INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team SECSI

Sécurité des systèmes d'information

Futurs

THEME 2A

Table of contents

1. Team

SECSI is a project common to INRIA and the Laboratoire Spécification et Vérification (LSV), itself a common lab between CNRS (UMR 8643) and the École Normale Supérieure (ENS) de Cachan.

Head of project-team
Jean Goubault-Larrecq [Full Professor, ENS Cachan]

Vice-head of project-team
Hubert Comon-Lundh [Full Professor, ENS Cachan]

Staff member, INRIA
Florent Jacquemard [CR]
Julien Olivain [Engineer on RNTL project DICO until Nov. 30, ingénieur associé (engineer) since Dec. 01]

Staff member, CNRS
Stéphane Demri [CR]

Staff member, ENS Cachan
Ralf Treinen [Maître de conférences (associate professor)]
Fabrice Parrennes [Postdoc on ACI "jeunes chercheurs" funding until Aug. 31, 1/2 ATER (Part-time teaching and research assistant) since Sep. 01]

PhD students
Mathieu Baudet [Corps des Télécoms INRIA grant, since July 01]
Vincent Bernat [student at ENS Cachan]
Alexandre Boisseau [AMN grant, École Doctorale Sciences Pratiques (Cachan), until Sep. 30]
Véronique Cortier [AMN grant, École Doctorale Sciences Pratiques (Cachan), until Sep. 30]
Stéphanie Delaune [CIFRE grant with France Télécom R&D, École Doctorale Sciences Pratiques (Cachan), since Oct. 01]
Pascal Lafourcade [MENRT grant on ACI "sécurité" Rossignol, École Doctorale Sciences Pratiques (Cachan) & Université de Provence (Marseilles), since Oct. 01]
Muriel Roger [MENRT grant, École Doctorale Sciences Pratiques (Cachan), until Sep. 30]
Kumar Neeraj Verma [MENRT grant, École Doctorale Sciences Pratiques (Cachan), until Sep. 30]
Yu Zhang [MENRT grant on ACI "cryptologie" funding, École Doctorale Sciences Pratiques (Cachan)]

2. Overall Objectives

SECSI is a common project between INRIA Futurs and the LSV (Laboratoire Spécification et Vérification), itself a common research unit of CNRS (UMR 8643) and the ENS (École Normale Supérieure) de Cachan.

The SECSI project is a research project on the security of information systems. It is organized around three main themes, and their mutual relationships:

- Automated verification of cryptographic protocols;
- Intrusion detection;
- Static analysis of programs, in order to detect security holes and vulnerabilities at the protocol level.

The objectives of the SECSI project are:

- to design new models and new logics for describing security properties: secrecy, authentication, anonymity, privacy, fair exchange, resistance to dictionary attacks, etc;
- to design and implement new automated cryptographic protocol verification algorithms;
- to invent, improve, implement and experiment with new model-checking techniques, particularly on-line model-checking techniques, with application to intrusion detection;

- to design and implement new static analysis techniques to evaluate the level of assurance of actual cryptographic code;
- to integrate static analysis techniques and dynamic monitoring techniques (intrusion detection).

3. Scientific Foundations

3.1. What is computer security? Do we need some?

Key words: *computer security*, *verification*, *cryptographic protocol*, *static analysis*, *intrusion detection*, *model-checking*.

Glossary

verification see model-checking.

model-checking a set of automated techniques aiming at ensuring that a formal model of some given computer system satisfies a given specification, typically written as a formula in some adequate logic.

protocol a sequence of messages defining an interaction between two or more machines, programs, or people.

cryptographic protocol a protocol using cryptographic means, in particular encryption, that attempts to satisfy properties of secrecy, authentication, or other security properties.

static analysis set of automated techniques that determine some properties satisfied by given programs, without having to execute them; based on analyzing source code, sometimes object code; essentially identical to abstract interpretation of programs.

intrusion detection set of methods attempting to detect attacks, intrusions, or anomalies in computer systems, by real-time monitoring networks and systems.

Security has been getting more and more attention recently, as attacks against even personal computers (viruses, worms, spam), or banking cards, or mobile phones, etc., are becoming more and more frequent, and more and more well-known to the general public.

The first and foremost property that one would like to enforce is *secrecy*, or *confidentiality*. You certainly would not like to be robbed by somebody who got hold of all the necessary information on your banking card; you would not like your health record to be public either; and you would not like your next (hopefully) big-selling software project to be known by your competitors in advance. This problem, ensuring that some given data are concealed to external, non-authorized people (or machines), is not new. Encryption has been used as a means of ensuring confidentiality in every armed forces around the world for ages. The new factor here is that computers and networks make it so easy to access any kind of information: in modern computer networks, reading data from your computer for an intruder may be just as easy as connecting a wire to an outlet on the wall.

A second property of interest is *authentication*. Maybe you'd like to communicate with trusted parties. But how can you be sure you're really talking to the right person? A long time ago, when you met face to face, it was easy enough to recognize whom you were talking to. Nowadays, computers talk through digital lines. Even payphones talk to smartcards (see [104] for an authentication attack on second-generation pay-phone cards), and mobile phones talk to servers and back, using encrypted channels. Each of these appliances need to check that they are really talking to the right appliance or computer. Otherwise you could spy on someone else's conversation on the phone, or you could intercept an encrypted email between two competitors, for example.

There are many other properties to be checked, in practice. *Denial of service* attacks do not steal valuable information from your hard disk (secrecy does not fail), they do not attempt at making you believe you're

receiving an email from your old friend Joa (authentication), rather they just make your machine unusable: suddenly your machine freezes, reboots, your network is overloaded: you may be victim of a denial of service attack.

Another one is *fair exchange*: when you sign a contract over Internet—and you do, as soon as you buy a train ticket or the latest Harry Potter book on the Internet—, you would like to be sure that you agree to buy and the reseller agrees to sell, or none of you agrees to the transaction, but that nothing else may happen. In particular, you would like to be sure that nobody can get a competitive advantage by first having the other agree to the transaction, then reporting the sales condition you obtained to a competitor, to eventually resign the transaction and make a deal with the competitor.

There would be many other properties that are worth considering. The goal of the SECSI project is, foremost, to design algorithms and tools to check such *security properties*. First, on abstract and idealized versions of what actually runs on your computer, banking card, or mobile phone: namely, on *cryptographic protocols*. This is important: one can cite dozens of published cryptographic protocols which nonetheless have been found faulty later on—the award certainly going to the Needham-Schroeder public-key protocol [85], which was believed to be correct for 17 years before an attack was found, and the protocol fixed, by G. Lowe [81].

Second, it would be desirable to check the same security properties on more and more concrete algorithms, until a level is reached where actual code can be analyzed. This is a technical challenge, involving the design of new static analysis techniques that mix reasoning on cryptographic protocols (only at a larger scale) and reasoning on pointers, functions, and other features of standard programming languages.

Third, once various more or less abstract versions of some piece of software have been proved correct, it may still be the case that some attacks remain. This may sound like a paradox, but look at it this way. When we reason on an abstract version of the given piece of software, we may have forgotten some important aspects of reality in the model. For instance, we may have modeled possible intruders on our system as being dishonest, all other participants being honest; but Lowe's attack on Needham-Schroeder's public-key protocol involves an intruder that is both honest *and* dishonest at the same time (in different sessions). It is all too easy to overlook the fact that anybody might be both good and evil. Another example is the fact that, to be able to say anything at all on a protocol, or some piece of code, simplifying assumptions have to be made. For example, a very convenient assumption until now was that of *perfect cryptography*, where the only way to get the plaintext from the ciphertext is to decipher the latter, using the right key. But many cryptographic primitives are not perfect, and A. Joux [80] has shown that Lowe's corrected version of the Needham-Schroeder public-key protocol was in fact flawed again, if you used the El Gamal encryption scheme to encrypt messages.

One of our efforts in the themes of cryptographic protocol verification, and also static code analysis to a lesser extent, is to take into account such weaknesses in the models, and repair them. This will provide us with more and more reliable security assessment tools.

However, there will always remain something that the models overlook. To take a last example, consider static analysis of code. When one analyzes actual programs, it is useful to simplify the semantics of the analyzed programming language, and e.g., assume that no pointer runs wild; otherwise, basically the analyzer must assume that anything may happen, and will more often than not that the analyzed program is probably vulnerable—even when it is not (in the given model, of course!). It is then fair to assume that some other means is used to ensure that no pointer indeed goes wild ([95] is a good start), and voila, we don't have to care about out-of-bounds access to arrays and records. In the present case, ignoring out-of-bounds accesses through pointers is precisely what makes the so-called *buffer-overflow attacks* so easy [66]. Let us say right away that the great majority of viruses, worms, and trojans propagate through such buffer-overflow vulnerabilities. It is therefore definitely relevant to monitor system activity *in real time* to detect and counter such attacks. The SECSI project team has had some preliminary success in doing so as part in 2003, using a new intrusion detection tool developed at LSV/SECSI and based on a novel approach to *on-line model-checking*: the attack is detected and reported in real-time, the sessions of the offender are killed and his account closed, in a jiffy. This is just an example of what can be achieved through intrusion detection, and this technology has already been applied to other system-level, and network-level security issues.

While SECSI is interested in many aspects of computer security, no cryptology per se is being done at SECSI. This is better left to cryptologists. SECSI does not guarantee either that your system can be made absolutely secure. After all, one of the most reliable source of unauthorized access to information is through *social engineering* (more or less subtle uses of the gullibility of people), against which science is impotent: see Mitnick and Simon's book [83].

To sum up, the focus of SECSI is on making small (PC) to large (mainframe) systems more secure, by checking once and for all (statically) security properties at a fairly abstract level, and going all the way to the concrete by monitoring (dynamically) security properties on actual computers and networks.

Scientifically, all themes are united by our reliance on rigorous approaches and logic: automated deduction, tree automata, abstract interpretation, model-checking.

3.2. Logic as a tool for assessing computer security

The various efforts of the SECSI team are united by the reliance on *logic* and rigorous methods. As already said in Section 3.1, SECSI does not do any cryptology per se.

As far as cryptographic protocol verification is concerned, one popular kind of model is that of Dolev and Yao (after [64], see [59] for a survey), where: the intruder can read and write on every communication channel, and in effect has full control over the network; the intruder may encrypt, decrypt, build and destruct pairs, as many times as it wishes; and, finally, cryptographic means are assumed to be *perfect*. The latter in particular means that the only way to compute the plaintext M from the ciphertext $\{M\}_K$ is to decrypt the latter using the inverse key K^{-1}. It also means that no ciphertext can be confused with any message that is not a ciphertext, and that $\{M\}_K = \{M'\}_{K'}$ implies $M = M'$ and $K = K'$. Thus, messages can be simply encoded as first-order terms, a fact which has been used by many authors.

This observation may be seen as the foundations for encoding cryptographic protocols in first-order logic [102][52]. Cryptographic protocols can also be analyzed using tree automata [58], as shown in [84][70], or using set constraints [57][47]. All these tools can be seen from an automated deduction perspective, as shown in [72][73]. Extensions to encryption primitives obeying algebraic laws are now being considered in the SECSI project, using deduction techniques modulo equational theories, as well as direct proof-theoretic techniques [60]. This is one of the themes of the RNTL project Prouvé.

Our work on intrusion detection also relies on logic. The crux of our method is a fast implementation of a fast algorithm for *on-line* model-checking of an application-specific temporal logic to *linear* Kripke models [93]. It also relies on specific *abstract interpretation* techniques to dramatically improve the speed of detection, by showing that certains threads waiting for specific sequences of events cannot succeed and therefore can be killed safely [75][71]. Of course, abstract interpretation is at the heart of our static analysis of C code project, too. In this framework, SECSI designs static analyses that generation sets of Horn clauses as constraints, which are then solved by automated deduction techniques... and this loops the loop.

Finally, it should be mentioned that SECSI also looks at alternative techniques. The most prominent is research conducted at LSV/SECSI on *logical relations* for λ-calculi enriched with primitives for fresh name creation, encryption and decryption, following Sumii and Pierce [98]. This is continuous work, started in [74] and pursued in [103]. The puzzling thing here is that the logical relations obtained there generalize the notion of *bisimulations* used in process algebra to a richer, higher-order framework.

4. Application Domains

4.1. Introduction

Key words: *smartcards*, *mobile phones*, *secure distributed architectures*, *SSL*, *TLS*, *security*, *intrusion detection*.

The application domains of SECSI cover a large part of computer security.

4.2. Cryptographic Protocols

Cryptographic protocols are used in more and more domains today, including smart card protocols, enterprise servers, railroad network architectures, secured distributed graphic user interfaces, mobile telephony, on-line banking, on-line merchant sites, pay-per-view video, etc. The SECSI project is not tied to any specific domain as far as cryptographic protocols are concerned. Our industrial partners in this domain are Trusted Logic S.A., France Télécom R&D, and CRIL Technology.

4.3. Static Analysis

Analyzing cryptographic protocols per se is fine, but a more realistic approach consists in analyzing actual code implementing specific roles of cryptographic protocols, such as `ssh` or `slogin`, which implement the SSL/TLS protocols [100] are are used on every personal computer running Unix today. SSL and TLS are, more widely, used in every Web browser today: as soon as you connect to a secured server, you are running SSL or TLS. Being able to analyze actual C implementations of these or similar protocols is a concrete application we would like to be able to deal with in the long term.

4.4. Intrusion Detection

Making sure that cryptographic protocols are secure is not enough to guarantee that your system is secure. In all these domains, and in general in every domain where you need to set up a computer or a computer network, intrusion detection is needed. A new application domain for intrusion detection is smartcard security. While intrusion detection, and in particular the kind addressed in SECSI, used to be impractical on smartcards, the amount of available memory has soared on modern smartcards, making our intrusion detection techniques attractive on small devices: banking cards perhaps, SIM cards in GSM mobile phones certainly.

Standard application domains include securing enterprise-wide networks, and telephony servers. Our industrial partners in this domain today are France Télécom R&D and Calyx/NetSecure, a small company specialized in intrusion detection solutions.

A slightly less standard application of our intrusion detection techniques is tracking, where the intrusion detection system is not used to detect attacks, but to sort clients' activities per client type/user preferences (e.g., in GSM user tracking, as done by GSM operators), or to sort hardware and software failures according to client, hardware type or brand in remote maintenance applications.

5. Software

5.1. Software Packages and Prototypes

The SECSI project started in 2002 with a relatively large software basis: tools to parse, translate, and verify cryptographic protocols which are part of the RNTL project EVA (including CPV, CPV2, Securify), a static analysis tool in the course of being developed (CSur), an intrusion detection tool (logWeaver). These programs were started before SECSI was created.

The SPORE Web page was new in 2002. It is a public and open repository of cryptographic protocols. Its purpose is to collect information on cryptographic protocols, their design, proofs, attacks, at the international level.

Since then, 2003 brought new developments. In intrusion detection, a completely new project has started, which benefited from the lessons learned in the DICO project (Section 8.1.2): faster, more versatile, the ORCHIDS intrusion detection system promises to become the most powerful intrusion detection system around. The CSur project went on, although some semantic problems delayed the completion of the first prototype. To support our work based on automated deduction and tree automata in cryptographic protocol verification and static analysis, two tools were created. The MOP modular platform allows one to experiment with new equational theories (and more), so as to enable testing new ideas quickly. And the H1 tool suite was

created to support the discovery for security proofs, to output corresponding formal proofs in the Coq proof assistant, and also to provide a suite of tools allowing one to manipulate tree automata automatically.

5.2. The EVA Architecture: the EVA library, eva2eva, eva2h1, eva2cpl, eva2tex

Participants: Florent Jacquemard [in charge], Jean Goubault-Larrecq, Véronique Cortier [until Sep. 30, 2003].

Key words: *EVA*, *cryptographic protocols*, *architecture*.

The EVA project (Explanation and Automated Verification of cryptographic protocols) is a project of the Réseau National des Technologies du Logiciel (RNTL), see Section 8.1.1. Its purpose is to create automated cryptographic protocol verification tools, and to extract explanations why they are secure (in case they are) from verification, as readable and independently checkable proofs.

As a piece of software, EVA consists of several tools. (The architecture changed a lot this year, under the impulse of Florent Jacquemard.) The core of EVA consists in the EVA library `evatrans.cma`, a 11 000 lines of OCaml code library for parsing, type-checking, and compiling cryptographic protocols in the LAEVA language (the input language of EVA) to internal data structures describing an operational model that tools like Verimag's Hermès, Véronique Cortier's Securify (SECSI), or Jean Goubault-Larrecq's `h1` (SECSI) can work on. An additional 200 line wrapper, `eva2eva`, allows one to see the result of the compilation in EVA-like readable syntax.

The high-level LAEVA notation focuses on the form of the messages exchanged during the protocol whereas the operational model describes the operations performed by the participants. Hence, roughly, the EVA compiler transforms a sequence of messages into a set of processes (or *programs*), one for each role of the protocol (user, server, signing authority, etc.) The implementability of protocols as well as the correctness of typing of messages is also checked during compilation.

For example, the Otway-Rees protocol [87] is written as follows in LAEVA. The part between braces is the actual specification of the protocol; the notation is standard, and should be readable by anyone in the field. The `session*` keyword states that we are interested in checking arbitrarily many sessions of this protocol in parallel. The `assume` lines mean that we assume that every principal other than `I` (the intruder) is honest, and that any two honest principals `Y` and `Z` share a key `shr (Y,Z)` that is initially secret (unknown to the intruder). The `claim` lines mean that we would like to verify a number of secrecy properties on shared keys. For example, the first one asks whether, at all times, if the `A` and `B` parameters of principal `A` (M`@`p denotes the value of message M as seen by principal p, and reads "M at p") denote honest agents, then the shared key between these agents is secret.

```
Otway_Rees
  parameter A, B, S : principal
  parameter fresh Kab : number
  parameter fresh N, fresh Na, fresh Nb : number
  constant  shr (principal,principal) : number secret
  alias Kas = shr(A,S)
  alias Kbs = shr(B,S)
  A knows A, B, Kas, N, Na
  B knows S, Kbs, Nb
  S knows S, shr, Kab
  {
    1. A -> B : N, A, B, {Na, N, A, B}_Kas
    2. B -> S : N, A, B, {Na, N, A, B}_Kas,
                         {Nb, N, A, B}_Kbs
    3. S -> B : N, {Na, Kab}_Kas, {Nb, Kab}_Kbs
    4. B -> A : N, {Na, Kab}_Kas
  }
```

```
session*
assume  Z != I => honest(Z) [ Z:principal ]
assume honest(Y), honest(Z) => secret(shr(Y,Z)) [ Y : principal, Z : principal ]
claim honest(A@A), honest(B@A) => secret(shr(A@A,B@A))
claim honest(A@B), honest(B@B) => secret(shr(A@B,B@B))
claim honest(A@A), honest(S@A) => secret(shr(A@A,S@A))
claim honest(A@B), honest(S@B) => secret(shr(A@B,S@B))
claim honest(B@A), honest(S@A) => secret(shr(B@A,S@A))
claim honest(B@B), honest(S@B) => secret(shr(B@B,S@B))
```

The `eva2eva` tool can then be used to show how this compiles internally.

The `eva2h1` tool instead converts the input protocol to a set of Horn clauses, in Prolog notation, or in TPTP format [99], a standard format used primarily for testing automated theorem provers, now become a de facto standard for writing first-order formulas. This can then be fed to an automated theorem prover like SPASS [101], Vampire [92], or Waldmeister [77] for example: if the automated theorem prover stops without finding a contradiction, then the protocol is secure. However, since first-order logic is undecidable, none of these automated theorem provers is guaranteed to terminate. It is therefore preferable to feed the output of `eva2h1` to `h1`, which was designed to terminate while losing as little information as possible on the initial protocol; `h1` is also designed so as to output a formal proof for the Coq proof assistant (developed in the LogiCal project). This allows for independent rechecking the security proofs found, in accordance with the goals of the EVA project.

Several utilities were written, based on the EVA library `evatrans.cma`. Apart from `eva2eva` and `eva2h1`, these utilities are comprised of: `eva2cpl` translates EVA specifications to a Lisp-like syntax called **CPL**, which is the input format of some tools of the RNTL EVA project; and `eva2tex`, which translates EVA specifications to LaTeX (in connection with SPORE, see Section 5.4).

The `evatrans.cma` library has also been used, out of the scope of the EVA project, to develop small prototypes that search forward exhaustively for attacks, starting with a finite set of participants), one prototype was in particular developed by Stéphanie Delaune during her DEA to validate her work (see Section 6.5).

5.3. The Securify Cryptographic Protocol Verification Tool

Participants: Véronique Cortier [in charge], Stéphanie Delaune.

The Securify tool is an implementation of an algorithm invented by Véronique Cortier as she was visiting Jon Millen at SRI in 2000, and which she pursued in the RNTL project EVA (Section 8.1.1). As such, this program was developed independently of SECSI.

This tool is devoted to proving secrecy properties of cryptographic protocols. It is described in [61], and is based on a paper presented at CSFW in 2001 [62].

This algorithm verifies secrecy in a very general setting: unbounded number of sessions, unbounded number of principals, symmetric or asymmetric keys. However, it assumes that messages are typed and does not handle composite keys.

The principle underlying this tool is to attempt to prove the protocol sound by induction on the protocol rules. So the algorithm verifies that none of the rules can compromise any secret. To this end, three basic tests allow one to conclude that the rule is sound. If all these tests fail, more information is searched backward, and the procedure is resumed.

The Securify tool is remarkably fast, already in the first version, as the following table shows.

A new version version of Securify was implemented in 2003, and is available at http://www.lsv.ens-cachan.fr/~cortier/EVA/securify2.tar.gz. This new version is capable of verifying more protocols, but is a bit slower. The changes have been proposed by Stéphanie Delaune during her DEA and implemented by Véronique Cortier.

Protocol	Proof	max # of "back" steps	Time ms.
Otway-Rees	Yes	0	0.35
Woo and Lam	Yes	0	0.11
Denning-Sacco	No	0	0.07
ISO Symmetric Key	Yes	0	0.15
Needham-Schroeder-Lowe	Yes	1	1.08
Needham-Schroeder public key	No	1	1.23
Wide-Mouthed-Frog (modified)	Yes	2	4.76
Kao-Chow	Yes	3	8.94
Yahalom	Yes	3	21.06

Figure 1.

5.4. SPORE: the Security Protocols Open Repository

Participants: Florent Jacquemard [correspondant], Ralf Treinen, Hubert Comon-Lundh, Alexandre Boisseau [until Sep. 30], Véronique Cortier [until Sep. 30], (non-exclusive list).

SPORE is a publicly accessible Web page (http://www.lsv.ens-cachan.fr/spore/). Its purpose is to provide anybody who wishes so a public repository of cryptographic protocoles, their various versions, the security properties that they have been claimed to satisfy, those that they genuinely satisfy and under which assumptions, and the known attacks against these protocols.

SPORE in particular contains a list of cryptographic protocoles identified by Trusted Logic S.A. in the course of the RNTL project EVA (Section 8.1.1).

A similar catalog had been published in 1997 by John Clark and Jeremy Jacob, in the second part of a widely distributed survey [56]. Notably, their catalog has often been used as a source of case studies for designers of automated cryptographic protocol verification tools.

The goal of the SPORE page is to continue Clark and Jacob's endeavor, first by updating the protocols in their survey, second by adding new entries. The whole repository is accessible on line, so as to cater for some interactivity with users and to promote its reusability by tool designers.

Each entry in the repository contains the description of one cryptographic protocol (in the semi-formal syntax of the reference paper [54], its claimed security properties, as well as comments and links to papers and pages about the protocol. References to works that propose formal proofs, or attacks, are given. The protocols of the repository can be downloaded in several formats, including printable ones (postscript, pdf) and text-only specifications of the sequence of messages defining the protocol.

The SPORE page contains about fifty protocols today. It was designed to be open: readers may comment on entries by email. More importantly, they may submit new protocols.

The new version of the EVA translator (Section 5.2) offers an option to translate protocol specifications in the LAEVA language into the format (LaTeX with special macros) used to store protocols in SPORE. This option considerably simplifies the synchronization between SPORE and the databases of case studies (in LAEVA language) developed in order to validate the verification tools in the EVA project, as well as future projects.

5.5. The MOP Modular Platform for Automated Theorem Proving

Participants: Muriel Roger [in charge, until Oct. 31, 2003], Jean Goubault-Larrecq.

It is often the case, and in particular when considering malleable encryption (e.g., RSA, Diffie-Hellman exchanges, various uses of exclusive-or) that one needs to experiment with reasoning modulo equational theories. Surprisingly, and although it has been well-known for thirty years how to apply resolution-based automated deduction techniques modulo equational theories [89], no tool exists that implements them.

Muriel Roger has implemented a modular prover based on ordered resolution with selection [49], called MOP, as part of her PhD work [94]. This is written in OCaml, and totals about 7 000 lines of code. The MOP prover is not so much a tool as it is a library. It provides a generic engine for ordered resolution with selection. To implement a resolution prover for first-order terms modulo a new equational theory, it suffices to create a new instance of the `TERM` signature, which must implement the chosen ordering, the chosen selection function, plus substitution application and a unification algorithm. Several instances of `TERM` are provided, among which a signature for plain first-order terms, one for rational terms, and one for a specific format of terms mixing first-order terms and terms built on a given associative-commutative symbol $\oplus$. The latter was in particular recently used to show automatically that the IKA.1 group key agreement protocol [97] was secure in the so-called *pure eavesdropper* model (where the intruder can only listen to channels), for up to 4 principals in the group; and that it was insecure in both the copycat model (where the intruder can also copy and divert messages) and the standard Dolev-Yao (where, additionally, the intruder can decrypt and encrypt at will). The latter was well-known [82]. These results was submitted to the special issue on processes and security of the journal of logic and algebraic programming.

The application domain of MOP naturally exceeds cryptographic protocol verification, and applies to any problem that can be coded as first-order clauses modulo some theory with finitary unification. A word of warning, though: MOP is not designed for speed, but so as to accomodate different theories as easily as possible. H1 (Section 5.6) is more specialized, but faster.

5.6. The H1 Tool Suite: h1, pl2tptp, auto2pl, pldet, autodot, tptpmorph, linauto

Participant: Jean Goubault-Larrecq [in charge].

The initial purpose of the `h1` tool is to decide Nielson, Nielson and Seidl's class $\mathcal{H}_1$ [86], as well as an automated abstraction engine that converts any clause set to one in $\mathcal{H}_1$. This was developed as part of the RNTL project EVA (Section 8.1.1): `h1` gets its input from EVA protocol specifications through the `eva2h1`package (see Section 5.2), then abstracts it to get a set of $\mathcal{H}_1$ clauses. Since $\mathcal{H}_1$ is decidable, `h1` always terminates. In case a contradiction is found, the `h1` proof is an indication of a plausible attack on the input protocol. In case no contradiction is found, then the input protocol is secure.

In accordance with goals of the EVA project, when no contradiction is found, `h1` is also able to produce a proof of security, in the form of an alternating tree automaton describing a finite model. The `h1` tool also model-checks the input clause set against the alternating tree automaton, and generates a Coq proof script automatically. This proof script can then be re-checked under the Coq proof assistant, so as to get a formal proof of security.

To give an idea of performance, here are a few figures, extracted from the forthcoming invited paper by Goubault-Larrecq [73]. On Needham and Schroeder's symmetric key protocol [85] (not Needham-Schroeder's public key protocol! there are two in this paper), a 28 clause problem, `h1` produces a 187 transition, 59 state alternating tree automaton as a security proof of the two claimed secrecy requirements that hold (over three). This takes 160 milliseconds on a 1.6 GHz Pentium IV laptop running Linux RedHat 2.4.20-24.7 with 512 Mb central memory. The corresponding finite model takes exponentially more space, and would not even fit in memory.

The `h1` tool then takes 180 milliseconds to check that the alternating tree automaton it just computed is indeed a model (a proof of security) of the input clauses, hence of the two valid secrecy claims on the protocol, and to output the corresponding Coq proof. This proof is 5 596 lines long, contains 865 auxiliary lemmas, and is checked by Coq in 17 seconds, using 51 Mb of memory.

A word of warning: `h1` is fast, but is currently limited to first-order reasoning, without taking into account any equational theory. For more versatility, but less speed, choose MOP (Section 5.5).

The `h1` program is just the keystone of the H1 *tool suite*. Since `h1` can also be understood as a finite tree automaton workbench ($\mathcal{H}_1$ clause sets can be seen as fancy extensions to alternating, two-way tree automata with equality tests, and always define regular tree languages), there was an opportunity to make `h1` the cornerstone of a more ambitious platform for handling rational tree languages.

While `h1` takes its input in TPTP format [99], just like MOP, (recall that `eva2h1` also outputs clauses in TPTP format, and notice that CSur does, too), `h1` outputs alternating tree automata in Prolog syntax, and—if asked so—the corresponding finite models (a.k.a., complete deterministic automata) in XML syntax. Various tools are provided in the H1 tool suite to convert between these formats: `pl2tptp` converts clause sets in Prolog notation to clause sets in TPTP format, while `auto2pl` converts XML deterministic tree automata to Prolog notation. The `auto2pl` utility also has an option to produce complements of automata; the combination of `h1` and `auto2pl` then allows one to decide sets of $\mathcal{H}_1$ clauses plus Prolog's stratified negation. (This negation is exactly automaton complementation.) The `pldet` utility determinizes alternating tree automata in Prolog notation, outputting deterministic tree automata in XML syntax.

The `autodot` utility converts deterministic tree automata in XML syntax to files in `dot`'s input format: `dot` is a publicly available graph layout engine. This allows one to visualize automata, at least small ones.

The `tptpmorph` utility computes the image of regular tree languages under term algebra homomorphisms. Additionally, `linauto` solves quantifier-free Presburger formulas, exploiting Comon and Boudet's efficient encoding of solutions to quantifier-free Presburger formulas into deterministic word automata [53]. Note that word automata are merely particular cases of tree automata. The `tptpmorph` utility can then in particular be used to apply projection morphisms, thus implementing existential quantification. That is, `linauto`, `tptpmorph`, and `auto2pl` together provide all needed tools to decide full Presburger arithmetic.

The H1 tool suite, consisting of all these utilities, is written in HimML (http://www.lsv.ens-cachan.fr/~goubault/himml-dwnld.html), a variant of the ML language with fast finite set and map operations, due to Jean Goubault-Larrecq. H1 consists of about 20 000 lines of HimML code.

5.7. The CSur Static Analysis Tool

Participants: Jean Goubault-Larrecq [in charge], Fabrice Parrennes, Mathieu Baudet, Julien Olivain.

Key words: *Static analysis*, *C*, *pointer analysis*, *cryptographic protocols*, *Horn clauses*.

CSur was started before SECSI was created, early 2002. It is developed by Fabrice Parrennes, once a postdoc on the ACI jeunes chercheurs "Sécurité informatique, protocoles cryptographiques et détection d'intrusions", attributed to Jean Goubault-Larrecq in 2001. Since September 01, 2003, Fabrice Parrennes has been part-time teaching assistant (1/2 ATER) at ENS Cachan.

CSur was initially the basis for a homework assignment to students of the static analysis course at DESS "Développement de logiciels sûrs", which counted as their final grade. This version of CSur was a static analyzer of C programs, written in OCaml, which detected arithmetic overflows, illegal memory accesses (array bounds overflows notably). It was written by Jean Goubault-Larrecq. Then, only specific modules were provided to students, the rest being only available to them in compiled form. The assignment was for students to rewrite the missing sources according to specification: see http://www.lsv.ens-cachan.fr/~goubault/Csur/csur.html.

This first version of CSur served as a foundation for the new CSur project, started in 2002 by Fabrice Parrennes under the direction of Jean Goubault-Larrecq. The new CSur is meant to detect cryptographic protocol-related vulnerabilities in C source code. It parses, analyzes C source files, then produces lists of Horn clauses that can be passed on to tools like SPASS [101] or H1 (Section 5.6) to detect plausible attacks, or better, to prove formally that the input C program is secure.

The basic idea is that Horn clauses can be used to represent both interaction with the network, in Dolev-Yao style, and to describe in-memory pointer aliasing. For example, calls to the `write`(2) primitive will trigger the generation of a clause of the form $\texttt{knows}(M) \Leftarrow \ldots$, where M is the message written to the network, and calls to the `read`(2) primitive triggers clauses $P(X) \Leftarrow \texttt{knows}(X)$, where `knows` is the predicate recognizing all messages known to a Dolev-Yao intruder, and which is axiomatized as in classical Dolev-Yao analysis of cryptographic protocols. On the other hand, points-to analyses [48][96] can also be recast as generating Horn clauses.

As an example of what CSur can analyze, see the following piece of C code. This is a working implementation of role A of the Needham-Schroeder public-key protocol, contributed by Julien Olivain and Fabrice

Parrennes. The specification of the actual protocol, in standard notation, is included in comments of the form `/***...***/` (right column), so as to let the reader appreciate the semantic gap between the (idealized) protocol and the C code that implements it.

```
#include <openssl/bn.h>
#include <openssl/rand.h>
#include "needham_type.h"
/* [Omitted other #includes] */
int main(int argc, char **argv) {
int conn_fd; // Communication socket.
char alice[30]; // A's name
int  alice_port;
char bob[30]; // B's name as seen from A.
int  bob_port;
// Temporaries:
unsigned char nonceA[16];
char temp[1024];
// Messages:
msg_t alice_mess_1; BIGNUM *cipher_1;
msg_t alice_mess_2; BIGNUM *cipher_2;
msg_t alice_mess_3; BIGNUM *cipher_3;
// Keys:
struct nskey_s *alice_key;
struct nskey_s *bob_key;
alice_key = malloc(sizeof(struct nskey_s));
bob_key   = malloc(sizeof(struct nskey_s));
/* [Omitted code] generate A's and B's
keys in alice_key and bob_key. */
// Initialization:
alice_port = PORT_ALICE;
strcpy(alice,"127.0.0.1");
strcpy(bob,argv[1]);
bob_port = atoi(argv[2]);
// Open connection to B:
conn_fd = connect_socket(bob, bob_port);
/*** 1. A -> B : {Na, A}_pub(B) ***/
// Create A's nonce Na:
RAND_bytes(nonceA, 16);
alice_mess_1.msg_type = MSG1;
alice_mess_1.msg.msg1.id = ALICE_ID;
memcpy(alice_mess_1.msg.msg1.nonce,
nonceA, sizeof(nonceA));
// Encrypt and get {Na, A}_pub(B)
cipher_1 = BN_new();
my_cypher(&alice_mess_1,bob_key,cipher_1);
write(conn_fd, BN_bn2hex(cipher_1), 128);
/*** 2. B -> A : {Na, Nb}_pub(A) ***/
cipher_2 = BN_new();
read(conn_fd, temp, 128);
BN_hex2bn(&cipher_2, temp);
// Decrypt and get Na, Nb:
```

```
my_decypher(cipher_2,alice_key,&alice_mess_2);
    // Check that Na is the expected one:
  if (strncmp(alice_mess_2.msg.msg2.nonce1,
                nonceA , sizeof(nonceA)))
                   exit (1);
     /*** 3. A -> B : {Nb}_pub(B) ***/
       alice_mess_3.msg_type = MSG3;
   memcpy(alice_mess_3.msg.msg3.nonce,
          alice_mess_2.msg.msg2.nonce2,
     sizeof(alice_mess_2.msg.msg2.nonce2));
          cipher_3 = BN_new();
my_cypher(&alice_mess_3,bob_key,cipher_3);
write(conn_fd, BN_bn2hex(cipher_3), 128);
               return 0;
                  }
```

The challenge here is in the subtle interaction between the Dolev-Yao world and the C pointer world. At the end of 2003, CSur is a functional prototype, but the abstract semantics has to be reengineered. (Once this is done, it will not be hard to implement the newer semantics. CSur was designed so as to be able to change easily the clause generation functions, without having to modify the rest of the analyzer.)

Let us talk a bit about the need for a new abstract semantics. The main point is that CSur currently uses a more or less standard points-to analysis. But, first, points-to analysis is not flow-sensitive enough: checks are made in any implementation of a cryptographic protocol that a message just received conforms to a specific format (see the `if (strncmp` test in the example above); the fact that these checks passed or failed is ignored in standard points-to analyses, but is required in CSur. Second, points-to analysis is not relational enough. To illustrate what this means, assume we know that the `a` field of some record x points to a, and the `b` field points to b, or that the `a` field points to b and the `b` field points to a. Standard points-to analyses just remember that the `a` field may point to a or b, and that the `b` field may also point to a or b. In particular, they do not exclude the impossible case where both fields point to a (or to b). This has to be corrected: it is often the case that freshness guarantees have to be checked (not in the example above, though, but certainly so in the Otway-Rees protocol, see example in Section 5.2); the way freshness is ensured is by having a record x containing a nonce field, say `a`, and a message field, say `b`, such that either `a` is fresh (equal to some expected value `na`) and the message is one that will be used later, or `a` is not fresh but we do not care about the message. If the freshness test for `a` passes, then we care about the message, and fortunately it is fresh; if it fails, then fortunately we do not care about the message.

This is work in progress, and is currently addressed in collaboration with Mathieu Baudet.

5.8. The ORCHIDS Intrusion Detection Tool

Participants: Julien Olivain [in charge], Jean Goubault-Larrecq, Stéphane Demri.

ORCHIDS is a new intrusion detection tool, capable of analyzing and correlating events over time, in real time. Its purpose is to detect, report, and take countermeasures against intruders in real time. The core of the engine is based on the algorithm in the second part of the paper by Muriel Roger and Jean Goubault-Larrecq [93], which had also been implemented in the logWeaver tool, a now defunct tool, proprietary to Bull S.A. and INRIA. Let us be clear: ORCHIDS is not a new version of logWeaver, it is an entirely new product, based on published ideas [93]. The precise algorithm is described in two reports [75][71].

Moreover, ORCHIDS is based on a fast virtual machine for a massively-forking virtual parallel machine, and uses a hierarchy of input modules to subscribe to, and parse incoming events, classified by input source and/or event format. A main event dispatcher reads from polled and real-time I/O, reads sequences of events in `syslog` format, `snare` (in its original text format and in a raw binary kernel structure format), `sunbsm`, `apache` and other various formats, coming from log files or directly through dedicated network connections,

and feeds the relevant events to the core engine through an event dispatcher. The core engine implements a search for matching *sequences* of events—not just individual events!—, by running a virtual machine on code compiled from high-level signature descriptions. ORCHIDS is able to do both system-level and network-based intrusion detection.

Plans for 2004 include integrating clock skew, timing differences between machines over a network, and the notion of precision of clocks in the way ORCHIDS handles timing; enriching the signature language, and the compiler accordingly, to handle aggregate events through interval logic [45]; evaluating performance and precision of detection thoroughly.

Most notably, ORCHIDS has recently been put to detect very subtle buffer overflows attacks. Most notably, ORCHIDS has recently been used to detect the race condition attack on the `ptrace()` system call of the Linux-2.4 kernel [91], using the `snare` input module. Detecting this attack involves recognizing that some user process called the `ptrace` primitive with arguments `PTRACE_GETREGS`, `PTRACE_POKE` and `PTRACE_CONT` in this order, not necessarily contiguously, on a kernel process. (This is used to insert a foreign piece of code, the *shellcode*, which will install a shell with root privileges for the offending user.) ORCHIDS then kills the offending user's session, and can be used to disable (or close) his account remotely, using a secure connection to the host (SSH or SSL). Or it can also report the sequence of actions that the inserted shellcode did, for further analysis. This is particular useful on honeypots.

Julien Olivain, the developer of ORCHIDS, was paid through a one-year contract as part of the RNTL DICO project (Section 8.1.2) until November 30, 2003. The november version of ORCHIDS was first submitted to the APP (program protection agency) end of november, through the CNRS, which manages the contracts of LSV for the period 2001–2005. From December 01, 2003, Julien Olivain is engineer (ingénieur associé) at INRIA, in the SECSI project.

6. New Results

6.1. Two Principals Are Enough

Participants: Véronique Cortier, Hubert Comon-Lundh.

This result, found in 2002, was published for the first time in 2003: given a very general model of cryptographic protocols, with fairly arbitrary security properties, it is shown that if any attack exists, then one already exists with only fairly few principals. To take the simplest case, if any attack exists on a given secrecy property, then one already exists with only 2 principals: one honest principal and one intruder. If the additional assumption is made that no honest principal may talk to itself, 2 principals may not be enough, but then $k + 1$ are, where k is the number of principals mentioned in the security property.

This justifies, and sharpens previous ad hoc techniques, where a protocol was estimated secure as soon as it was shown secure with a bounded number of principals. Finally, this is useful to reduce the complexity of models of cryptographic protocols.

This was presented at ESOP 2003 [16], and submitted to the Special Issue of the Journal of Science of Computer Programming (SCP) in extended form [17].

6.2. Algebraic Properties in Cryptographic Protocol Verification

Participants: Hubert Comon-Lundh, Véronique Cortier, Ralf Treinen, Kumar Neeraj Verma, Muriel Roger, Jean Goubault-Larrecq.

The usual Dolev-Yao model makes the so-called *perfect cryptography* assumption. This in particular means that the only way to compute the plaintext M from the ciphertext $\{M\}_K$ is to decrypt the latter using the inverse key K^{-1}. It also means that no ciphertext can be confused with any message that is not a ciphertext, and that $\{M\}_K = \{M'\}_{K'}$ implies $M = M'$ and $K = K'$. Thus, messages can be simply encoded as first-order terms, a fact which has been used by many authors, and no algebraic law, except the trivial one $M = M$, holds on messages.

While this is a fine assumption if one uses encryption algorithms such as DES, RC5 or IDEA for example, other operations definitely obey non-trivial algebraic laws. There are at least two important examples of this that have been examined at SECSI:

- Diffie-Hellman primitives [63]. While these are usually implemented through modular exponentiation, the general framework can be described by the use of one unary function e and an associative-commutative (AC), or even an Abelian group law $\oplus$ [76] [4] with unit 0. For example, the original Diffie-Hellman protocol for establishing a common secret key between two principals A and B has A send the message $e(N_a)$ to B, where N_a is a nonce (implemented as a fresh, random number) created by A, B send the message $e(N_b)$ to A, where N_b is another nonce created by B. Assuming that, once you know $e(M)$ and M', you may deduce $e(M \oplus M')$, both A and B can build the common secret key $e(N_a \oplus N_b)$. The standard implementation is by using numbers in $\mathbb{Z}/p\mathbb{Z}^*$ (p prime) for messages, letting $e(M)$ be g^M mod p for some primitive element (generator) of $\mathbb{Z}/p\mathbb{Z}^*$, and $\oplus$ be multiplication of exponents. Note that we may alternatively use the law of other groups, e.g., that of an elliptic curve on $\mathbb{Z}/p\mathbb{Z}$ for $\oplus$ instead.
  The Diffie-Hellman primitive is used not only in the famous Diffie-Hellman protocol, but also in the so-called Diffie-Hellman ephemeral key exchange, in El Gamal encryption and El Gamal signature [67], and in the group Diffie-Hellman key exchange.
- The bitwise exclusive or (xor) operation. This is used in many implementations of protocols. E.g., some protocols actually define the ciphertext $\{M\}_K$ as just the xor $M \oplus K$ of M and K. Bellare and Rogaway's OAEP scheme calls xor twice to generate ciphertexts. It is important to take the associativity, commutativity, unit and cancellation ($M \oplus M = 0$) properties of xor: A. Joux' attack on the (corrected) Needham-Schroeder-Lowe protocol with xor encryption [80] is an example of this.

Hubert Comon-Lundh and Vitaly Shmatikov [18] present decidability results for the verification of cryptographic protocols in the presence of equational theories corresponding to xor and Abelian groups. They extend the conventional Dolev-Yao model by permitting the intruder to exploit these properties. It is shown that the ground reachability problem is in NP for the extended intruder theories in the cases of xor and Abelian groups. This result follows from a normal proof theorem. This result is lifted to reachability (e.g., secrecy) analysis in the xor case, by defining a symbolic constraint system, in the case of finitely many sessions. This constraint system is decidable. The techniques rely in particular on an extension of combination algorithms for unification procedures. As a corollary, this enables automatic symbolic verification of cryptographic protocols employing xor for a fixed number of sessions.

Hubert Comon-Lundh and Véronique Cortier [15] consider a new extension of the Skolem class for first-order logic and prove its decidability by resolution techniques. This class has the equational theory of xor built in. This class is shown decidable by automated deduction (resolution) techniques. This provides another technique to verify cryptographic protocols with xor. This technique applies to an unbounded numbers of sessions, contrarily to the previous one, and works by approximating the set of possible traces, e.g., relaxing the nonce freshness assumption. This provides new decidability results for the xor case.

Kumar Neeraj Verma and Jean Goubault-Larrecq attacked the problem of algebraic properties in cryptographic protocols through the use of suitable clause sets, resembling in that the approach above. Again, properties are checked by approximating clause sets representing cryptographic protocols. Automated deduction techniques had been used in 2001 to get first decidability properties on the resulting clause sets in the associative-commutative case [76]. An application was given to modeling the group Diffie-Hellman key exchange protocol IKA.1. Kumar Neeraj Verma pursued the study in 2003 from a more automata-theoretic angle, providing for new decidability results for protocols based on Diffie-Hellman primitives, but also on the xor primitive and other primitives obeying certain equations that include associativity and commutativity. (See Section 6.3 for more details.)

A new abstraction technique was then developed by Jean Goubault-Larrecq and Muriel Roger [94], using automated deduction techniques and a novel automated abstraction technique based on the clever design of so-called light grey oracles. This was implemented in Muriel Roger's modular platform for automated theorem proving MOP [94], and used to give the first formal, fully automated verification of the IKA.1 distributed key exchange protocol in the so-called pure eavesdropper model: see Section 5.5.

Finally, some other algebraic properties have been investigated, too. Hubert Comon-Lundh and Ralf Treinen investigate extensions of the Dolev-Yao model, including some algebraic properties of cryptographic primitives [19]. They provide sufficient conditions under which the intruder deduction problem is decidable (resp. decidable in polynomial time). They apply this result to the equational theory of homomorphism, namely $\{M_1, M_2\}_K = \{M_1\}_K, \{M_2\}_K$, and show that in this case the intruder deduction problem is linear, provided that the messages are in normal form. The theory of homomorphism corresponds in practice to block encryption.

6.3. Closure Properties for Two-Way Alternating Tree Automata

Participants: Kumar Neeraj Verma, Jean Goubault-Larrecq.

Two-Way tree automata provide a suitable framework for modeling intruder knowledge in order to verify security properties of cryptographic protocols. In this approach terms are used to represent messages. Although this is sufficient when cryptographic primitives are perfect, actual protocols often use complex cryptographic primitives where distinct terms may represent the same message. Such properties can be modeled using an equational theory. In particular the theory of associativity-commutativity with unit (ACU), and its variants like the theory of Abelian groups (AG) and the theory of exclusive-or (XOR) are frequently used in cryptographic protocols. This necessitates studying two-way tree automata modulo these equational theories.

Kumar Neeraj Verma showed [27][4] that unlike in the non-equational case, emptiness of two-way tree automata modulo the above theories is undecidable in general. However under some suitable restrictions on the form of clauses used to describe the two-way automata, two-way tree automata modulo several theories including ACU, AG, xor, can be effectively reduced to equivalent one-way tree automata modulo the same theories. In addition, the one-way tree automata modulo these theories are shown to be closed under intersection and to have decidable emptiness problem. This implies that intersection emptiness of two-way tree automata modulo these equational theories is decidable, which is what is required for verifying secrecy properties of cryptographic protocols. He has also completely answered [26][4] the question of closure under complementation of one-way tree automata modulo these associative-commutative theories.

6.4. Extended Vector Addition Systems with States

Participants: Kumar Neeraj Verma, Jean Goubault-Larrecq.

To deal with certain classes of two-way equational tree automata, Kumar Neeraj Verma and Jean Goubault-Larrecq introduced [4] an extension of Vector Addition Systems with States (VASS), or equivalently, Petri Nets. This will be submitted in 2004.

These Extended VASS have addition transitions which allow one to merge two configurations. Runs in Extended VASS are branching tree-like structures instead of linear ones as in the case of VASS. They showed that the construction of Karp-Miller trees for VASS can be generalized to the case of Extended VASS. This implies that emptiness, boundedness and coverability of Extended VASS is decidable. While this was needed to solve some problems in two-way tree automata modulo associativity and commutativity, this might have a more general scope. Extended VASSes also arise naturally as a common generalization of Parikh images of context-free languages and of VASS.

6.5. A Theory of Dictionary Attacks

Participants: Hubert Comon-Lundh, Stéphanie Delaune, Florent Jacquemard.

During her DEA, under the supervision of Hubert Comon-Lundh and Florent Jacquemard, Stéphanie Delaune has proposed a theory of dictionary attacks on cryptographic protocols. Such attacks occur when an intruder is able to guess poorly chosen user passwords (or other data belonging to a reasonably small domain) by an offline brute force iteration through a dictionary, using messages previously collected on the network to verify his guesses at each step.

The theory is presented in [37] by an inference system modeling an intruder which extends the classical Dolev-Yao rules with guessing abilities. Using proof rewriting techniques, she shows a locality lemma for the inference system which yields the PTIME-completeness of the intruder deduction problem, i.e., whether the intruder can deduce some data from a given finite set of partially encrypted messages.

This result was presented in the Workshop on Security Protocols Verification (SPV'2003) [21], and was tested using a prototype for the exhaustive research of attacks (starting with a finite set of participants), implemented by Stéphanie Delaune.

Stéphanie Delaune and Florent Jacquemard next consider the more general problem of automating proofs of cryptographic protocols in presence of an intruder able to mount dictionary attacks. They lift the above result [37][21] to the simultaneous solving of intruder deduction constraints with variables. Constraint solving is the basis of a NP algorithm for the protocol insecurity problem in presence of guessing attacks, assuming a bounded number of sessions. This extends the classical NP-completeness result for the Dolev-Yao model. The procedure is illustrated with published protocols examples. This will be submitted in 2004.

6.6. Reducing Verification in Probabilistic Polynomial-Time Frameworks to Simple Reachability

Participant: Mathieu Baudet.

For several decades, two different communities have been working on the formal security of cryptographic protocols. Many efforts have been done recently to take benefit of both approaches, in brief: the comprehensiveness of computational models and the automatizability of formal methods.

In computational models, security is defined in a semantic way by requiring the probability of success of any random polynomial-time attacks to be negligible [68][69]. For encryption schemes, different security levels can be defined [51]: one-wayness (OW), indistinguishability of encryption (IND) and non-malleability (NM), whereas the considered attacks may use encryption oracles (chosen-plaintext attacks—CPA), plaintext-checking oracles (plaintext-checking attacks—PCA) or decryption oracles (chosen-ciphertext attacks—CCA). In this approach proofs of security consist in reducing any hypothetical attack to a random polynomial algorithm that solves a reputedly intractable problem. Some authors even strengthen this notion by requiring the reduction to be not only random polynomial but also practical [50][90]. Computational security proofs, when they can be achieved, are thus considered strong evidence of security.

A second class of models is used by the community of formal methods, and includes typically the Dolev-Yao model [64] and the spi-calculus [44]. By focusing on the protocol layer, these models aim to account for a variety of attacks resulting from complex interactions between an active attacker and a possibly unbounded number of parallel sessions. This is indeed a very hard task in the computational models, where already a passive attacker may lead to highly complex reduction proofs. Formal method models, notably extensions of the Dolev-Yao model, have proved all the more useful since, in many cases, they admit fully-automatic algorithms for the verification of protocols (this has been said enough times in the SECSI Activity Report 2003 already). This is certainly a salient feature in a world where new versions of protocols are published every day and cannot be all analyzed by specialists.

Motivated by these observations, efforts have been done recently to relate the two views of cryptography. Better understanding the links between the two approaches would indeed benefit both communities:

- For the formal method approaches, this would help providing more precise justifications and give directions for extending the expressivity of the models and the automatic analyzers.

- For the computational models, it would give elements toward partially automatizing the security proofs. One could imagine, e.g., proofs in two steps: the first would establish sufficient computational-security hypotheses on the cryptographic primitives, the second would deal with the protocol aspects by an automatic procedure.

Mathieu Baudet discovered a novel approach to relate these two views, namely to extend existing Dolev-Yao models to account for random polynomial-time (Las Vegas) computability. This is done first by noticing that Dolev-Yao models can be seen as transition systems, possibly infinite. These transition systems are then extended with computation times and probabilities. The extended models can account for normal Dolev-Yao transitions as well as nonstandard operations such as inverting a one-way function (cracking a key, e.g., by brute force). The main contribution of the report [29] (submitted) consists in showing that under sufficiently realistic assumptions the extended models are equivalent to standard Dolev-Yao models as far as security is concerned: checking that no attack can be perpetrated in random (Las Vegas) polynomial time reduces to simple graph reachability, hence to just verification in a run-of-the-mill Dolev-Yao model. Thus this work enlarges the scope of existing decision procedures.

6.7. Logical Relations for Name Creation and Encryption

Participants: David Nowak, Yu Zhang, Jean Goubault-Larrecq.

Pitts and Stark's nu-calculus [88] is a typed lambda-calculus which forms a basis for the study of interaction between higher-order functions and dynamically created names. A similar approach has received renewed attention recently through Sumii and Pierce's cryptographic lambda-calculus [98], an original approach to security protocols.

Logical relations are a powerful tool to prove properties of such a calculus, notably observational equivalence. While Pitts and Stark construct a logical relation for the nu-calculus, it rests heavily on operational aspects of the calculus and is hard to extend. David Nowak and Yu Zhang proposed an alternative Kripke logical relation for the nu-calculus, which is derived naturally from the categorical model of the nu-calculus and the general notion of Kripke logical relation. They show that their Kripke logical relation, which extends the definition of Goubault-Larrecq et al. [74], is equivalent to Pitts and Stark's up to first-order types; their definition rests on purely semantic constituents, and dispenses with the detours through operational semantics that Pitts and Stark use.

Next, it is shown that a simple trick allows one to get a logical relation for a richer lambda-calculus that also contains very general encryption and decryption mechanisms. This trick, which basically just consists in saying that logical relations should relate decryption primitives, and should relate encryption primitives, vastly generalizes Sumii and Pierce, thus providing natural behavioral equivalence checks for higher-order cryptographic languages. This was submitted to the special issue on processes and security of the journal of logic and algebraic programming.

6.8. Fair-Exchange Protocols

Participant: Alexandre Boisseau.

Alexandre Boisseau studied several approaches to abstraction in verification of cryptographic protocols in his PhD thesis [1]. One original theme is motivated by the problem of verifying fair-exchange protocols. The typical case is as follows. Assume two principals A and B, who wish to exchange signatures on a given contract text C. The simple solution is to make A sign C, then send the signed message $sig_A(C)$ to B, and to make B sign C and send the signed message $sig_B(C)$. However, assume A signs first. Once B receives $sig_A(B)$, B has got an unfair advantage over A. E.g., B may prove to a third person that A is ready to conclude a deal with him, without himself getting tied to the contract. *Fair-exchange protocols* aim to guarantee that neither A nor B can get such an unfair advantage over the other. This concerns not just contract-signing protocols, but also e-commerce, non-repudiation protocols, electronic mail with proof of receipt, notably.

Alexandre Boisseau, in collaboration with Steve Kremer and Jean-François Raskin (University of Brussels), defined a model of such protocols using alternating transition systems (ATS), originally introduced by Alur et al. [46]. He showed that the main properties to be proved (fairness, timeliness, viability, and abuse-freeness) can be defined in alternating time logic (ATL), also defined by Alur et al. The real advance came in 2003 when Alexandre Boisseau designed abstraction techniques that make model-checking ATL formulae on fair-exchange ATSes practical. This was applied successfully to the GJM contract-signing protocol [65].

An article is in preparation.

6.9. Checking Opacity Properties

Participants: Alexandre Boisseau, Jean Goubault-Larrecq.

Alexandre Boisseau studied several approaches to abstraction in verification of cryptographic protocols in his PhD thesis [1]. One class of properties that has little been studied is the family of *opacity* properties, foremost anonymity and privacy. In particular, it is now well-known that anonymity, privacy, and confidentiality are orthogonal concepts. In fact, opacity properties are very different from secrecy, and logically more complicated. In short, while secrecy states that some sensitive data cannot be accessed by an intruder, anonymity properties state that an external observer cannot make the difference between one principal or another doing some (possibly well-known) action.

Boisseau starts from a definition of these properties due to Hughes and Shmatikov [78], based on observational equivalence. This is already more complicated than standard uses of observational equivalence in secrecy: the goal is not to test observational equivalence, but to explore all processes of a certain form that are observationally equivalent to a given one. More precisely, given a set W of parameters that may vary, let $P(w)$ be a process parameterized by $w \in W$, and $k(w)$ some information that an external observer may be interested in. Opacity of k means that the equivalence relation induced by k on W contains the observational equivalence on processes $P(w)$, $w \in W$. Boisseau uses the finite spi-calculus [44] as a formalism for representing processes, exploiting that framed bisimilarity is decidable for this fragment of the spi-calculus [79]. He then proposes an abstraction for opacity properties. The challenge is to be able to enumerate quickly finite, large enough subsets of parameters $w' \in W$ such that $P(w')$ is framed-bisimilar (hence observationally equivalent) to $P(w)$. The resulting abstraction algorithm is then applied to Chaum's dining cryptographers problem [55] and to an election protocol.

8. Other Grants and Activities

8.1. National Actions

8.1.1. RNTL project EVA (preexisting SECSI)

Participants: Hubert Comon-Lundh, Jean Goubault-Larrecq, Florent Jacquemard, Véronique Cortier, Vincent Bernat, Alexandre Boisseau.

This exploratory project, funded by the national network for software technology (RNTL), is a collaboration between Trusted Logic S.A. (leader, INRIA startup, Versailles), the LSV (Cachan), and Verimag (Grenoble). It was notified in fall 2000, and ended in fall 2003.

The title, EVA, means **E**xplanation and **A**utomated **V**erification of cryptographic protocols. The purpose of this project is to develop verification techniques that are automated and not just computer-assisted of cryptographic protocols. The properties to check are various notions of confidentiality and authentication, but the property language is slightly richer. Stress is put on verification in parallel multi-session mode, i.e., where there are potentially unboundedly many principals obeying each role.

The **E** in EVA denotes explanation, and is an important part of this project. "Explanation" covers several research themes. One consists in extracting either a readable argument why the protocol is correct from a proof of the protocol, or a readable script of an attack (or of a trace indicating possible attacks). One of the successes

of EVA is to have fostered the development of tools that can provide proof scripts that a proof assistant like Coq (LogiCal project, INRIA) can check independently. Verimag's Hermès tool does it from a trace of its verification algorithm, and the `h1` tool at LSV (Section 5.6) does it from a model of the clauses output by the `eva2h1` tool (Section 5.2 from the given protocol specification.

See http://www-eva.imag.fr/index_eva.html for the EVA home page.

8.1.2. RNTL project DICO (preexisting SECSI)

Participants: Jean Goubault-Larrecq, Julien Olivain, Stéphane Demri.

This exploratory project, funded by the national network for software technology (RNTL), groups NetSecureOne (formerly Calyx/NetSecure, Maisons-Alfort, and before that NetSecure Software, Neuilly; leader of the project), France Telecom R&D (Caen), the LSV (Cachan), the IRISA (Rennes), ONERA/DTIM (Toulouse), FERIA/IRIT (Toulouse), and the École Supérieure d'Électricité (Rennes). It was notified in december 2001, and ended in december 2003; the project leader is waiting for an answer from the ministry of research and technology to a request to prolong the project until March 2004, for technical reasons.

The title, DICO, means **Co**operative **I**ntrusion **D**etection. It is therefore, first, an intrusion detection project, mixing signature-based approaches such as the one currently implemented in the ORCHIDS tool (Section 5.8) at LSV, behavior-based approaches (e.g., Bayesian networks), and alert correlation. The last item is the keystone of the project, and consists in merging alerts, or infering alerts coming from various tools, even from different servers. ORCHIDS also counts as one of the tools providing alerts in the DICO architecture.

8.1.3. ACI cryptologie "VERNAM" (preexisting SECSI)

Participants: Hubert Comon-Lundh, Jean Goubault-Larrecq, Ralf Treinen, Vincent Bernat, Véronique Cortier, Kumar Neeraj Verma.

The ACI "VERNAM" consists of the LIF (Laboratoire d'Informatique de Marseille), the LSV (Cachan), and the INRIA project PROTHEO, working on the search for decidable subclasses of cryptographic protocols, in relation with decidable subclasses of first-order logic. It is an ACI (action concertée incitative), on the theme of cryptology, of the ministry of research. It started in fall 2000, and ended in fall 2003.

8.1.4. ACI cryptologie "PSI-Robuste" (preexisting SECSI)

Participants: Jean Goubault-Larrecq, Stéphane Demri, Fabrice Parrennes, Julien Olivain, Muriel Roger, Yu Zhang.

The ACI "PSI-Robuste" is a crystallization action at LSV, on the theme of protecting computer systems. More specifically, it consists in developing new intrusion detection approaches, new static code analysis techniques aiming at detecting possible vulnerabilities, and making both interact. This is an ACI (action concertée incitative) on the theme of cryptology of the ministry of research. It started in fall 2001, for three years.

8.1.5. ACI jeunes chercheurs "Sécurité informatique, protocoles cryptographiques et détection d'intrusions" (preexisting SECSI)

Participants: Jean Goubault-Larrecq, Fabrice Parrennes, Stéphane Demri, Alexandre Boisseau, Véronique Cortier, Muriel Roger, Yu Zhang.

This is an ACI on the "jeunes chercheurs" programme ("young researcher"), attributed to Jean Goubault-Larrecq. This provides him and his colleagues funding for three years, starting from fall 2001.

The themes of this ACI are essentially the same as those of SECSI: automated cryptographic protocol verification, intrusion detection mainly, with a gist of static analysis. Fabrice Parrennes was funded through this ACI until September 2003, starting from September 2002. He is now part-time teaching assistant at ENS Cachan.

8.1.6. RNTL project Prouvé

Participants: Ralf Treinen [in charge], Hubert Comon-Lundh, Jean Goubault-Larrecq, Florent Jacquemard, Stéphanie Delaune, Pascal Lafourcade.

The exploratory project "Prouvé" was proposed to the national network for software technology (RNTL) in 2003, and was ranked among the first five projects (the first five projects obtained the same ranking). At the time of this writing the final approval by the RNTL is still pending. The partners of this project are CRIL Technology, France Télécom R&D (Lannion), the CASSIS project at INRIA Lorraine (Nancy), LSV (Cachan), and Verimag (Grenoble).

The Prouvé project (for "Protocoles cryptographiques: outils de vérification automatique", i.e., cryptographic protocols: automated verification tools) will be based on the foundations layed by the EVA project, which ended late 2003. It will address some questions left open in EVA:

- One of the goals of Prouvé is to define a semantics of cryptographic protocols that would be independent of the particular security property under consideration, and to define a language of security properties which would allow one to express all properties of interest, independently of the protocol studied.
- Another goal of the project is to extend the known methods of protocol verification by weakening the so-called perfect cryptography assumption. In particular, it should be possible to verify cryptographic protocols while taking into consideration algebraic properties of cryptographic primitives (such as those of modular arithmetic, as frequently used in public key cryptography), and substitution of nonces by timestamps or counters. Algebraic properties are already actively studied in the SECSI project, see Section 5.5 and Section 6.2.

Finally, the techniques to be developed in this project will be validated in case studies provided by one of the industrial partners of the project, France Télécom R&D.

8.1.7. ACI cryptologie "Rossignol"

Participants: Hubert Comon-Lundh [in charge], Pascal Lafourcade, Vincent Bernat, Stéphanie Delaune, Jean Goubault-Larrecq, Florent Jacquemard, Ralf Treinen.

The "Rossignol" project, submitted and accepted as an ACI sécurité informatique, started in december 2003. The partners of the projects are the LIF (Laboratoire d'Informatique Fondamentale de Marseille), the CoMeTe action of INRIA Futurs (Laboratoire d'Informatique de l'École Polytechnique, Saclay), the LSV (Cachan) and Verimag (Grenoble). All the participants at LSV are members of the SECSI project.

The goal of the project Rossignol is to create a framework for security protocol verification that takes into account the operational semantics of protocols, the theory of the intruder that defines the attackers capabilities, and the semantics of the intended properties, which will be defined independently from the description of the protocols using appropriate formalisms and logics. The project aims at fitting more closely to actual cryptographic protocol practice.

9. Dissemination

9.1. Teaching

Jean Goubault-Larrecq and Hubert Comon-Lundh gave a series of lectures on cryptographic protocol verification and automated deduction at the DEA "Programmation"; amount: 30 h. (TD equivalent), 15 h. each.

Ralf Treinen has given a 15 h. course (TD equivalent) in the module "Vérification de systèmes concurrents" of the DEA "Programmation".

Hubert Comon-Lundh and Ralf Treinen taught logic ("Logique") in the first term of the first year of the magistère STIC, ENS Cachan. Total volume 80 h. (TD equivalent). This course started in October 2002 and extended into the year 2003. They again gave this course from October 2003, extending into 2004, for another 80 h.

Hubert Comon-Lundh and Ralf Treinen taught computability 2 ("Calculabilité 2") in the second term, first year of the magistère STIC, ENS Cachan. Total volume 80 h. (TD equivalent).

Jean Goubault-Larrecq gave the first half of the module “Programmation 1” in the first term, first term of the magistère STIC, ENS Cachan. Total volume: 30 h. (TD equivalent).

Ralf Treinen taught one half of the module “Programmation 2” in the second term of the first year of the magistère STIC. Total volume: 40 h. (TD equivalent).

Fabrice Parrennes gave programming courses in the module “Programmation avancée”, second year of the magistère STIC, ENS Cachan. Volume: 96 h. for 2003-2004, of which 53 h. in 2003. The aim of the course is to program an application which uses system programming and concurrency programming.

Hubert Comon-Lundh and Ralf Treinen taught part of the module on logic and automata (“logique et automates”) of the first term of the second year of the magistère STIC, ENS Cachan. Total volume of the Comon-Lundh/Treinen part: 60 h. (TD equivalent). This course started in October 2003 and extends into the year 2004.

Jean Goubault-Larrecq gave the course on logic and computer science (“logique et informatique”), second term of first year, common to the magistère STIC, ENS Cachan, and the magistère de mathématiques fondamentales et appliquées à l’informatique (MMFAI), ENS (rue d’Ulm). Volume: 36 h. (TD equivalent).

Florent Jacquemard gave the TDs (exercise sessions) of the above course on logic and computer science. Volume: 24 h. (TD equivalent).

Jean Goubault-Larrecq gave the course on static code analysis and abstract interpretation at the DESS “développement de logiciels sûrs” (CNAM, Paris 7, ENS Cachan). Volume: 30 h. (TD equivalent).

Jean Goubault-Larrecq accompanied the students of ENS Cachan for a visit to the labs at INRIA Sophia-Antipolis, March 10.

As moniteur, Véronique Cortier gave TDs (exercise sessions) on computability to students of the magistère de Mathématiques et d’Informatique, ENS Cachan. Volume: 32 h. She also gave TPs (programming exercise sessions) to students of the magistère de génie électrique (electrical engineering), ENS Cachan. Volume: 32 h. (i.e., 21 h., TD equivalent).

Stéphane Demri gave lectures on “Algorithms: maximal flux problem, NP-completeness, and approximation algorithms”, Magistère STIC, ENS Cachan, 2002/2003. Volume: 15 h.

9.2. Scientific and Administrative Charges

Hubert Comon-Lundh organized, together with Ralf Treinen, the first meeting of Action Spécifique “Sécurité logicielle: modèles et vérification”, Cachan, February 2003, 35 participants. He is one of the persons in charge of this action.

Hubert Comon-Lundh is member of the scientific board of the Action Concertée Incitative (ACI) “Sécurité Informatique”, and member of the bureau.

Hubert Comon-Lundh is in charge of the computer science teaching department at ENS Cachan.

Hubert Comon-Lundh is member of the commission de spécialistes of University Paris 7, Section 27 (Computer Science). He participated at meetings on April 22 and May 6, 7, and 9, 2003.

Hubert Comon-Lundh is member of the scientific committee of the Laboratoire d’Informatique Algorithmique, Fondements et Applications (LIAFA), December 2003.

Ralf Treinen is supplementary member of the commission de spécialistes of University Lille 1, Section 27 (Computer Science). He participated at meetings on April 28, 2003 and May 16, 2003.

Jean Goubault-Larrecq participated in the interview process and the jury of admissibility of the young researcher (CR2) exam, INRIA Futurs, Paris, May 05–06 and June 05.

Jean Goubault-Larrecq was member of evaluation commission (commission d’évaluation) of the laboratory PPS (Preuves, Programmation, Systèmes), November 14.

9.3. Supervision, Advisorship

Jean Goubault-Larrecq supervised the following students:

- Kumar Neeraj Verma, PhD defended Sep. 30, 2003 at ENS Cachan [4];

- Muriel Roger, PhD defended Oct. 24, 2003, ENS Cachan [3];
- Alexandre Boisseau (together with Michel Bidoit), PhD defended Sep. 19, 2003 at ENS Cachan [1];
- Yu Zhang (together with David Nowak), second-year PhD student, working on verification of cryptographic protocols, the cryptographic λ-calculus, and logical relations;
- Mathieu Baudet, first year PhD student (since July 01, 2003), working on cryptographic protocol verification and static code analysis.

David Nowak supervised Yu Zhang's PhD work in collaboration with Jean Goubault-Larrecq (see above). Hubert Comon-Lundh supervised the following students:

- Véronique Cortier, PhD defended March 2003 [2], CNRS researcher in Nancy since Oct. 1st, 2003;
- Vincent Bernat, second-year PhD student (since Sep. 2001), on reduction of authentication properties to secrecy, and related problems;
- Stéphanie Delaune, DEA "Programmation" student, defended september 2003, supervisors Hubert Comon-Lundh and Florent Jacquemard; 1st year PhD student since october 2003 (advisors Hubert Comon-Lundh and Francis Klay [France Télécom R&D], supported by a CIFRE grant with France Télécom), on the subject of cryptographic protocol verification in the presence of weak passwords, and dictionary attacks.

Florent Jacquemard supervised Stéphanie Delaune's PhD work in collaboration with Hubert Comon-Lundh (see above).

Ralf Treinen and Denis Lugiez (Marseilles, visiting LSV on sabbatical leave) supervised Pascal Lafourcade, a 1st year PhD student since October 01, 2003. The field is verification of cryptographic protocols. The subject of his thesis is the extension of the Dolev-Yao intruder model by algebraic properties of cryptographic primitives. His thesis is funded by a grant from the ACI Rossignol.

9.4. Participation to PhD or habilitation juries

Hubert Comon-Lundh was reviewer (rapporteur) of Mathieu Turuani's PhD thesis in Nancy, France, December 2003. He was examiner of Diane Bahrami's PhD thesis in Évry, France, January 2003, of Véronique Cortier's PhD thesis in Cachan, France, March 2003, of Steve Kremer's PhD thesis at Université Libre de Bruxelles, Belgium, December 2003, and of Sylvain Peyronnet's PhD these at LRI, Université Paris XI Orsay, December 2003.

Jean Goubault-Larrecq was reviewer (rapporteur) of Valérie Viêt-Triêm-Tông's PhD thesis in Rennes, France, December 2003. He was examiner of Alexandre Boisseau's PhD thesis in Cachan, France, September 2003, of Kumar Neeraj Verma's PhD thesis in Cachan, France, September 2003, and of Muriel Roger's PhD thesis in Cachan, France, October 2003.

9.5. Participation to conference program committees or journal editorial boards

David Nowak is member of the program committee of the Third Workshop on Automated Verification of Critical Systems (AVoCS'03), April 2-3 2003, Southampton, UK.

Ralf Treinen is member of the organizing committee of the 18th International Workshop on Unification (UNIF'04), which is planned to be held as a satellite workshop of the International Joint Conference on Automated Reasoning in Cork, Ireland, July 4–8, 2004.

Ralf Treinen in member of the program committee of the Second International Joint Conference on Automated Reasoning (IJCAR 2004), July 4–8, 2004, Cork, Ireland.

Ralf Treinen is member of the program committee of the 15th International Conference on Rewriting Techniques and Applications (RTA 2004) June 3–5, 2004, Aachen, Germany.

Ralf Treinen was elected member of the steering committee of the International Conference on Rewriting Techniques and Application (RTA).

Hubert Comon-Lundh is member of the program committee of the 1st workshop on Security Protocol Verification (SPV; a satellite of CONCUR'2003), September 2003, Marseilles, France.

Jean Goubault-Larrecq is member of the program committee of the 11th International Conference on Automated Theorem Proving with Analytic Tableaux and Related Methods (Tableaux), September 2003, Rome, Italy.

Jean Goubault-Larrecq is member of the programming committee of the 6th International Workshop on Termination (WST'03), June 2003, Valencia, Spain.

Jean Goubault-Larrecq was elected member, then nominated vice-president of the steering committee of the automated theorem proving with tableaux and related methods conference, from September 2003, for three years.

9.6. Participation to symposia, seminars, invitations

Véronique Cortier gave a talk at ESOP'03 in Warsaw, Poland, in April 2003 (accepted paper, [16]) and at RTA'03 in Valencia, Spain, in June 2003 (accepted paper, [15]).

Véronique Cortier has been invited by Martén Abadi, at the University of Santa Cruz of California for 6 weeks (October and November 2003).

Kumar Neeraj Verma gave a talk at RTA'03 (accepted paper, [27])), Valencia, Spain, June 2003, and one at LPAR'03 (accepted paper, [26]), Almaty, Kazakhstan, September 2003.

Kumar Neeraj Verma gave a talk at ACI Crypto VERNAM project meeting, Grenoble, 4 March 2003 on 'Two-Way Equational Tree Automata and Application to Verification of Cryptographic Protocols'.

Hubert Comon-Lundh gave a talk at LICS'03 (accepted paper, [18]), Toronta, Canada, June 2003.

Hubert Comon-Lundh gave a talk at the International Symposium on Verification (Theory & Practice), celebrating Zohar Manna's 1000000_2-th (64th) Birthday in Taormina, Italy, June–July 2003.

Hubert Comon-Lundh gave a talk of synthesis of the Action Spécifique "Sécurité logicielle: modèles et vérification", at the meeting of the RTP (multidisciplinary thematic network) SECC (complex or constrained embedded systems) at the ministry of research, June 2003.

Ralf Treinen gave a talk at the 17th International Workshop on Unification on 'Easy Intruder Deductions' (joint work with Hubert Comon-Lundh, LSV), Valencia, Spain, June 8–9, 2003.

Vincent Bernat gave a talk at the SPV Workshop (satellite of CONCUR 2003) in Marseilles, France, September 2003 (accepted paper, [14]).

Vincent Bernat gave a talk on classical cryptography at the EEA department (electrical engineering), ENS Cachan, 2003. Vincent Bernat gave an invited talk in Bordeaux on reduction of authentication to secrecy in cryptographic protocols.

Stéphanie Delaune gave a talk at the SPV Workshop (satellite of CONCUR 2003) in Marseilles, France, September 2003 (accepted paper, [21]).

Stéphanie Delaune gave a talk at the France Télécom R&D site of Lannion, Brittany, in August 2003, to present her DEA work on intruder deductions in the presence of dictionary attacks. She went to Lannion to work with Francis Klay, France Télécom R&D, on real protocols in November 2003.

Stéphane Demri spent 5 weeks at the Department of Electrical and Electronic Engineering (Melbourne, Australia) to work with Dr. Jennifer Davoren in the framework of the Australian-French cooperation project "Expressive power and complexity of temporal logics for the verification".

Stéphane Demri gave seminar talks at the Department of Philosophy (Melbourne), at the Research School of Information Sciences and Engineering (Canberra), and at the Institute of Computer Science (Namur) on "How to simply translate a bunch of modal logics into a decidable fragment of first-order classical logic", joint work with Hans de Nivelle (MPII, Saarbrücken), March and May 2003; a talk on "LTL with concrete domain", joint work with Deepak D'Souza (Bangalore) during the "Journées Bordeaux-Cachan", Bordeaux, May 2003; an invited talk on "A Parametric Analysis of the State Explosion Problem in Model Checking" joint work with

François Laroussinie and Philippe Schnoebelen during the Dagsthul seminar "Fixed-Parameter Algorithms", July 2003; an invited talk on "(Modal) logics for semistructured data (bis)" during the "3rd workshop on Methods for Modalities", Nancy, September 2003.

Yu Zhang gave a talk at CSL'03 in Vienna, Austria, in August 2003 (accepted paper, together with David Nowak, [28]).

Yu Zhang participated the Advanced School on Mobile Computing, Pisa, Italy, September 2003.

Mathieu Baudet gave an invited talk at the seminar of the LSV, ENS Cachan, November 18, 2003.

Mathieu Baudet gave a talk at the 1st Asian Symposium on Programming Languages (APLAS'2003) in Beijing, China, 27-29th oct 2003 (accepted paper, [23]).

These three events concerned previous results about "controlling and optimizing the usage of one resource" that were obtained during his DEA internship at the Gemplus Research Lab, La Ciotat, 2002.

Jean Goubault-Larrecq gave a talk at the "journées du GDR ALP", CNAM, Paris, January 30, 2003, on 'deux ou trois points de vue sur la vérification de protocoles cryptographiques' (two or three views on cryptographic protocol verification).

Jean Goubault-Larrecq gave a talk at the "journées CATIA", Montpellier, France, February 12–14, on 'Relations logiques pour types monadiques ... et homologie?' (Logical relations for monadic types... and homology?). Jean Goubault-Larrecq gave again the same talk at the "1$^{\text{res}}$ journées Squier and all that", Paris 7, Paris, May 15.

9.7. Miscellaneous

Ralf Treinen maintains, together with Nachum Dershowitz (Tel Aviv University, Israel), the list of open problems of the conference series Rewriting Techniques and Applications (RTA). The list contains currently 100 problems (both open and closed). The list is online at the address http://www.lsv.ens-cachan.fr/rtaloop/.

Ralf Treinen moderates the mailing list Constraints in Computational Logics, which was created in the Esprit working group of the same name, and which continues to operate after the end of the working group. The mailing list currently has 126 subscribers in the field of computational logics and mainly carries announcements of interest to the community. Further information about the mailing list, including an archive of past messages, is available at http://www.lsv.ens-cachan.fr/ccl/.

Ralf Treinen maintains the home page of the International Workshop on Unification (UNIF), which provides detailed information about the past events in UNIF's 17-years history. The UNIF home page is available at http://www.lsv.ens-cachan.fr/unif/.

9.8. Prizes

Jean Goubault-Larrecq was awarded the best referee award of the journal Theoretical Computer Science.

Véronique Cortier obtained the SPÉCIF award for best PhD thesis in 2003 for her work on automated verification of cryptographic protocols [2].

10. Bibliography

Doctoral dissertations and "Habilitation" theses

[1] A. BOISSEAU. *Abstractions pour la vérification de propriétés de sécurité de protocoles cryptographiques.* Ph. D. Thesis, ENS de Cachan, September, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Boisseau-these.ps.

[2] V. CORTIER. *Vérification automatique des protocoles cryptographiques.* Ph. D. Thesis, ENS de Cachan, March, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Cortier-these.ps, Prix SPÉCIF 2003.

[3] M. ROGER. *Raffinements de la résolution et vérification de protocoles cryptographiques.* Ph. D. Thesis, ENS de Cachan, October, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Roger-these.ps.

[4] K. N. VERMA. *Automates d'arbres bidirectionnels modulo théories équationnelles.* Ph. D. Thesis, ENS de Cachan, September, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Verma-these.ps.

Articles in referred journals and book chapters

[5] N. ALECHINA, S. DEMRI, M. DE RIJKE. *A Modal Perspective on Path Constraints.* in « Journal of Logic and Computation », 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/final-jlc-adr.ps, To appear.

[6] H. COMON, V. CORTIER. *Tree Automata with One Memory, Set Constraints and Cryptographic Protocols.* in « Theoretical Computer Science », 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/ComonCortierTCS1.ps, To appear.

[7] H. COMON, F. JACQUEMARD. *Ground Reducibility is EXPTIME-Complete.* in « Information and Computation », number 1, volume 187, 2003, pages 123–153.

[8] H. COMON, Y. JURSKI. *Counter Automata, Fixed Points and Additive Theories.* in « Theoretical Computer Science », 2003, To appear.

[9] H. COMON, P. NARENDRAN, R. NIEUWENHUIS, M. RUSINOWITCH. *Deciding the Confluence of Ordered Term Rewrite Systems.* in « ACM Trans. Computational Logic », number 1, volume 4, 2003, pages 33–55.

[10] S. DEMRI. *A Polynomial Space Construction of Tree-Like Models for Logics with Local Chains of Modal Connectives.* in « Theoretical Computer Science », number 1–3, volume 300, 2003, pages 235–258, http://www.lsv.ens-cachan.fr/Publis/PAPERS/demri-tcs02.ps.

[11] J. GOUBAULT-LARRECQ. *Extensions of Valuations.* in « Math. Struct. in Comp. Science », 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2002-17.rr.ps, To appear.

[12] J. GOUBAULT-LARRECQ, É. GOUBAULT. *On the Geometry of Intuitionistic S4 Proofs.* in « Homology, Homotopy and Applications », number 2, volume 5, 2003, pages 137–209, http://www.lsv.ens-cachan.fr/Publis/PAPERS/S4G.ps.

[13] F. JACQUEMARD. *Reachability and Confluence are Undecidable for Flat Term Rewriting Systems.* in « Information Processing Letters », number 5, volume 87, 2003, pages 265–270, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-6.rr.ps.

Publications in Conferences and Workshops

[14] V. BERNAT. *Towards a Logic for Verification of Security Protocols.* in « Proc. Workshop on Security Protocols Verification (SPV'2003) », Marseille, France, September, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Bernat-spv2003.ps.

[15] H. COMON-LUNDH, V. CORTIER. *New Decidability Results for Fragments of First-Order Logic and Application to Cryptographic Protocols.* in « Proc. 14th Int. Conf. Rewriting Techniques and Applications

(RTA'2003), Valencia, Spain, June 2003 », series Lecture Notes in Computer Science, volume 2706, Springer, pages 148–164, 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-2.rr.ps.

[16] H. COMON-LUNDH, V. CORTIER. *Security Properties: Two Agents Are Sufficient.* in « Proc. 12th European Symposium on Programming (ESOP'2003), Warsaw, Poland, Apr. 2003 », series Lecture Notes in Computer Science, volume 2618, Springer, pages 99–113, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/ComonCortierESOP03.ps.

[17] H. COMON-LUNDH, V. CORTIER. *Security Properties: Two Agents Are Sufficient.* in « Special Issue of the Journal of Science of Computer Programming (SCP) », 2003, To appear.

[18] H. COMON-LUNDH, V. SHMATIKOV. *Intruder Deductions, Constraint Solving and Insecurity Decision in Presence of Exclusive Or.* in « Proc. 18th IEEE Symp. Logic in Computer Science (LICS'2003), Ottawa, Canada, June 2003 », IEEE Comp. Soc. Press, pages 271–280, June, 2003.

[19] H. COMON-LUNDH, R. TREINEN. *Easy Intruder Deductions.* in « Proc. Int. Symp. Verification (Theory & Practice). Celebrating Zohar Manna's 1000000_2-th Birthday. Taormina, Italy, June–July 2003 », series Lecture Notes in Computer Science, volume 2772, Springer, 2003, To appear.

[20] V. CORTIER. *Résultats de décidabilité et d'inécidabilité pour les protocoles crypto graphiques.*. in « Technique et Science Informatiques (TSI) », 2003, To appear.

[21] S. DELAUNE. *Intruder Deduction Problem in Presence of Guessing Attacks.* in « Proc. Workshop on Security Protocols Verification (SPV'2003) », pages 26–30, Marseille, France, September, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Del-spv2003.ps.

[22] S. DEMRI, H. DE NIVELLE. *Relational Translations into GF2.* in « Proc. Third Workshop on Methods for Modalities », pages 93–108, Nancy, France, September, 2003, (Informal proceedings).

[23] A. GALLAND, M. BAUDET. *Controlling and Optimizing the Usage of One Resource.* in « Proc. 1st Asian Symp. on Programming Languages and Systems (APLAS'03), Beijing, China, Nov. 2003 », series Lecture Notes in Computer Science, volume 2895, Springer, pages 195–211, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/GB03aplas.ps.

[24] A. GALLAND, M. BAUDET. *Économiser l'or du banquier.* in « Actes 3ème Conférence Française sur les Systèmes d'Exploitation (CFSE'3) », INRIA, M. AUGUIN, F. BAUDE, D. LAVENIER, M. RIVEILL, editors, pages 638–649, La Colle sur Loup, France, October, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/GB03cfse.ps.

[25] K. N. VERMA, J. GOUBAULT-LARRECQ. *Karp-Miller Trees for an Extension of VASS.* in « AMAST'2004 », 2003.

[26] K. N. VERMA. *On Closure under Complementation of Equational Tree Automata for Theories Extending AC.* in « Proc. 10th Int. Conf. Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'2003), Almaty, Kazakhstan, Sep. 2003 », series Lecture Notes in Artificial Intelligence, volume 2850, Springer, pages 183–195, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Verma-lpar03.ps.

[27] K. N. VERMA. *Two-Way Equational Tree Automata for AC-like Theories: Decidability and Closure Properties.* in « Proc. 14th Int. Conf. Rewriting Techniques and Applications (RTA'2003), Valencia, Spain, June 2003 », series Lecture Notes in Computer Science, volume 2706, Springer, pages 180–196, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Ver-rta03.ps.

[28] Y. ZHANG, D. NOWAK. *Logical Relations for Dynamic Name Creation.* in « Proc. 17th Int. Workshop Computer Science Logic (CSL'2003) and 8th Kurt Gödel Coll. (KGL'2003), Vienna, Austria, Aug. 2003 », series Lecture Notes in Computer Science, volume 2803, Springer, pages 575–588, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/ZN-csl2003.ps.

Internal Reports

[29] M. BAUDET. *Random Polynomial-Time Attacks and Dolev-Yao Models.* Research Report, number LSV-03-16, Lab. Specification and Verification, ENS de Cachan, Cachan, France, December, 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-16.rr.ps, 15 pages.

[30] H. COMON-LUNDH, V. SHMATIKOV. *Constraint Solving, Exclusive Or and the Decision of Confidentiality for Security Protocols Assuming a Bounded Number of Sessions.* Research Report, number LSV-03-1, Lab. Specification and Verification, ENS de Cachan, Cachan, France, January, 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-1.rr.ps, 17 pages.

[31] H. COMON-LUNDH, R. TREINEN. *Easy Intruder Deductions.* Research Report, number LSV-03-8, Lab. Specification and Verification, ENS de Cachan, Cachan, France, April, 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-8.rr.ps, 17 pages.

[32] S. DEMRI, D. D'SOUZA. *An Automata-Theoretic Approach to Constraint LTL.* Research Report, number LSV-03-11, Lab. Specification and Verification, ENS de Cachan, Cachan, France, August, 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-11.rr.ps, 40 pages.

[33] S. DEMRI, H. DE NIVELLE. *Deciding Regular Grammar Logics with Converse through First-Order Logic.* Research Report, number LSV-03-4, Lab. Specification and Verification, ENS de Cachan, Cachan, France, February, 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-4.rr.ps, 29 pages.

[34] S. DEMRI. *LTL over Integer Periodicity Constraints.* Research Report, number LSV-03-13, Lab. Specification and Verification, ENS de Cachan, Cachan, France, October, 2003, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2003-13.rr.ps, 34 pages.

[35] J. GOUBAULT-LARRECQ. *The ħ1 Tool Suite.* LSV, CNRS UMR 8643 & INRIA projet SECSI & ENS Cachan, 2003, Software, version 1.1.

Miscellaneous

[36] H. COMON-LUNDH, R. TREINEN. *Preliminary lecture notes, logic course, first term, first year, magistère STIC.* Lecture notes, 2003.

[37] S. DELAUNE. *Vérification de protocoles de sécurité dans un modèle de l'intrus étendu.* Mémoire de DEA, DEA Programmation, Paris, September, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Delaune-

dea2003.ps, 62 pages.

[38] S. DEMRI. *(Modal) Logics for Semistructured Data (Bis).* Invited Talk. Third Workshop on Methods for Modalities, Nancy, France, September, 2003.

[39] S. DEMRI, M. DUCASSÉ, J. GOUBAULT-LARRECQ, L. MÉ, J. OLIVAIN, C. PICARONNY, J.-P. POUZOL, É. TOTEL, B. VIVINIS. *Algorithmes de détection et langages de signatures.* Sous-projet 3, livrable 3 du projet RNTL DICO. Version 1, October, 2003, 72 pages.

[40] J. GOUBAULT-LARRECQ. *Programmation.* Lecture notes of the course "programmation", first term, first year, magistère STIC, 2003, http://www.lsv.ens-cachan.fr/~goubault/cours.html#programmation, 78 pages.

[41] J. GOUBAULT-LARRECQ. *Résolution ordonnée avec sélection et classes décidables de la logique du premier ordre.* Lecture notes for the course "démonstration automatique et vérification de protocoles cryptographiques" (with Hubert Comon-Lundh), DEA "programmation", 2003, http://www.lsv.ens-cachan.fr/~goubault/SOresol.ps, 70 pages.

[42] F. JACQUEMARD. *The EVA translator, version 6.* Rapport numéro 9 du projet RNTL EVA, July, 2003, 38 pages.

[43] R. TREINEN. *Constraint Solving and Decision Problems of First-Order Theories of Concrete Domains.* Lecture notes, 2003, http://www.lsv.ens-cachan.fr/~treinen/publi/concrete.ps.gz.

Bibliography in notes

[44] M. ABADI, A. D. GORDON. *A Calculus for Cryptographic Protocols: The Spi Calculus.* in « Proc. 4th ACM Conference on Computer and Communications Security (CCS) », pages 36–47, 1997.

[45] J. F. ALLEN. *Time and Time Again: The Many Ways to Represent Time.* in « International Journal of Intelligent Systems », volume 6, 1991, pages 341–355, http://citeseer.nj.nec.com/allen91time.html.

[46] R. ALUR, T. A. HENZINGER, O. KUPFERMAN. *Alternating-Time Temporal Logic.* in « Compositionality: The Significant Difference—Revised Lectures of the International Symposium on Compositionality (COMPOS'97), September 1997 », Springer Verlag LNCS 1536, pages 23–60, Bad Malente, Germany, 1998.

[47] R. AMADIO, W. CHARATONIK. *On Name Generation and Set-Based Analysis in the Dolev-Yao Model.* in « CONCUR'02 », Springer-Verlag LNCS 2421, 2002, pages 499-514.

[48] L. O. ANDERSEN. *Program Analysis and Specialization for the C Programming Language.* Ph. D. Thesis, DIKU, University of Copenhagen, 1994, ftp://ftp.diku.dk/pub/diku/semantics/papers/D-203.dvi.Z.

[49] L. BACHMAIR, H. GANZINGER. *Resolution Theorem Proving.* J. A. ROBINSON, A. VORONKOV, editors, in « Handbook of Automated Reasoning », volume I, North-Holland, 2001, chapter 2, pages 19–99.

[50] M. BELLARE. *Practice-Oriented Provable Security.* in « Proc. 1st Information Security Workshop (ISW) », Springer Verlag LNCS 1561, pages 1–15, 1997.

[51] M. BELLARE, A. DESAI, D. POINTCHEVAL, P. ROGAWAY. *Relations Among Notions of Security for Public-Key Encryption Schemes.* in « Advances in Cryptology – Proc. CRYPTO '98 », Springer Verlag LNCS 1462, pages 26–45, 1998.

[52] B. BLANCHET. *An Efficient Cryptographic Protocol Verifier Based on Prolog Rules.* in « 14th IEEE Computer Security Foundations Workshop (CSFW-14) », IEEE Computer Society Press, 2001, pages 82–96.

[53] A. BOUDET, H. COMON. *Diophantine Equations, Presburger Arithmetic and Finite Automata.* in « Colloquium on Trees in Algebra and Programming (CAAP'96) », Springer Verlag LNCS 1059, H. KIRCHNER, editor, pages 30–43, 1996.

[54] M. BURROWS, M. ABADI, R. NEEDHAM. *A Logic of Authentication.* in « Proceedings of the Royal Society », number 1871, volume 426, December, 1989, pages 233–271.

[55] D. L. CHAUM. *The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability.* in « Journal of Cryptology », number 1, volume 1, 1988, pages 65–75.

[56] J. CLARK, J. JACOB. *A Survey of Authentication Protocol Literature: Version 1.0..* 1997, http://www.cs.york.ac.uk/~jac/papers/drareview.ps.gz.

[57] H. COMON, V. CORTIER, J. MITCHELL. *Tree Automata with One Memory, Set Constraints and Ping-Pong Protocols.* in « Proc. 28th International Conference on Automata, Languages and Programming (ICALP) », Springer-Verlag LNCS 2076, 2001, pages 682–693.

[58] H. COMON, M. DAUCHET, R. GILLERON, F. JACQUEMARD, D. LUGIEZ, S. TISON, M. TOMMASI. *Tree Automata Techniques and Applications.* http://www.grappa.univ-lille3.fr/tata, 1997.

[59] H. COMON, V. SHMATIKOV. *Is it possible to decide whether a cryptographic protocol is secure or not ?.* J. GOUBAULT-LARRECQ, editor, in « Journal of Telecommunications and Information Technology, Special Issue on Models and Methods for Cryptographic Protocol Verification », volume 4, Instytut Łącsności (Institute of Telecommunications), Warsaw, Poland, December, 2002, pages 3–13.

[60] H. COMON-LUNDH, R. TREINEN. *Easy Intruder Deductions.* in « Proc. Int. Symp. Verification (Theory & Practice). Celebrating Zohar Manna's 1000000_2-th Birthday », Springer Verlag LNCS 2772, Taormina, Italy, June–July, 2003.

[61] V. CORTIER. *Outil de vérification SECURIFY.* Report number 7 of the RNTL project EVA, May, 2002, http://www.lsv.ens-cachan.fr/~cortier/EVA-TR7.pdf, 6 pages.

[62] V. CORTIER, J. MILLEN, H. RUESS. *Proving Secrecy is Easy Enough.* in « Proc. 14th IEEE Computer Security Foundations Workshop », 2001.

[63] W. DIFFIE, M. HELLMAN. *New Directions in Cryptography.* in « IEEE Transactions on Information Theory », number 6, volume IT-22, 1976, pages 644–654.

[64] D. DOLEV, A. C. YAO. *On the Security of Pubic Key Protocols.* in « IEEE Transactions on Information Theory », number 2, volume IT-29, March, 1983, pages 198–208.

[65] J. A. GARAY, M. JAKOBSSON, P. MACKENZIE. *Abuse-Free Optimistic Contract Signing.* in « Advances in Cryptology—Proc. 19th Annual Int. Cryptology Conference (CRYPTO'99) », Springer Verlag LNCS 1666, M. J. WIENER, editor, pages 449–466, Santa Barbara, CA, August, 1999.

[66] O. GAY. *Exploitation avancée de buffer overflows.* Technical report, Security and Cryptography Laboratory (LASEC), École Polytechnique Fédérale de Lausanne, June, 2002, http://www.k-otik.com/papers/Exploitation_Avancee_BOF.pdf.

[67] S. GOLDWASSER, M. BELLARE. *Lecture Notes on Cryptography.* Available at http://www-cse.ucsd.edu/~mihir/papers/gb.ps.gz, August, 1999.

[68] S. GOLDWASSER, S. MICALI. *Probabilistic Encryption.* in « Journal of Computer and System Sciences », volume 28, 1984, pages 270–299.

[69] S. GOLDWASSER, S. MICALI, R. L. RIVEST. *A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks.* in « SIAM Journal on Computing », number 2, volume 17, 1988, pages 281–308.

[70] J. GOUBAULT-LARRECQ. *A Method for Automatic Cryptographic Protocol Verification (Extended Abstract).* in « Proceedings of the Workshop on Formal Methods in Parallel Programming, Theory and Applications (FMPPTA'2000) », series Lecture Notes in Computer Science LNCS 1800, Springer Verlag, 2000, pages 977–984.

[71] J. GOUBAULT-LARRECQ. *Un Algorithme pour l'Analyse de Logs.* Research Report, number LSV-02-18, Lab. Specification and Verification, ENS de Cachan, Cachan, France, November, 2002, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2002-18.rr.ps, 33 pages.

[72] J. GOUBAULT-LARRECQ. *Vérification de protocoles cryptographiques : la logique à la rescousse !.* in « Actes du 1er workshop international sur la sécurité des communications sur Internet (SECI'02) », INRIA, collection didactique, J. GOUBAULT-LARRECQ, editor, pages 119–152, 2002, http://www.lsv.ens-cachan.fr/~goubault/SECI-02/Final/JGL/jgl.ps.

[73] J. GOUBAULT-LARRECQ. *Une fois qu'on n'a pas trouvé de preuve, comment le faire comprendre à un assistant de preuve?.* V. MÉNISSIER-MORAIN, editor, in « Actes des 12èmes Journées Francophones des Langages Applicatifs (JFLA'04) », INRIA, collection didactique, 2004.

[74] J. GOUBAULT-LARRECQ, S. LASOTA, D. NOWAK. *Logical Relations for Monadic Types.* in « Proc. 16th Int. Workshop Computer Science Logic (CSL'2002) », Springer-Verlag LNCS 2471, pages 553–568, Edinburgh, Scotland, September, 2002.

[75] J. GOUBAULT-LARRECQ, J.-P. POUZOL, S. DEMRI, L. MÉ, P. CARLE. *Langages de Détection d'Attaques par Signatures.* Sous-projet 3, livrable 1 du projet RNTL DICO. Version 1, June, 2002, 30 pages.

[76] J. GOUBAULT-LARRECQ, K. N. VERMA. *Alternating two-way AC-tree automata.* Research Report, number LSV-02-11, LSV, September, 2002, http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2002-

11.rr.ps.

[77] T. HILLENBRAND, A. BUCH, R. VOGT, B. LOCHNER. *WALDMEISTER — High-Performance Equational Deduction.* in « Journal of Automated Reasoning », number 2, volume 18, 1997, pages 265-270, http://citeseer.nj.nec.com/hillenbrand97waldmeister.html.

[78] D. HUGHES, V. SHMATIKOV. *Information Hiding, Anonymity and Privacy: A Modular Approach.* in « Journal of Computer Security », 2003, To appear.

[79] H. HÜTTEL. *Deciding Framed Bisimilarity.* in « Proc. 4th Int. Workshop on Verification of Infinite-State Systems (INFINITY'02) », Electronic Notes in Theoretical Computer Science 68, A. KUCERA, R. MAYR, editors, Brno, Czech Republic, August, 2002.

[80] A. JOUX. *Qu'est-ce que la sécurité d'un algorithme de chiffrement?.* 18 September, 2002, Talk at the DCSSI-LogiCal-SECSI meeting, Rocquencourt.

[81] G. LOWE. *An Attack on the Needham-Schroeder Public-Key Authentication Protocol.* in « Information Processing Letters », number 3, volume 56, 1996, pages 131–133.

[82] J. MILLEN, G. DENKER. *CAPSL and MuCAPSL.* J. GOUBAULT-LARRECQ, editor, in « Journal of Telecommunications and Information Technology, Special Issue on Models and Methods for Cryptographic Protocol Verification », volume 4, Instytut Łącsności (Institute of Telecommunications), Warsaw, Poland, December, 2002, pages 15–25.

[83] K. D. MITNICK, W. L. SIMON. *The Art of Deception: Controlling the Human Element of Security.* Wiley Publishing Company, October, 2002, ISBN 0471237124.

[84] D. MONNIAUX. *Abstracting Cryptographic Protocols with Tree Automata.* in « 6th International Static Analysis Symposium (SAS'99) », Springer-Verlag LNCS 1694, 1999, pages 149–163.

[85] R. M. NEEDHAM, M. D. SCHROEDER. *Using Encryption for Authentication in Large Networks of Computers.* in « Communications of the ACM », number 12, volume 21, 1978, pages 993–999.

[86] F. NIELSON, H. R. NIELSON, H. SEIDL. *Normalizable Horn Clauses, Strongly Recognizable Relations and Spi.* in « 9th Static Analysis Symposium (SAS) », Springer Verlag LNCS 2477, 2002.

[87] D. OTWAY, O. REES. *Efficient and Timely Mutual Authentication.* in « Operating Systems Review », number 1, volume 21, 1987, pages 8–10.

[88] A. PITTS, I. STARK. *Observable Properties of Higher Order Functions that Dynamically Create Local Names, or: What's new?.* in « Mathematic Foundations of Computer Science (MFCS'93) », series LNCS, number 711, Springer-Verlag LNCS 711, pages 122–141, 1993, http://www.dcs.ed.ac.uk/~stark/publications/obsphown.html.

[89] G. PLOTKIN. *Building in Equational Theories.* in « Machine Intelligence », volume 7, 1972, pages 73–90.

[90] D. POINTCHEVAL. *Asymmetric cryptography and practical security.* in « Journal of Telecommunications and Information Security », volume 4, 2002, pages 41–56.

[91] W. PURCZYŃSKI. *Linux Kernel Privileged Process Hijacking Vulnerability.* http://www.securityfocus.com/bid/7112, March, 2003, BugTraq Id 7112.

[92] A. RIAZANOV, A. VORONKOV. *Vampire 1.1 (System Description).* in « Proc. 1st Intl. Joint Conf. Automated Reasoning », Springer Verlag LNCS 2083, R. GORÉ, A. LEITSCH, T. NIPKOW, editors, pages 376–380, Siena, Italy, June, 2001.

[93] M. ROGER, J. GOUBAULT-LARRECQ. *Log Auditing through Model Checking.* in « Proc. 14th IEEE Computer Security Foundations Workshop (CSFW'01), Cape Breton, Nova Scotia, Canada, June 2001 », IEEE Comp. Soc. Press, pages 220–236, 2001, http://www.lsv.ens-cachan.fr/Publis/PAPERS/RogGou-csfw01.ps.

[94] M. ROGER. *Raffinements de la résolution et vérification de protocoles cryptographiques.* Ph. D. Thesis, ENS de Cachan, October, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/Roger-these.ps.

[95] A. SIMON, A. KING. *Analyzing String Buffers in C.* in « Intl. Conf. on Algebraic Methods and Software Methodology (AMAST'2002) », pages 365–379, 2002.

[96] B. STEENSGARD. *Points-to Analysis in Almost Linear Time.* in « 24th ACM SIGPLAN-SIGACT Symp. Principles of Programming Languages », pages 32–41, January, 1997.

[97] M. STEINER, G. TSUDIK, M. WAIDNER. *Key Agreement in Dynamic Peer Groups.* in « IEEE Transactions on Parallel and Distributed Systems », number 8, volume 11, 2000, pages 769–780.

[98] E. SUMII, B. C. PIERCE. *Logical Relations for Encryption.* in « Proc. 14th Computer Security Foundations Workshop », pages 256–272, 2001.

[99] G. SUTCLIFFE, C. SUTTNER. *The TPTP Problem Library for Automated Theorem Proving.* http://www.cs.miami.edu/~tptp/, 2001.

[100] S. A. THOMAS. *SSL & TLS Essentials: Securing the Web.* Wiley, 2000, ISBN 0471383546.

[101] C. WEIDENBACH, U. BRAHM, T. HILLENBRAND, E. KEEN, C. THEOBALD, D. TOPIĆ. *SPASS Version 2.0.* in « Proceedings of the 18th International Conference on Automated Deduction », Springer-Verlag LNAI 2392, A. VORONKOV, editor, 2002.

[102] C. WEIDENBACH. *Towards an Automatic Analysis of Security Protocols.* in « Proceedings of the 16th International Conference on Automated Deduction (CADE-16) », Springer-Verlag LNAI 1632, H. GANZINGER, editor, pages 378–382, 1999.

[103] Y. ZHANG, D. NOWAK. *Logical Relations for Dynamic Name Creation.* in « Proc. 17th Intl. Workshop Computer Science Logic (CSL'2003) », Springer Verlag LNCS 2803, pages 575–588, 2003, http://www.lsv.ens-cachan.fr/Publis/PAPERS/ZN-csl2003.ps.

[104] BY <JBS>. *La carte à puce nouvelle génération T2G est hackable.* in « The Hackademy Journal », volume 9, June, 2003, pages 3–6.