Security through Analysis and Verification

Organised by
Pierpaolo Degano (Pisa), Roberto Gorrieri (Bologna),
Chris Hankin (London), Flemming Nielson (Aarhus),
Hanne Riis Nielson (Aarhus).

1 Introduction

Security has increasing relevance and importance for real life applications such as Internet transactions, electronic commerce, electronic voting and smart cards thereby stressing the need for appropriate checking measures. Techniques from program verification and program logics have already proven their worth for the machine-assisted validation of secure communication protocols. This seminar emanates from recent applications of program analysis which allow fully automatic validation of systems against certain types of attack. It took place from 10. December 2000 to 15. December 2000 and comprised a number of talks, as indicated by the following abstracts, informal discussions as well as a discussion in plenum forming the basis for this report.

The nature of security. Security has many facets: confidentiality, integrity, authentication, watermarking, prevention of denial of service, auditing etc. It emerged quite clearly during the seminar that there is no clear consensus about the precise definition of many of these concepts. There was agreement that to the extent possible one should try to reconcile and clarify the differences, possibly in the form of defining a formal language or logic for security requirements. During the discussions it became clear that confidentiality, as in certified mail, was a hard concept to capture precisely, despite the fact that consideration of information flow (both forward and backward) account for many ingredients of integrity and secrecy.

Formal approaches to security. The main techniques represented at the seminar involve flow-based program analysis, model checking, type systems, simulation techniques, process algebras, cryptographic proofs and dynamic enforcement of security policies. Several presentations aimed at spanning a range of these and a particularly promising direction was the integration of cryptographic reasoning into the more classical approaches to formal methods.

One approach to the security of a system centered around the idea that the harder to make an intrusion the more secure the system. Several presentations expanded on the ability to intercept, manipulate and forge messages, including polynomial-time bounded attackers, Dolev-Yao saboteurs and other forms of characteristic intruders.

As pointed out by the security community there is a need to address a variety of more realistic applications and protocols. Indeed a number of techniques have only been applied to the simplest protocols. By contrast, applications towards multicast protocols and group-key management seem to challenge state-of-the-art in the most advanced approaches.

There also is a need to identify the most appropriate specification formalisms and programming calculi for capturing both qualitative and quantitative, possibilistic and probabilistic aspects of secure systems and protocols.

There was no agreement about whether the approach taken should aim at studying protocols and procedures at the abstract level or should eventually aim at studying the fielded software system. However, there was consensus that special care is required when applying well-defined mathematical theories and formal validations to actual systems.

Conclusion. Thanks to the pleasant surroundings the different communities taking part in the meeting were very successful in establishing an amiable social atmosphere thereby facilitating the necessary cross-fertilization for the field to progress. Given the predictable growth in electronic commerce and mobile computing the problems discussed are increasingly important and can only benefit from the interaction of a variety of approaches. It is our prediction that the field will only be slightly perturbed by future developments in quantum cryptography and the emerging computational models.

2 Abstracts

Secrecy Types for Asymmetric Communication

Bruno Blanchet, INRIA Rocquencourt

Joint work with Martín Abadi.
We develop a typed process calculus for security protocols in which types convey secrecy properties. We prove a secrecy theorem that shows that well-typed processes do not reveal their secrets. We focus on asymmetric communication primitives, especially on public-key encryption. These present special difficulties, partly because they rely on related capabilities (e.g., "public" and "private" keys) with different levels of secrecy and scopes. Their treatment constitutes the main novelty of this work.

Our type system can be applied to some small but subtle security protocols. For example, in the Needham-Schroeder public-key protocol (a standard test case), one might expect a certain nonce to be secret; however, the protocol fails to typecheck under the assumption that this nonce is secret. This failure is not a shortcoming of the type system: it manifests Lowe's attack on the protocol. On the other hand, a corrected version of the protocol does typecheck under the assumption. Our secrecy theorem yields the expected secrecy property in this case.

Randomness Recycling in Constant-Round Private Computation

Carlo Blundo, Università di Salerno

Joint work with Clemente Galdi and Pino Persiano.
Consider a set of n players $\mathcal{P} = \{P_1, \dots, P_n\}$, each possessing a private input x_i, who wish to compute a certain function f of these variables. A *private* protocol allows the players to distributively compute the value of the function $f(x_1, \dots, x_n)$, in such a way that at the end of the protocol each player knows the value of the function but no player has information about the other players' input more than what can be inferred from its own input and from the value of the function. Obviously some players can collude together in order to infer information about other players' inputs. A protocol is said to be t-private if any coalition consisting of at most t-players cannot learn additional information from the protocol execution.

In this work we study the randomness complexity needed to distributively perform k XOR computation in a t-private way using constant-round protocols.

We show that cover-free families allow the recycling of random bits for constant-round private protocols. More precisely, we show that after an 1-round initialization phase during which random bits are distributed among the players, it is possible to perform each of k XOR computations using 2-rounds of communication. In each phase the random bits are used according to a cover-free family and this allows to use each random bit for more than one computation.

For $t = 2$, we design a protocol that uses $O(n \log k)$ random bits instead of $O(nk)$ bits if no recycling is performed. More generally, if $t > 1$ then $O(kt^2 \log n)$ random bits are sufficient to accomplish this task, for $t = O(n^{1/2-\epsilon})$ for constant $\epsilon > 0$.

Secrecy and Non-Interference for History Dependent Cryptography

Chiara Bodei, Università di Pisa

Joint work with Pierpaolo Degano, Flemming Nielson and Hanne Riis Nielson.
We introduce the ν SPI-calculus that strengthens the notion of "perfect symmetric cryptography" of the spi-calculus by taking time into account. This involves defining an operational semantics, defining a control flow analysis (CFA) in the form of a flow logic, and proving semantic correctness. Our first result is that secrecy in the sense of Dolev-Yao can be expressed in terms of the CFA. Our second result is that also non-interference in the sense of Abadi can be expressed in terms of the CFA; unlike Abadi we find the non-interference property to be an extension of the Dolev-Yao property.

A Flexible Framework for Formalizing Security Protocols (kicking the devil out of the details)

Iliano Cervesato, ITT Industries Inc.

When it comes to security protocol analysis, the devil lies in the detail... or more precisely in how they are expressed. We present Typed MSR, an unambiguous, flexible, powerful and relatively simple specification framework for crypto-protocols. Typed MSR is a strongly typed multiset rewriting language over first order atomic formulas. It uses existential quantifiers to model the generation of fresh data, memory predicates to encode systems consisting of a collection of coordinated subprotocols, and guards to handle objects belonging to complex interpretation domains in an abstract and modular way. Its typing infrastructure, based on the theory of dependent types with subsorting, supports type-checking and access control verification. Access control is shown to be intimately connected to the Dolev-Yao intruder. We also discuss the execution model of Typed MSR and present an example.

Process Algebraic Analysis of Cryptographic Protocols

Rocco De Nicola, Università di Firenze

Joint work with Michele Boreale and Rosario Pugliese.

Recent approaches to the analysis of crypto-protocols build on concepts which are well-established in the field of process algebras, such as labelled transition systems and observational semantics. A protocol is modelled as a concurrent system, described in some process calculus, and natural security concepts such as secrecy and authenticity are expressed as “behavioural equivalence”. Moreover, compositional reasoning becomes possible. In the talk we outlined recent work in this direction that stems from the use of cryptographic versions of the pi-calculus (Abadi and Gordon’s spi-calculus) as protocol description languages. A major line of research is centered around the notions of observational equivalence, which permit making such notions as “attacker” and “secrecy” rigorous. The definitions of these equivalences, while rigorous and intuitive, suffer from universal quantification over contexts (attackers), that makes equivalence checking very hard. We showed how to avoid such quantification and obtain more tractable characterizations. The latters are based on an “environment-sensitive” lts (as opposed to the “standard” *lts*, which only explains process intentions). We discussed the impact of these approaches on a specific example, a simplified version of the Kerberos protocol.

Flow Logic for Security

Pierpaolo Degano, Università di Pisa

Joint work with Chiara Bodei, Flemming Nielson and Hanne Riis Nielson.

Flow Logic predicts at static time safe and computable approximates, or estimates, of the set of values that the objects of a program may assume at run time. We advocate here a specific static analysis of this kind, namely Control Flow Analysis (CFA), for checking security properties. In this talk we shall discuss the following typical CFA paradigm:

1. choose the values of interest for the required application, so defining the shape of the estimates;

2. state, through some logical clauses, when estimates are valid, i.e. when they are solutions;

3. prove that all solutions are semantically correct;

4. show that there always exists a least solution;
5. construct, often in polynomial time, the least solution.

The above will be exemplified on the pi- and the spi-calculi. The applications to security will then require to

6. select a dynamic property;
7. define a static check on a solution of a system (computed once and for all!) that, if passed, implies the selected dynamic property;
8. show that, even if the property is checked for a system in isolation, it will also hold in the presence of an unknown attacker (often expressed as a ”most powewrful” attacker).

We shall briefly show the kind of checks needed to ensure some security properties, among which secrecy (á la Dolev-Yao) and non-interference (á la Abadi). Finally, we shall mention other properties and other calculi for whicc a CFA has been defined, and the existence of “efficient” tools for computing solutions and checking security properties.

Unicast and Multicast Protocol Analysis in CAPSL

Grit Denker, SRI International

Joint work with Jon Millen.
CAPSL, a Common Authentication Protocol Specification Language, is a high-level language to support security analysis of cryptographic authentication and key distribution protocols. The CAPSL Integrated Protocol Environment provides parser, type checker and connectors to analysis tools like the PVS theorem prover, and the model checkers Athena and Maude. CAPSL employs an intermediate language, CIL, that expresses state transitions with term-rewriting rules. Our experience has shown that this is an appropriate representation easily adaptable for many analysis tools. I will report on the current status of the protocol environment.

The current version of CAPSL is restricted to unicast protocols where each message has a single addressee. We are currently investigating extensions of CAPSL in order to handle group management protocols. These protocols impose new challenges for modelling and analysis techniques. Group management includes such activities as enrolling and disenrolling group members, designating a leader and changing that designation, distributing a common encryption key, and achieving full or majority consensus. I will report on our first results to extend CAPSL with powerful and elegant constructs for expressing secure group multicast protocols.

KLAIM: Design and Implementation

GianLuigi Ferrari, Università di Pisa

Joint work with Lorenzo Bettini, Rocco De Nicola, Rosario Pugliese and Betti Venneri.

In the design of programming languages for highly distributed systems where processes can migrate and execute on new hosts, the integration of security mechanisms is a major challenge. We report our experience in the design of an experimental programming language, called Klaim (Kernel Language for Agent Interaction and Mobility), which provides mechanisms to customize access control policies. Klaim security architecture exploits a capability-based type system to provide mechanisms for specifying and enforcing policies that control uses of resources and authorize migration and execution of processes.

We illustrate the design of Klaim access control model and show how a capability-based type system is a useful and effective tool to write secure network applications. The main features of the approach are summarized below.

- Access rights are explicitly recorded in type specification, thereby providing declarative specifications of access control policies.
- Subtyping of access rights: alternative access policies can be defined by replacing the default policy with a new, more restrictive policy that is a subtype of the default policy.
- Mobile processes are typed by their access control requirements; these are automatically generated by the type inference procedure.
- Process access control requirements are clearly separated from node access control policies: the node access control policy is formulated by the authority (the owner) of the node and is dynamically enforceable at runtime.

  Klaim type system is sufficiently powerful to express access patterns of policies for customizing and confining the route of process migration.

The language and the design philosophy underlying Klaim are presented in [3]. The mathematical foundations (decidability and soundness) of the kernel of Klaim type system can be found in [4] (a preliminary presentation appeared in [2]). The prototype implementation of the language is described in [1].

[1] L. Bettini, R. De Nicola, G. Ferrari, R. Pugliese. Interactive Mobile Agents in XKlaim. *IEEE Seventh International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, *Proceedings* (P. Ciancarini, R. Tolksdorf, Eds.), IEEE Computer Society Press, 1998.

[2] R. De Nicola, G. Ferrari, R. Pugliese. Coordinating Mobile Agents via Blackboards and Access Rights. *Coordination Languages and Models* (COORDINATION'97), *Proceedings* (D. Garlan, D. Le Metayer, Eds.), LNCS 1282, pp. 220-237, Springer, 1997.

[3] R. De Nicola, G. Ferrari, R. Pugliese. Klaim: a Kernel Language for Agents Interaction and Mobility. *IEEE Transactions on Software Engineering*, Vol.24(5):315-330, IEEE Computer Society Press, 1998.

[4] R. De Nicola, G. Ferrari, R. Pugliese, B. Venneri. Types for Access Control. To appear, *Theoretical Computer Science*, 2000.

Secure Implementation of Distributed Languages

Cedric Fournet, Microsoft Research

Joint work with Martín Abadi and Georges Gonthier.

Adopting a programming-language perspective, we study the problem of implementing secrecy and authentication in open distributed systems. Communications processing is a key part of distributed systems, with facilities such as RPC and RMI. For security reasons, messages may require cryptographic operations in addition to ordinary marshalling. More abstractly, these operations can (sometimes) be used to implement a language as a whole, so that every program written in the language can be compiled to a distributed system with strong security properties. We describe such a method that wraps communications processing around an entity with secure local communication, such as a single machine or a protected network. The wrapping extends security properties of local communication to distributed communication. We formulate and analyse the method within a process calculus.

Cryptanalysis of SPIFI II and ENROOT II

Willi Geiselmann, Universität Karlsruhe

Joint work with Feng Bao, Thomas Beth, Robert H. Deng, Claus Schnorr, Rainer Steinwandt and Hongjun Wu.

Banks, Lieman, Shparlinski, and To suggested strengthened versions of their identification scheme SPIFI and their public key encryption system ENROOT at ICISC-2000 in Seoul. Both improved systems make use of sparse polynomials over a finite ring, e. g., the integers modulo n. These systems are based on two hard mathematical problems: finding a *sparse* interpolation polynomial over a fixed set of coefficients and finding common zeros of sparse multivariate polynomials. The transformation of these problems to crytographic protocols adds a lot of structure for the resulting systems. With this additional structure the computations performed throughout the execution of the protocol can be recovered and thus the secret key is revealed. The calculations necessary for these attacks essentially involve basic arithmetic in the integers modulo some number and solving a small system of linear equations; the secret keys can be disclosed within seconds of computing time.

Analysing the Analysis of Cryptographic Protocols

Dieter Gollmann, Microsoft Research

The design of cryptographic protocols has the reputation of being difficult and error-prone. In my view, this is due to the fact that the (informal) goals a security protocol is designed to meet and the environment in which the protocol is expected to operate often remain vague. In this respect, protocol analysis is equally difficult and error-prone when goals and environmental assumptions are picked arbitrarily without taking sufficient care to clarify these aspects before making them precise.

Entity authentication and formal semantics for SDSI are given as examples to demonstrate that formal security verification may be precise about the problem being addressed without being clear about what the problem is. Correspondence properties as used in the formal analysis of authentication protocols may capture that

- the responder is “alive” (see ISO/IEC 9798-1)
- two parties are establishing a secure connection (see ISO/IEC 7498-2)
- the protocol run itself simulates a connection (Roscoe’s canonical intensional specification)

or some other property with no clear relationship to what is called “authentication” somewhere in the security literature. Lowe’s renowned attack against the Needham-Schroeder protocol does not break the first of the goals above, and one could argue that the property he chose to capture authentication for the responder is not a conventional authentication property. Moreover, in his attack the traditional all-powerful attacker who interferes with the traffic between honest principals is replaced by an insider who does not follow the protocol rules.

With SDSI, we find logics with formal semantics that fit the intuitions of access control but do not fully match SDSI name resolution, and logics that do precisely that but have unclear relevance for access control. (We note that the authors of SDSI state access control as their intended application.)

Coping with Denial of Service due to Malicious Java Applets

Roberto Gorrieri, Università di Bologna

In this talk I argue that static approches, widely exploited for improving Java security, are of little use when considering the typical security problems caused by malicious Java applets, e.g., denial of service and antagonism. As a consequence, a monitoring application (called Signed Applet Watch-Dog — SAWD) is proposed to control the execution of malicious Java applets and to stop (or suspend) their execution when some critical conditions on, e.g., resource usage are met. The application is a signed Java applet, to be executed outside of the sandbox, simple to use and easily configurable by the user, because it works like a user interface. SAWD can stop (almost) all the applets causing denial of service (screen obscuration, generation of dozens of windows, usage of all CPU time, etc.) and many causing antagonism (unwanted sounds and images, “work for me” applets, etc.) at the price of a modest degradation of the Web browser. SAWD seems also a necessary tool for software development environments for Java applets, as it offers a much better environment to experiment on applets than the JDK appletviewer.

Modeling Security Goals and Systems for Trust Management

Joshua Guttman, The MITRE Corporation

In this talk, we will stress three main points. First, security is not one thing, but different things in different contexts, so that modeling is needed to determine exactly what class of security goals is relevant. Second, this modeling process suggests algorithms that can be used to determine whether goals are met in even complex systems. Third, trust management to achieve security goals despite complexity is frequently more pressing than the design of isolated security mechanisms. We illustrate the points with reference to packet filtering and the IP security protocols.

Deciding Bisimilarity in the Spi Calculus ***(Results and Directions)***

Hans Hüttel, Aalborg University

Joint work with Josva Kleist, Uwe Nestmann and Björn Victor.

We present results whose common aim is to shed light on the problem of deciding bisimilarity for (a simple version of) Abadi and Gordon's spi calculus. In particular, we show the negative result that even a finite-control spi-calculus, where processes may only contain a fixed number of parallel components, suffices for an encoding of Minsky's two-counter machines. Thus, in contrast to the situation for the finite-control π-calculus, no non-trivial notion of behavioural equivalence can be decidable for finite-control spi calculus processes.

Even for processes without recursion we encounter the problem of infinite branching on inputs. We describe a *symbolic semantics* in the style of Hennessy and Lin. The symbolic semantics captures exactly the transition properties of the environment-sensitive labelled transition semantics proposed by Boreale, De Nicola and Pugliese. We briefly discuss our ongoing work on defining a notion of symbolic bisimulation equivalence.

Model Checking Security Properties of Control Flow Graphs

Thomas Jensen, Université de Rennes, IRISA

Joint work with Frederic Besson, Daniel Le Metayer and Tommy Thorn.

A fundamental problem in software-based security is whether local security checks inserted into the code are sufficient to implement a global security property. This article introduces a formalism based on a linear-time temporal logic for specifying global security properties pertaining to the control flow of the program, and illustrates its expressive power with a number of existing properties. We define a minimalistic, security-dedicated program model that only contains procedure call and run-time security checks and propose an automatic method for verifying that an implementation using local security checks satisfies a global security property. We then show how to instantiate the framework to the security architecture of Java 2 based on stack inspection and privileged method calls.

Quantitative Measures of Security

Erland Jonsson, Chalmers University of Technology

Security is a multifaceted, composite concept. In order for quantitative measures to be attributed to security we have found that it has to be divided into its different aspects: confidentiality, integrity and availability. Since there seems to be an overlap with dependability aspects, an overall integrated approach between security and dependability has to be taken. Thus, we have found that some aspects, the behavioural aspects reliability and availability, describe the system behaviour with respect to its intended (authorized) users. The aspect of confidentiality defines its behaviour with respect to the non-authorized users. In this context safety is a sub-aspect, referring to a failure of the others with catastrophic consequences. Behavioural measures can be derived using traditional Markov models, including phase-type modelling. The integrity aspect, however, defines the ability of the system to protect itself against unauthorized detrimental influence, e.g. intrusions. This aspect can be covered using a protective measure. We have suggested that the effort it takes to make an intrusion be used as a measure of integrity ("intrusion security"). The hypothesis is that "the more effort it takes to make an intrusion the more secure is the system". In order to get some data that would confirm this hypothesis we have carried out a number of intrusion experiments. During these, undergraduate students were asked to attack a system while reporting all the details of how it was done. Data from the experiments have been used for some initial modelling, which shows that it is at least possible in principle to use such experimental data to get measures of intrusion security.

Bridging the gap: Formal vs. Complexity-Theoretical Reasoning about Cryptography

Jan Jürjens, Oxford University

Joint work with Martín Abadi.
We compare two views of symmetric cryptographic primitives in the context of the systems that use them. We express those systems in a simple programming language; each of the views yields a semantics for the language. One of the semantics treats cryptographic operations formally (that is, symbolically). The other semantics is more detailed and computational; it treats cryptographic operations as functions on bitstrings. Each semantics leads to a definition of equivalence of systems with respect to eavesdroppers. We establish the soundness of the formal definition with respect to the computational one. This result provides a precise computational justification for formal reasoning about security against eavesdroppers.

Computationally Secure Information Flow

Peeter Laud, Universität des Saarlandes

We propose a definition of secure information flow in programs. Our definition is not based on noninterference, but on computational indistinguishability, i.e. we do not require that the low security outputs of the program do not depend in any way from the high security inputs, but only require that the dependency is such, that no computationally bounded attacker can make use of it. In this sense our definition is somewhat similar to the definition of semantic security. This definition allows us to handle cryptographic primitives, whose security is usually defined in similar terms — only security against computationally bounded adversaries is required. For example, according to our definition, the program that assigns to a low security variable the result of encrypting the value of a high security variable under a high security key, is deemed secure.

We also give a data flow analysis which allows us to certify programs for secure information flow in our sense. The analysis is quite similar to that of D. Denning. It finds all such subsets of the set of variables (possibly overestimating), for which it holds, that an adversary that knows the final values of the variables in such a subset, has significantly better chances in guessing something about the high security part of the input of the program, than an adversary that has does not know the final values of any variables. If all such subsets also contain high security variables, then the information flow in the program has been certified secure.

An Approach to Information Flow Control

Heiko Mantel, German Research Center for Artificial Intelligence (DFKI)

Information flow properties, like e.g. non-interference, can be used to express confidentiality as well as integrity requirements. In this process, one first identifies a set of security domains and then decides if information may flow between these domains or not. This results in a flow policy. Next, a definition of information flow must be chosen. The common intuition underlying such definitions is that information flows from a domain D1 to a domain D2 if the behaviour of D2 can be affected by actions of D1. However, this intuition can be formalized in different ways and, at least for non-deterministic systems, no agreement on an optimal definition has been reached. Rather a collection of definitions co-exists. In order to deal with this variety, uniform frameworks have been proposed in which the different definitions can be easily compared to each other. One limitation of the definitions of information flow (for non-deterministic systems) which have been proposed so far is, that they cannot cope with intransitive flow policies, i.e. flow policies in which the interference relation is not transitive. However, such policies are necessary in order to deal with concepts like channel control, information filters, or explicit downgrading. In this talk, I will illustrate this problem and propose novel definitions of information flow as a solution. These proposed definitions are compatible with intransitive flow policies.

The results presented in this talk are derived from the following articles.

Heiko Mantel. Possibilistic Definitions of Security - An Assembly Kit -. In Proceedings of the 13th IEEE Computer Security Foundations Workshop, pp. 185-199, IEEE 2000.

Heiko Mantel. Information Flow Control and Applications - Bridging a Gap -. to appear in Proceedings of Formal Methods Europe, Berlin, 2001.

A General Framework for Security Analysis

Fabio Martinelli, IAT-CNR

Joint work with Riccardo Foccardi and Roberto Gorrieri.
We set up a framework to describe and verify both network and system security properties. The idea is that a system should guarantee a secure behavior even under the attack of whatever hostile environments. If we describe the secure behavior as an abstract process then we obtain the so-called GNDC schema. This is based on a suitable extension of the notion of non-interference (NI) to

security protocols. Several existing properties have been recast as specific instances of this schema. Once these security properties are uniformly described then it is easier to compare them. Indeed, we studied several relationships among different security properties, e.g. authentication and NI. Moreover, we can re-use analysis techniques and software tools studied for a property encoded in this schema to deal with other ones in the same schema. For example, the Cosec tool for the compositional verification of Non-interference properties (e.g., NDC) has been extended to deal also with secrecy and authentication properties of cryptographic protocols. Furthermore, we are able to give sufficient conditions for avoiding the necessity of the quantification over the possible hostile environments by considering the most general one. In our framework we can also specify security properties through temporal logic formulas (e.g., mu-calculus). The system is checked against every possible hostile environment which can interact with it. This verification problem, known in concurrency theory as module checking problem, is solved by means of partial model checking techniques and satisfiability procedures for the logic. Our analysis approach works also when the so-called most general intruder approach seems useless, e.g. for checking bisimulation-based non interference properties. A software tool (SecTool) which implements this methodology has been realized and tested over significant case studies.

Checking Security Properties using History-Dependent Automata

Ugo Montanari, University of Pisa

Joint work with Marco Pistore.
History-dependent automata (HD-automata in brief) are an extension of ordinary automata that overcomes their limitations in dealing with history-dependent formalisms. In a history-dependent formalism, the actions that a system can perform refer to events in the past history of the system. The most interesting examples are process description languages equipped with name mobility, like the pi-calculus: channel names can be created by some actions and they can then be referenced by successive actions. Also security protocols can be modeled using HD-automata, since nonces and keys can be represented as fresh names, and protocol verification can be often expressed via semantic equivalence.

The coalgebraic framework developed for classical process algebras, and in particular its advantages concerning minimal realizations, do not fully apply to history-dependent formalisms, due to the constraints on freshly generated names that appear in the definition of bisimilarity.

Recent work by the authors has proposed to model HD-automata (and in particular the transition system of the pi-calculus) as a coalgebra on a category of name permutation algebras and to define its abstract semantics as the final coalgebra on such a category. It is shown that permutations are sufficient to represent in an explicit way fresh name generation, thus allowing for the definition of minimal realizations up to the ordinary notion of bisimilarity.

Flow Logics for Hardest Attackers

Flemming Nielson, Aarhus University

Joint work with Hanne Riis Nielson and René Rydhof Hansen.

Mobile ambients show the way to the next generation internet languages where not only code (like the applets in Java) may move between sites but where actual computations may traverse the internet under their own control. Static analysis is called for to obtain decidable approximations to the behaviour of such computations in all possible execution contexts. The spirit of the flow logic approach is similar to that of type systems and logic specifications but aims at transferring state-of-the-art techniques from data flow analysis, set constraints and abstract interpretation to the more dynamic and concurrent scenario offered by mobile ambients.

In the talk we develop a simple flow logic for mobile ambients showing which ambients may end up inside what other ambients. Based on this we validate a proposed firewall as being impenetrable by all agents not knowing the right passwords. Despite the fact that there are infinitely many such agents, we are able to identify a "hardest attacker" (in the manner of hardest problems in a complexity class) and to prove that the inability of the "hardest attacker" to penetrate the firewall implies the inability of all agents to penetrate the firewall unless they know the passwords.

Cryptographic Security of Reactive Systems

Birgit Pfitzmann, Universität des Saarlandes

Joint work with Matthias Schunter and Michael Waidner.

The overall goal of this work is to provide a cryptographic semantics for "abstract" specifications, so that the "reality" of cryptographic definitions can be combined with the brevity or, if a formal language is used, the precision and tool-support, of abstract specifications.

Speicifically, we present a general definition of security for reactive systems, generalizing previous definitions relying on the simulatability paradigm.

The main novel aspects are a separate treatment of honest users, precise synchronous and asynchronous switching models, and easy inclusion of various trust models. We also believe to have the first general strategy to deal abstractly with tolerable imperfections (such as leakage of traffic patterns), and the first worked-out serious-size examples within a general model. Most importantly, our model has the first reactive composition theorem, and a link to requirements formulated in logics.

More details under `http://www.semper.org/sirene`.

Primitives for Authentication in Process Algebras

Corrado Priami, Università di Verona

Joint work with Chiara Bodei, Pierpaolo Degano and Riccardo Focardi.

We extend the π-calculus and the spi-calculus with two primitives that guarantee authentication. They enable us to abstract from various implementations/specifications of authentication, and to obtain idealized protocols which are "secure by construction". The main underlying idea, originally proposed by

Focardi for entity authentication, is to use the "locations" of processes in order to check who is sending a message (authentication of a party) and who originated a message (message authentication). The theory of local names developed by Bodei, Degano and Priami for the π-calculus, gives us almost for free both the partner authentication primitive and the message authentication one.

Probability-Sensitive Secure Information Flow

Andrei Sabelfeld, Chalmers University of Technology

Joint work with David Sands.
When is an untrusted program safe to use? One aspect of safety is confidentiality. Given you have some confidential (high) data and some public (low) data in your computer, you want to make sure the attacker - the supplier of the untrusted code - will not learn anything about your personal data, despite the fact that the application (e.g., a spreadsheet) may require legitimate access to the confidential data in order to perform its task, and legitimate communication with the supplier of the code (e.g., a registration process for all users).

We assume that the attacker is external to the (trusted) system upon which the program is run. Our aim is to specify when a program is safe to run - from the point of view of its confidentiality properties - with an aim to provide automatic methods for certifying programs prior to execution.

This talk proposes a probability-sensitive confidentiality specification - a form of probabilistic noninterference - for a small multi-threaded programming language with dynamic thread creation. Probabilistic covert channels of information flow arise from a scheduler which is probabilistic. Since scheduling policy is typically outside the language specification for multi-threaded languages, we describe how to generalise the security condition in order to define robust security with respect to a wide class of schedulers, not excluding the possibility of deterministic (e.g., round-robin) schedulers and program-controlled thread priorities. The formulation is based on an adaptation of Larsen and Skou's notion of probabilistic bisimulation. We show how the security condition satisfies compositionality properties which facilitate straightforward proofs of correctness for, e.g., security type systems. We illustrate this by defining a security type-system and proving it correct.

A New Type System for Secure Information Flow

Geoffrey Smith, Florida International University

With the variables of a program classified as L (low, public) or H (high, private), we wish to prevent the program from leaking information about H variables into L variables. Given a multi-threaded imperative language with probabilistic scheduling, the goal can be formalized as a property called Probabilistic Noninterference. Previous work identified a type system sufficient to guarantee Probabilistic Noninterference, but at the cost of severe restrictions: to prevent timing leaks, H variables were disallowed from the guards of while-loops. Here we present a new type system that gives each command a type of the form τ_1 cmd τ_2; this type says that the command assigns only to variables of level τ_1 (or higher) and has running time that depends only on variables of level τ_2 (or lower). Also we use types of the form τ cmd n for commands that terminate in exactly n steps. With these typings, we can prevent timing leaks by demanding that no assignment to an L variable may sequentially follow a command whose running time depends on H variables. As a result, we can use H variables more flexibly; for example, under the new system, a thread that involves only H variables is always well typed.