.fr/hal-00863372

[13] K. BHARGAVAN, C. FOURNET, R. CORIN, E. ZALINESCU. *Cryptographically verified implementations for TLS*, in "ACM Conference on Computer and Communications Security", 2008, pp. 459-468

[14] K. BHARGAVAN, C. FOURNET, R. CORIN, E. ZALINESCU. *Verified Cryptographic Implementations for TLS*, in "ACM Transactions Inf. Syst. Secur.", March 2012, vol. 15, n^{o} 1, pp. 3:1–3:32, http://doi.acm.org/10.1145/2133375.2133378

[15] K. BHARGAVAN, C. FOURNET, A. D. GORDON. *Modular Verification of Security Protocol Code by Typing*, in "ACM Symposium on Principles of Programming Languages (POPL'10)", 2010, pp. 445–456

[16] K. BHARGAVAN, C. FOURNET, A. D. GORDON, N. SWAMY. *Verified Implementations of the Information Card Federated Identity-Management Protocol*, in "Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS'08)", ACM Press, 2008, pp. 123–135

[17] K. BHARGAVAN, C. FOURNET, M. KOHLWEISS, A. PIRONTI, P.-Y. STRUB. *Implementing TLS with Verified Cryptographic Security*, in "IEEE Symposium on Security & Privacy (Oakland)", San Francisco, United States, 2013, pp. 445-462, http://hal.inria.fr/hal-00863373

[18] B. BLANCHET, M. ABADI, C. FOURNET. *Automated Verification of Selected Equivalences for Security Protocols*, in "20th IEEE Symposium on Logic in Computer Science (LICS 2005)", Chicago, IL, IEEE Computer Society, June 2005, pp. 331–340

[19] B. BLANCHET, M. ABADI, C. FOURNET. *Automated Verification of Selected Equivalences for Security Protocols*, in "Journal of Logic and Algebraic Programming", February–March 2008, vol. 75, n^{o} 1, pp. 3–51

[20] B. BLANCHET. *Automatic Proof of Strong Secrecy for Security Protocols*, in "IEEE Symposium on Security and Privacy", Oakland, California, May 2004, pp. 86–100

[21] B. BLANCHET. *An Automatic Security Protocol Verifier based on Resolution Theorem Proving (invited tutorial)*, in "20th International Conference on Automated Deduction (CADE-20)", Tallinn, Estonia, July 2005

[22] B. BLANCHET. *Security Protocols: From Linear to Classical Logic by Abstract Interpretation*, in "Information Processing Letters", September 2005, vol. 95, n^{o} 5, pp. 473–479

[23] B. BLANCHET. *A Computationally Sound Mechanized Prover for Security Protocols*, in "IEEE Symposium on Security and Privacy", Oakland, California, May 2006, pp. 140-154

[24] B. BLANCHET. *Computationally Sound Mechanized Proofs of Correspondence Assertions*, in "20th IEEE Computer Security Foundations Symposium (CSF'07)", Venice, Italy, IEEE, July 2007, pp. 97–111

[25] B. BLANCHET. *A Computationally Sound Mechanized Prover for Security Protocols*, in "IEEE Transactions on Dependable and Secure Computing", October–December 2008, vol. 5, n^{o} 4, pp. 193–207

[26] B. BLANCHET. *Vérification automatique de protocoles cryptographiques : modèle formel et modèle calculatoire. Automatic verification of security protocols: formal model and computational model*, Université Paris-Dauphine, November 2008, Mémoire d'habilitation à diriger des recherches

[27] B. BLANCHET. *Automatic Verification of Correspondences for Security Protocols*, in "Journal of Computer Security", July 2009, vol. 17, n^{o} 4, pp. 363–434

[28] B. BLANCHET. *Using Horn Clauses for Analyzing Security Protocols*, in "Formal Models and Techniques for Analyzing Security Protocols", V. CORTIER, S. KREMER (editors), Cryptology and Information Security Series, IOS Press, March 2011, vol. 5, pp. 86–111

[29] B. BLANCHET. *Automatically Verified Mechanized Proof of One-Encryption Key Exchange*, in "25th IEEE Computer Security Foundations Symposium (CSF'12)", Cambridge, MA, USA, IEEE, June 2012, pp. 325–339, http://prosecco.gforge.inria.fr/personal/bblanche/publications/BlanchetCSF12.pdf

[30] B. BLANCHET, A. CHAUDHURI. *Automated Formal Analysis of a Protocol for Secure File Sharing on Untrusted Storage*, in "IEEE Symposium on Security and Privacy", Oakland, CA, IEEE, May 2008, pp. 417–431

[31] B. BLANCHET, A. D. JAGGARD, A. SCEDROV, J.-K. TSAY. *Computationally Sound Mechanized Proofs for Basic and Public-key Kerberos*, in "ACM Symposium on Information, Computer and Communications Security (ASIACCS'08)", Tokyo, Japan, ACM, March 2008, pp. 87–99

[32] B. BLANCHET, M. PAIOLA. *Automatic Verification of Protocols with Lists of Unbounded Length*, in "CCS'13 - ACM Conference on Computer and Communications Security", Berlin, Germany, ACM, 2013, pp. 573–584 [*DOI :* 10.1145/2508859.2516679], http://hal.inria.fr/hal-00918849

[33] B. BLANCHET, A. PODELSKI. *Verification of Cryptographic Protocols: Tagging Enforces Termination*, in "Theoretical Computer Science", March 2005, vol. 333, n^{o} 1-2, pp. 67–90, Special issue FoSSaCS'03

[34] B. BLANCHET, D. POINTCHEVAL. *Automated Security Proofs with Sequences of Games*, in "CRYPTO'06", Santa Barbara, CA, C. DWORK (editor), Lecture Notes on Computer Science, Springer Verlag, August 2006, vol. 4117, pp. 537–554

[35] M. BORTOLOZZO, M. CENTENARO, R. FOCARDI, G. STEEL. *Attacking and Fixing PKCS#11 Security Tokens*, in "Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS'10)", Chicago, Illinois, USA, ACM Press, October 2010, pp. 260-269

[36] D. CADÉ. *Proved Implementations of Cryptographic Protocols in the Computational Model*, Université Paris VII, December 2013

[37] V. CORTIER, B. SMYTH. *Attacking and fixing Helios: An analysis of ballot secrecy*, in "Journal of Computer Security", 2013, vol. 21, n^{o} 1, pp. 89–148

[38] V. CORTIER, G. STEEL, C. WIEDLING. *Revoke and let live: a secure key revocation api for cryptographic devices*, in "ACM Conference on Computer and Communications Security (CCS'12)", 2012, pp. 918-928

[39] S. DELAUNE, S. KREMER, G. STEEL. *Formal Analysis of PKCS#11 and Proprietary Extensions*, in "Journal of Computer Security", November 2010, vol. 18, n^{o} 6, pp. 1211-1245

[40] S. KREMER, R. KÜNNEMANN, G. STEEL. *Universally Composable Key-Management*, in "Proceedings of the 18th European Symposium on Research in Computer Security (ESORICS'13)", Egham, U.K., J. CRAMPTON, S. JAJODIA, K. MAYES (editors), Lecture Notes in Computer Science, Springer, September 2013, vol. 8134, pp. 327-344

[41] A. PIRONTI, R. SISTO. *Provably Correct Java Implementations of Spi Calculus Security Protocols Specifications*, in "Computers & Security", 2010, vol. 29, pp. 302 - 314

[42] B. SMYTH, D. BERNHARD. *Ballot secrecy and ballot independence coincide*, in "ESORICS'13: 18th European Symposium on Research in Computer Security", LNCS, Springer, 2013, vol. 8134, pp. 463-480, http://www.bensmyth.com/publications/2013-ballot-independence-for-election-schemes/

[43] N. SWAMY, J. CHEN, C. FOURNET, P.-Y. STRUB, K. BHARGAVAN, J. YANG. *Secure distributed programming with value-dependent types*, in "16th ACM SIGPLAN international conference on Functional Programming", 2011, pp. 266-278

[44] N. SWAMY, J. CHEN, C. FOURNET, P.-Y. STRUB, K. BHARGAVAN, J. YANG. *Secure distributed programming with value-dependent types*, in "J. Funct. Program.", 2013, vol. 23, n^{o} 4, pp. 402-451

Publications of the year

Doctoral Dissertations and Habilitation Theses

[45] M. PAIOLA. *Verification of cryptographic protocols with lists of unbounded lengths*, Université Paris-Diderot (Paris 7), May 2014, https://hal.inria.fr/tel-01103104

Articles in International Peer-Reviewed Journals

[46] M. BACKES, C. HRIŢCU, M. MAFFEI. *Union, Intersection, and Refinement Types and Reasoning About Type Disjointness for Secure Protocol Implementations*, in "Journal of Computer Security", 2014, vol. 22, n^{o} 2, pp. 301-353 [*DOI :* 10.3233/JCS-130493], https://hal.inria.fr/hal-01102192

[47] C. BANSAL, K. BHARGAVAN, A. DELIGNAT-LAVAUD, S. MAFFEIS. *Discovering concrete attacks on website authorization by formal analysis*, in "Journal of Computer Security", 2014, vol. 22, n^{o} 4, pp. 601-657 [*DOI :* 10.3233/JCS-140503], https://hal.inria.fr/hal-01102202

[48] D. CADÉ, B. BLANCHET. *Proved Generation of Implementations from Computationally Secure Protocol Specifications*, in "Journal of Computer Security", 2015, 135 p. , forthcoming, https://hal.inria.fr/hal-01102382

[49] V. CORTIER, G. STEEL. *A Generic Security API for Symmetric Key Management on Cryptographic Devices*, in "Information and Computation", 2014, vol. 238, 25 p. , https://hal.inria.fr/hal-00881072

International Conferences with Proceedings

[50] G. BANA, H. COMON-LUNDH. *A Computationally Complete Symbolic Attacker for Equivalence Properties*, in "2014 ACM SIGSAC Conference on Computer and Communications Security", Scottsdale, United States, ACM, November 2014, pp. 609-620 [*DOI :* 10.1145/2660267.2660276], https://hal.inria.fr/hal-01102216

[51] E.-I. BARTZIA, P.-Y. STRUB. *A Formal Library for Elliptic Curves in the Coq Proof Assistant*, in "Interactive Theorem Proving", Vienna, Austria, G. KLEIN, R. GAMBOA (editors), Lecture Notes in Computer Science, Springer, July 2014, vol. 8558, pp. 77-92 [*DOI :* 10.1007/978-3-319-08970-6_6], https://hal.inria.fr/hal-01102288

[52] B. BEURDOUCHE, K. BHARGAVAN, A. DELIGNAT-LAVAUD, C. FOURNET, M. KOHLWEISS, A. PIRONTI, P.-Y. STRUB, J. K. ZINZINDOHOUE. *A Messy State of the Union: Taming the Composite State Machines of TLS*, in "IEEE Symposium on Security & Privacy 2015", San Jose, United States, IEEE, May 2015, forthcoming, https://hal.inria.fr/hal-01114250

[53] K. BHARGAVAN, A. DELIGNAT-LAVAUD, C. FOURNET, A. PIRONTI, P.-Y. STRUB. *Triple Handshakes and Cookie Cutters: Breaking and Fixing Authentication over TLS*, in "IEEE Symposium on Security & Privacy", San Jose, United States, IEEE, April 2014 [*DOI :* 10.1109/SP.2014.14], https://hal.inria.fr/hal-01102259

[54] K. BHARGAVAN, A. DELIGNAT-LAVAUD, A. PIRONTI. *Verified Contributive Channel Bindings for Compound Authentication*, in "Network and Distributed System Security Symposium (NDSS'15)", San Diego, United States, February 2015, forthcoming, https://hal.inria.fr/hal-01114248

[55] K. BHARGAVAN, C. FOURNET, M. KOHLWEISS, A. PIRONTI, P.-Y. STRUB, S. ZANELLA-BÉGUELIN. *Proving the TLS Handshake Secure (As It Is)*, in "CRYPTO 2014", Santa Barbara, United States, J. A. GARAY, R. GENNARO (editors), Lecture Notes in Computer Science, Springer, August 2014, vol. 8617, pp. 235-255 [*DOI :* 10.1007/978-3-662-44381-1_14], https://hal.inria.fr/hal-01102229

[56] A. DELIGNAT-LAVAUD, M. ABADI, M. BIRRELL, I. MIRONOV, T. WOBBER, Y. XIE. *Web PKI: Closing the Gap between Guidelines and Practices*, in "Network and Distributed System Security Symposium", San Diego, United States, Internet Society, February 2014 [*DOI :* 10.14722/NDSS.2014.23305], https://hal.inria.fr/hal-01102254

[57] A. DELIGNAT-LAVAUD, K. BHARGAVAN. *Network-based Origin Confusion Attacks against HTTPS Virtual Hosting*, in "24th International Conference on World Wide Web", Florence, Italy, ACM, May 2015, forthcoming, https://hal.inria.fr/hal-01114246

[58] U. DHAWAN, C. HRITCU, R. RUBIN, N. VASILAKIS, S. CHIRICESCU, J. M. SMITH, J. T. F. KNIGHT, B. C. PIERCE, A. DEHON. *Architectural Support for Software-Defined Metadata Processing*, in "20th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2015)", Istanbul, Turkey, ACM, March 2015, forthcoming, https://hal.inria.fr/hal-01102378

[59] T. GAUTHIER, C. KALISZYK, C. KELLER, M. NORRISH. *Beagle as a HOL4 external ATP method*, in "PAAR - Fourth Workshop on Practical Aspects of Automated Reasoning", Vienne, Austria, July 2014, https://hal.inria.fr/hal-01089316

[60] S. KREMER, R. KÜNNEMANN. *Automated Analysis of Security Protocols with Global State*, in "35th IEEE Symposium on Security and Privacy (S&P'14)", San Jose, United States, I. C. SOCIETY (editor), 2014, pp. 163–178 [*DOI :* 10.1109/SP.2014.18], https://hal.inria.fr/hal-01091241

[61] A. MCCARTHY, B. SMYTH, E. A. QUAGLIA. *Hawk and Aucitas: e-Auction Schemes from the Helios and Civitas e-Voting Schemes*, in "FC 2014 - Financial Cryptography and Data Security", Christ Church, Barbados, N. CHRISTIN, R. SAFAVI-NAINI (editors), LNCS - Lecture Notes in Computer Science, Springer, March 2014, vol. 8437, pp. 51-63 [*DOI :* 10.1007/978-3-662-45472-5_4], https://hal.inria.fr/hal-01102159

[62] N. SWAMY, C. FOURNET, A. RASTOGI, K. BHARGAVAN, J. CHEN, P.-Y. STRUB, G. BIERMAN. *Gradual Typing Embedded Securely in JavaScript*, in "POPL 2014 - 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages", San Diego, CA, United States, ACM, January 2014, pp. 425-437 [*DOI :* 10.1145/2535838.2535889], https://hal.inria.fr/hal-00940836

[63] A. A. DE AMORIM, N. COLLINS, A. DEHON, D. DEMANGE, C. HRITCU, D. PICHARDIE, B. C. PIERCE, R. POLLACK, A. TOLMACH. *A Verified Information-Flow Architecture*, in "41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)", San Diego, CA, United States, 2014 [*DOI :* 10.1145/2535838.2535839], https://hal.inria.fr/hal-00918847

National Conferences with Proceedings

[64] G. BANA, K. HASEBE, M. OKADA. *Computationally Complete Symbolic Adversary and Key Exchange*, in "31st Symposium on Cryptography and Information Security", Kagoshima, Japan, 2014, https://hal.inria.fr/hal-01102293

Scientific Books (or Scientific Book chapters)

[65] K. BHARGAVAN, A. DELIGNAT-LAVAUD, S. MAFFEIS. *Defensive JavaScript*, in "Foundations of Security Analysis and Design VII", A. ALDINI, J. LOPEZ, F. MARTINELLI (editors), Lecture Notes in Computer Science, Springer, 2014, vol. 8604, pp. 88-123 [*DOI :* 10.1007/978-3-319-10082-1_4], https://hal.inria.fr/hal-01102144

[66] B. BLANCHET. *Automatic Verification of Security Protocols in the Symbolic Model: The Verifier ProVerif*, in "Foundations of Security Analysis and Design VII", A. ALDINI, J. LOPEZ, F. MARTINELLI (editors), Lecture Notes in Computer Science, Springer, 2014, vol. 8604, pp. 54-87 [*DOI :* 10.1007/978-3-319-10082-1_3], https://hal.inria.fr/hal-01102136

Research Reports

[67] K. BHARGAVAN, C. FOURNET, M. KOHLWEISS, A. PIRONTI, P.-Y. STRUB, S. ZANELLA-BÉGUELIN. *Proving the TLS Handshake Secure (as it is)*, Cryptology ePrint Archive, March 2014, n^{o} 2014/182, 48 p. , https://hal.inria.fr/hal-01102231

[68] C. HRITCU, L. LAMPROPOULOS, A. SPECTOR-ZABUSKY, A. A. DE AMORIM, M. DÉNÈS, J. HUGHES, B. C. PIERCE, D. VYTINIOTIS. *Testing Noninterference, Quickly*, arXiv, September 2014, n^{o} arXiv:1409.0393, 50 p. , https://hal.inria.fr/hal-01102224

[69] B. SMYTH, D. BERNHARD. *Ballot secrecy with malicious bulletin boards*, Cryptology ePrint Archive, October 2014, n^{o} 2014/822, 7 p. , https://hal.inria.fr/hal-01102306

[70] B. SMYTH, A. PIRONTI. *Truncating TLS Connections to Violate Beliefs in Web Applications*, Inria Paris, October 2014, This document extends https://hal.inria.fr/hal-00863371v1, https://hal.inria.fr/hal-01102013

References in notes

[71] M. ABADI, B. BLANCHET. *Analyzing Security Protocols with Secrecy Types and Logic Programs*, in "Journal of the ACM", January 2005, vol. 52, n^{o} 1, pp. 102–146

[72] M. ABADI, B. BLANCHET, C. FOURNET. *Just Fast Keying in the Pi Calculus*, in "ACM Transactions on Information and System Security (TISSEC)", July 2007, vol. 10, n^{o} 3, pp. 1–59

[73] J. BENGTSON, K. BHARGAVAN, C. FOURNET, A. D. GORDON, S. MAFFEIS. *Refinement types for secure implementations*, in "ACM Trans. Program. Lang. Syst.", 2011, vol. 33, n^{o} 2, 8 p.

[74] K. BHARGAVAN, A. DELIGNAT-LAVAUD, S. MAFFEIS. *Language-Based Defenses Against Untrusted Browser Origins*, in "Proceedings of the 22th USENIX Security Symposium", 2013

[75] K. BHARGAVAN, C. FOURNET, R. CORIN, E. ZALINESCU. *Verified Cryptographic Implementations for TLS*, in "ACM Transactions Inf. Syst. Secur.", March 2012, vol. 15, n^{o} 1, 3:1 p.

[76] K. BHARGAVAN, C. FOURNET, A. D. GORDON. *Modular Verification of Security Protocol Code by Typing*, in "ACM Symposium on Principles of Programming Languages (POPL'10)", 2010, pp. 445–456

[77] K. BHARGAVAN, C. FOURNET, A. D. GORDON, N. SWAMY. *Verified Implementations of the Information Card Federated Identity-Management Protocol*, in "Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS'08)", ACM Press, 2008, pp. 123–135

[78] B. BLANCHET, M. ABADI, C. FOURNET. *Automated Verification of Selected Equivalences for Security Protocols*, in "Journal of Logic and Algebraic Programming", February–March 2008, vol. 75, n^{o} 1, pp. 3–51

[79] B. BLANCHET. *An Efficient Cryptographic Protocol Verifier Based on Prolog Rules*, in "14th IEEE Computer Security Foundations Workshop (CSFW'01)", 2001, pp. 82–96

[80] B. BLANCHET. *Automatic Verification of Correspondences for Security Protocols*, in "Journal of Computer Security", July 2009, vol. 17, n^{o} 4, pp. 363–434

[81] B. BLANCHET, A. PODELSKI. *Verification of Cryptographic Protocols: Tagging Enforces Termination*, in "Theoretical Computer Science", March 2005, vol. 333, n^{o} 1-2, pp. 67–90, Special issue FoSSaCS'03

[82] J. CLULOW. *On the Security of PKCS#11*, in "CHES", 2003, pp. 411-425

[83] S. DELAUNE, S. KREMER, G. STEEL. *Formal Analysis of PKCS#11 and Proprietary Extensions*, in "Journal of Computer Security", November 2010, vol. 18, n^{o} 6, pp. 1211-1245

[84] D. DOLEV, A. YAO. *On the security of public key protocols*, in "IEEE Transactions on Information Theory", 1983, vol. IT–29, n^{o} 2, pp. 198–208

[85] C. FOURNET, M. KOHLWEISS, P.-Y. STRUB. *Modular Code-Based Cryptographic Verification*, in "ACM Conference on Computer and Communications Security", 2011

[86] R. NEEDHAM, M. SCHROEDER. *Using encryption for authentication in large networks of computers*, in "Communications of the ACM", 1978, vol. 21, n^{o} 12, pp. 993–999

[87] N. SWAMY, J. CHEN, C. FOURNET, P.-Y. STRUB, K. BHARGAVAN, J. YANG. *Secure distributed programming with value-dependent types*, in "16th ACM SIGPLAN international conference on Functional Programming", 2011, pp. 266-278

[88] N. SWAMY, J. CHEN, C. FOURNET, P.-Y. STRUB, K. BHARGAVAN, J. YANG. *Secure distributed programming with value-dependent types*, in "J. Funct. Program.", 2013, vol. 23, n^{o} 4, pp. 402-451