



# Sensoria

016004

*Software Engineering for Service-Oriented  
Overlay Computers*

## **D3.1a: Techniques for Security and Trust for Services**

Lead contractor for deliverable: ISTI

Author(s): Massimo Bartoletti (PISA), Roberto Lucchi (UNIBO), Fabio Martinelli (ISTI), Fabio Massacci (UNITN), Hanne Riis Nielson (DTU), Davide Sangiorgi (UNIBO), Roberto Zunino (PISA)

Due date of deliverable: September 1, 2006

Actual submission date: October 9, 2006

Revision: V09

Dissemination level: PU

Contract start date: September 1, 2005 Duration: 48 months

Project coordinator: LMU

Partners: LMU, UNITN, ULEICES, UWARSAW, DTU,  
PISA, DSIUF, UNIBO, ISTI, FFCUL, UEDIN, ATX,  
TILab, FAST, BUTE, S&N, LSS-Imperial, LSS-UCL



Integrated Project funded by the  
European Community under the  
“Information Society Technologies”  
Programme (2002—2006)

## **Executive Summary**

This document contains the results on the research activities on techniques for security and trust for services performed in the initial twelve months of the SENSORIA project. These research activities address several topics described in the Task 3.1 of WP3. The number and the variety of the results are due to both the interest of the research topics covered by SENSORIA and the fact that in the initial period of the project researchers mainly contributed to the stated goals of the project, and in particular Task 3.1, by bringing their own experience and methodologies, although always paying attention to techniques exploitable for service oriented architectures.

Thirteen original contributions have emerged as result of this research activity. In the following sections of this document an overview is given of the performed activity and the results are described with a limited level of technical details. The full details of the contributions can be found in the attached papers (see Section 9) which should be considered as an integral part of this deliverable.

## Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>Models and analysis techniques for security and trust</b>	<b>5</b>
<b>3</b>	<b>Static analysis techniques for the analysis of crypto-protocols</b>	<b>7</b>
<b>4</b>	<b>Security issues on shared space coordination languages</b>	<b>9</b>
<b>5</b>	<b>Process interference in mobile systems</b>	<b>10</b>
<b>6</b>	<b>Secure service composition</b>	<b>12</b>
<b>7</b>	<b>Ensuring constraints on interfaces between services</b>	<b>14</b>
<b>8</b>	<b>Autonomic security mechanisms</b>	<b>16</b>
<b>9</b>	<b>Relevant Sensoria Publications and Reports</b>	<b>17</b>
<b>10</b>	<b>Conclusion and future work</b>	<b>18</b>

## 1 Introduction

This document reports on the scientific results obtained by SENSORIA researchers during the first twelve months of the project on techniques for security and trust in service oriented architectures (SOA). In particular, the researchers during their work addressed the research issues identified in Task 3.1. Being such a task inserted in WP3, devoted to qualitative techniques for ensuring the security and trust for services, most of the results actually exploit formal and rigorous tools as, for instance, process calculi, logics, model checking, static analysis techniques to achieve the desired level of confidence on the modeled services.

The current approaches address several issues as confidentiality, integrity, non interference, access control and trust management. Also specific issues related to SOA as service composition have been successfully addressed. The analysis scenarios and threat models discussed range from the common Dolev-Yao model ([8]) to ad hoc scenarios as mobile wireless systems one, where attackers have to obey to domain specific restrictions. On a more practical side, also specific techniques for access control to services have been designed. These techniques are effective and have been successfully implemented.

As a whole, the results are interesting, published in scientifically relevant forums as well as potentially adaptable to future calculi and models developed in the project.

Indeed, the next efforts should be more devoted on focussing the ideas and methodologies developed so far, on the calculi are currently under development by the SENSORIA project in other WPs in order to provide a coherent framework for security and trust for SOA.

Briefly, we summarize the main results.

- *Models and analysis techniques for security and trust.* (Section 2) We aim at defining an integrated framework for the specification and (automated) analysis for security and trust in complex and dynamic scenarios. In particular, we have shown how the same machinery used for the formal verification of security protocols may be used to analyze access control policies based on trust and reputation management. The framework is based on a process calculi (CCS-like) equipped with suitable inference constructs for manipulating messages.
- *Static analysis techniques for the analysis of crypto-protocols.* (Section 3) We developed a static analysis technique for the verification of cryptographic protocols, specified in a process calculus. Rather than assuming a specific, fixed set of cryptographic primitives, we only require them to be specified through a term rewriting system, with no restrictions.
- *Security issues on shared space coordination languages.* (Section 4) We studied the definition and formalization of a coordination infrastructure supporting a secure service interaction. Specific attention is paid to shared data coordination languages.
- *Process interference in mobile systems.* (Section 5) In wireless systems, the communication mechanism combines features of broadcast, synchrony, and asynchrony. We developed an operational semantics for a calculus of wireless systems that takes into account possible communication interferences. We first considered a core calculus, essentially with only the primitives for communication, and then a few extensions.
- *Secure service composition.* (Section 6) A distributed calculus is proposed for describing networks of services. We modeled service interaction through a call-by-property invocation mechanism, by specifying the security constraints that make their composition safe. A static approach is then proposed to determine how to compose services and guarantee that their execution is always secure, without resorting to any dynamic check.
- *Ensuring constraints on interfaces between services.* (Section 7) We developed techniques for ensuring that the constraints part of an interface correctly describes the operation of the code implementing the interface, and develop analysis on top of combinations of constraints for ensuring the relevant security properties.

- *Autonomic security mechanisms.* (Section 8) We developed some autonomic security mechanisms for interactive access control. We have a prototype implementation using logical policies in logic programs interoperable with real digital certificates. We also defined an architecture for fine grained control of computational services based on this framework and on a process calculi.

## 2 Models and analysis techniques for security and trust

### The overall framework

We are interested in the development of integrated modeling and analysis techniques for security and trust for services.

The modeling and the analysis of services oriented architectures, especially of web services, have received a considerable attention among researchers. In particular, the higher is the attention for automation in composition of (web) services, the more the need for expressive languages able to suitably describe their complex interactions. Formal models as process calculi seemed to be a natural candidate for modeling and analyzing web services *per se* as well as their interactions.

One important aspect in services deployment is security. The approach of modeling services with process calculi may be also fruitful since there exist several well-founded theories that allow the modeling and automated analysis of security properties with such process calculi.

Moreover, relevant aspects in distributed environments are trust and reputation management as well as access control. Indeed, in service oriented architectures (SOA), common access control mechanisms mainly based on entity identification do not scale. Thus, new access control mechanisms based on entity attributes have been developed and languages for suitably describing such access control mechanisms and policies have been defined. In such languages, one entity may witness properties of other entities. Thus, decisions about the access to a resource may be based also on trust relationships among the entities. In addition, entities may delegate their own access rights to other entities. Such delegation mechanisms may be expressed by means of credentials issuing. Furthermore, in business oriented applications, interactions (i.e., business services) among entities may be based on trust relationships. Trust can derive both from personal experience and reputation, that can, in turn, be also based on recommendation from others. The capability of representing, evaluating and monitoring trust relationships is thus paramount for a complete and powerful SOA modeling language.

Formal methods and tools have been successfully applied for the analysis of security properties of communication protocols. The protocol under investigation is described in a given language, then a formal specification of the security property to be analyzed is defined. Whether or not the security property is fulfilled is investigated by formally analyzing the protocol in a hostile environment, i.e., considering the presence of an adversary running in parallel with the honest participants.

In process calculi, cryptography is usually modeled by representing encryptions as terms of an algebra, e.g.,  $E(m, k)$  may represent the encryption of a message  $m$  with a key  $k$ . Usually, the so-called perfect encryption abstraction is adopted: encryptions are considered as injective functions which can be inverted only by knowing the correct information, i.e. the decryption key. For instance, common inference rules for modeling the behavior of the encryption and decryption (in a shared-key schema) are the following:

$$\frac{m \quad k}{E(m, k)} \quad \frac{E(m, k) \quad k}{m} \quad (1)$$

which should be read as: from a message  $m$  and a key  $k$  we can build the encryption  $E(m, k)$ ; from an encryption  $E(m, k)$  and a decryption key  $k$  we can obtain the encrypted message  $m$ .

The inference relation could be defined in many ways. This would give a complex inference system. Such inference systems allow us to cope with the variety of different crypto-systems that can be found in the literature.

However, when one analyzes a security protocol, usually assumes that public keys, digital certificates, and generally speaking credentials are already given, and does not usually check how these are created,

negotiated and managed. Such a limited view seems not completely appropriate for dynamic, fully interconnected systems, where access control policies may change and typically may also depend on credentials presented by users. Similarly, when one wishes to formally analyze (e.g., see [7]) access control systems, the authentication mechanisms (usually a security protocol) are considered a priori “secure”, without further specification. While separation of concerns is often desirable, this is not always possible. The interplay between security protocols and access control mechanisms/policies is crucial. Moreover, a good specification and analysis framework should take an holistic point of view.

## Work done

By considering the previous arguments, we focussed on the integrated formal modeling and analysis of security and trust. In particular, we uniformly model security protocols/services and several forms of access control based on trust management.

As a matter of fact, we have shown that the idea proposed by CryptoCCS process calculi (e.g., see [9]) of using inference constructs is also useful to model access control mechanisms based on credentials in distributed systems (e.g., see [22, 23]).

**Example.** Indeed, consider a set of credentials, i.e. (signed) messages containing information about access rights. Assume that  $\{A, ob_1, +\}_{pr(C)}$  means that the user  $C$  (via the signature with its private key  $pr(C)$ ) asserts  $A$  has the right to access the object  $ob_1$  and may grant this access to other users (this is denoted through the symbol  $+$ ). A rule like:

$$\frac{\{A, ob_1, +\}_{pr(C)} \quad pr(C) \quad \{grant \ B, ob_1\}_{pr(A)}}{\{B, ob_1, +\}_{pr(C)}} (acc_C)$$

may be used by the controller  $C$  to issue other access right credentials, after receiving an indication by  $A$ , i.e. the signed message  $\{grant \ B, ob_1\}_{pr(A)}$ .

Thus, we may consider the inference rules as an abstract mechanism to express security policies usually defined using other mathematical models and logics.

In particular, we were able to encode the RT trust management system [21] as well as the trust model proposed in [17]. Such languages allow to model and reason about trust relationships among entities. Having a unique language will allow us to model the interplay between security protocols that use the trust relationships among different users, and the ways in which these relationships are created (that often rely on security/interaction protocols). We presented also an integrated inference system for describing both trust and recommendation relationships. In doing so, we also relate the previously mentioned models of trust. In addition, we present a flexible framework for giving metrics on trust relationships, and we provide a suitable inference system modeling the unified framework that results from our relationships.

It is worthy noticing that the CryptoCCS has been previously defined to set up a uniform framework for the analysis of security properties and information flow (non-interference) with the same machinery (e.g., see [12, 9]). This helped us quite a lot in establishing a precise correspondence of properties of trust negotiation protocols with non-interference ones (as hinted in [33]).

**Authentication properties.** On the overall framework of modeling and analyzing security properties, we investigated more deeply the notion of authentication.

As a matter of fact, property’s formal specification and its informal definition are crucial steps for the analysis. Indeed, even a common notion as authentication is usually considered a slippery security property. Several definitions have been proposed in the security community. Henceforth, to define a formal model for authentication is a tricky task and many different formalisms actually exist (e.g., see [2, 30]). Their expressiveness, usability and accuracy are often questioned, and some effort has been devoted to compare them. In [10] three different authentication properties are compared, and the comparison is carried out within the Generalized Non Deducibility on Compositions schema (GNDC,

[9, 12]). GNDC has been proposed for defining and analyzing security properties and it is based on the notion of non-interference.

We tried to formally capture the meaning of two different flavors of authentication, *credit* and *responsibility*. Abadi first focused his attention on these two properties in [1], arguing about two possible uses of an authenticated message  $m$ . Suppose that a principal A sends the message to a principal B. Then, as far as responsibility is concerned, “B may believe that  $m$  is being supported by A’s authority”. Also, as far as credit is concerned, “B may attribute credit for  $m$  to A”. Abadi expresses the concepts of credit and responsibility through several examples, by discussing their distinction in the design and analysis of protocols. Different cryptographic operators (digital signatures, encryptions, decryptions) may lead either to responsibility or credit. Our investigation starts from the intuitions of Abadi by rephrasing them: we suggest that responsibility supports orders, i.e., public messages which speak for some principal, while credit is related to a principal for secret messages known in advance only by that principal, that wants to be acknowledged as the holder of the secret messages.

We were able to express also this property as a non-interference one, in particular as a GNDC property.

### SENSORIA relevance and future plans

The work described here addresses some relevant topics related to the Workpackage 3 and especially Task 3.1. The work, although very preliminary, has shown how the integrated analysis of security properties as well as trust management ones may be very useful. In particular, we were able to show how non-interference properties in process calculi may be used to model property of trust/service negotiation - a specific feature of SOA. For the future we plan to extend this theory to other calculi as well as to implement its analysis. Also this work is related to the one for autonomic services (see Section 8), since the cryptoCCS languages has been used then to describe the usage control policies for computational resources. We also plan to study specific authentication properties for services. As we have seen the notion of authentication is very slippery and the capability to identify correctly entities in SOA is crucial especially for accounting purposes.

### References to official Sensoria publications

**Paper delivered:** [14] R. Gorrieri, F. Martinelli, and M. Petrocchi. A formalization of credit and responsibility within the gndc schema. *Electr. Notes Theor. Comput. Sci.*, 157(3):61–78, 2006.

**Paper delivered:** [22] F. Martinelli and M. Petrocchi. On relating and integrating two trust management languages. *Electr. Notes Theor. Comput. Sci.* In proc. of the 2<sup>nd</sup> International Workshop on Views On Designing Complex Architectures (VODCA’06) 2006. To appear in ENTCS.

**Paper delivered:** [23] F. Martinelli and M. Petrocchi. A uniform framework for security and trust modeling and analysis with cryptoccs. *Electr. Notes Theor. Comput. Sci.* In proc. of the International Workshop on Computer Security (ICS 06). Timisoara, Romania. September 29-30 2006. To appear in ENTCS. (Invited Talk)

## 3 Static analysis techniques for the analysis of crypto-protocols

### The overall framework.

In the ambit of static analysis techniques, we developed a technique for the verification of cryptographic protocols, specified in a process calculus. Rather than assuming a specific, fixed set of cryptographic primitives, we provide a general framework for handling crypto primitives and their associated algebraic

laws. This enables us to study protocols that use complex crypto primitives, as those for managing protocol sessions.

### Work done.

The so-called perfect encryption assumption is consistent with a formal view of cryptography, where crypto-operations are seen as abstract functions. Nevertheless, in some settings it appears too strict. For instance, session protocols have been designed to guarantee a smooth degradation of security when some component, e.g. a session key is compromised.

In the computational view of cryptography, crypto-operations are seen instead as functions on bit strings. Cryptography is here dealt with more in concrete, using notions of probability and computational power.

In [35] we compared the two approaches and linked them, so to point out when the computational security definitions are implied by the simpler ones found in formal models. When this holds, one can therefore use formal techniques, such as static analysis, to prove some security properties in the more complex computational world. To this purpose, we considered secrecy and authentication properties.

In [36] a technique is defined to check protocols secure. This technique has been implemented in the Rewrite tool: Rewrite is free software and can be found at <http://www.di.unipi.it/~zunino/software>. The tool takes a protocol specification and runs the static analysis developed in [36]: when the analysis succeeds, the protocol is provably safe. The main novelty of the approach lies in the handling of general cryptographic primitives and their algebraic properties. These primitives are not arbitrarily fixed, or hard-coded in the tool, but are specified by the user in a fairly natural way through a rewriting system: the definition of the primitives is then passed to the tool together with the protocol specification. So, complex primitives such as  $\text{exp,*}$ , and  $\text{xor}$  can be handled, and protocols involving them can be verified. Examples are provided to support our analysis, including Diffie-Hellman ( $\text{exp,*}$ ), Kerberos (timestamps), and one-time-pad (XOR). In the Diffie-Hellman example our techniques proved a forward secrecy property, stating that the secrecy of completed sessions of the protocol still holds even if some long-term keys are disclosed. This is a significative smooth degradation property that session-establishing protocols often provide. Our Rewrite tool was able to prove this property without user intervention. Further, in [35] we started adapting our techniques to cryptographic APIs, which are usually more complex than protocols and play an important role in web services.

### Sensoria relevance and future work.

The results are consistent with the goals of the Task 3.1. Indeed, security of web services often depends on the security of the protocols used for exchanging data and establish high-level communication mechanisms, such as sessions. So, it is important to verify the underlying low-level protocols, since a flaw in them can easily compromise the security of web services.

As future work, we plan to extend our techniques for the verification of higher-level communication primitives. The primitives used in the Sensoria core calculus may provide interesting challenges. Also, the development of techniques for API verification would help the design of the web services interfaces.

Moreover, we plan to augment our techniques and tools so that, when a security property is established, an actual proof of that property is produced. Our tools, when verification succeeds, would provide a machine-checkable proof of the security of the system at hand. This would greatly increase the confidence on the established security facts, since then, one could regard the (complex) verification tool as untrusted, and rely on the (simple) proof checker correctness, only.

### References to official Sensoria publications

**Paper delivered:** [35] R. Zunino. *Models for Cryptographic Protocol Analysis*. PhD thesis, University of Pisa, 2006.

**Paper delivered:** [36] R. Zunino and P. Degano. Handling exp,  $\times$  (and timestamps) in protocol analysis. In *9th International Conference on Foundations of Software Science and Computation Structures FoSSaCS*, volume LNCS 3921, pages 413–427, 2006.

## 4 Security issues on shared space coordination languages

### The overall framework

New networking technologies are calling for the definition of models and languages adequate for the design and management of new classes of applications. In particular we consider the challenges emerging when the Service Oriented Computing paradigm is considered. In this case the challenge is to develop service-based applications without knowing, at design time, the overall structure of the system as well as the entities that will be involved. Indeed, these systems are usually open to new entities or components which are unknown beforehand. Furthermore, our society is becoming more and more dependent on computer networks: the enormous amount of data that is elaborated, transmitted or stored needs some form of protection. Hence, in order to guarantee some security properties, several procedures based on cryptography (see, e.g. [31]) have been proposed in the literature. The goals of these protocols cover a large area of applications, e.g. privacy and authentication of the messages, personal identification, digital signatures, electronic money transfer, credit card transactions and many other critical applications. Actually, security protocols and the applications exploiting them are typically based on a channel based topology.

This work represents a study towards the definition and formalization of a coordination infrastructure supporting a secure service interaction. Specific attention is paid to shared data coordination languages.

### Work done

Coordination models and languages, which advocate a distinct separation between the internal behavior of the entities and their interaction, represent a promising approach for the development of applications for this class of dynamic and open systems. For instance, we assist to a renewed interest in data-driven coordination infrastructures originated by Linda [13] as exemplified by recent commercial products, such as JavaSpaces [32] and TSpaces [34], which are two coordination middlewares for distributed Java [15] programming proposed by Sun Microsystem and IBM, respectively. Both proposals exploit the so-called *generative communication* [13]: a sender communicates with one or more receivers through a shared tuple space (TS for short), where emitted tuples are collected; a receiver can read or consume the tuples from the TS; a tuple generated by an agent has an independent existence in the tuple space until it is explicitly withdrawn by a receiver; in fact, after its insertion in TS, a tuple becomes equally accessible to all entities, but it is bound to none. This form of communication is referred to as generative communication because when a datum is produced, it has an existence which is independent of its producer, and it is equally accessible to all entities.

In [11] we outline the main security threats which occur when the Linda coordination model is used in untrusted environments. Then we classify the most interesting solutions available in the literature in two classes by proposing two abstract calculi which implement access control mechanisms in two different manners: *entity-driven* approach and *knowledge-driven* approach. Essentially, the former one exploits the notion of entity to express access permissions and to govern the accesses to the resources, while the latter one uses some reserved information that are required when accessing the resources.

### Sensoria relevance and future work

The secure coordination infrastructure we propose can be used to support the discovery of new services and, in general, for enabling the collaboration among services which are not known at design time. Such

functionality can be particularly useful when programming with core languages (WP 2) whose aim is to compose services.

As future work it could be interesting to investigate how to integrate both languages thus obtaining a complete framework for service composition and coordination.

## References to official Sensoria publications

**Paper delivered:** [11] R. Focardi, R. Lucchi, and G. Zavattaro. Secure shared data-space Coordination Languages: a Process Algebraic survey. *Science of Computer Programming*, 63:3–15, November 2006.

## 5 Process interference in mobile systems

### The overall framework

Wireless communication is a widespread technology for the exchange of messages in Service-Oriented Computing architectures (and more broadly, in distributed systems). The use of wireless technologies has considerable impact over the scenario and the hypothesis under which security and trust properties are studied. For instance, since wireless devices communicate by means of broadcasting, intercepting a message does not mean that message will not be delivered to its intended recipient, as it happens in traditional point-to-point communication. One can, by contrast, make a message unreadable to a given wireless device by interfering with the transmission of that message, i.e., simultaneously transmitting other messages. At present, little exists on the semantics of wireless systems. The objective of our work has been to develop an operational semantics for a calculus of wireless systems in which the possible communication interferences, specific of wireless systems, are correctly described. Interferences occur when two distinct transmissions overlap in space and in time; that is, a device is simultaneously reached by the two transmissions, or parts of them.

### Work done

We have defined a basic calculus of wireless systems, and its associated operational semantics. The most important contribution of this work has been the semantics. Here we had to deal with the peculiarities of wireless systems, which combine features of broadcast, synchrony, and asynchrony, as we outline below.

Wireless devices communicate by *broadcast* of messages. This is quite different from the more conventional wired-based broadcast that we find in networks with Ethernet and that, from a semantic point of view, is well-understood ([29]). First, in Ethernet-like systems broadcasting is global, i.e., the messages transmitted reach all nodes of the system. By contrast, in wireless system broadcasting is local, i.e., a transmission spans over a limited area, called *cell*, and therefore reaches only a – possibly empty – subset of the devices in the system. Second, the communication channel for an Ethernet device is full-duplex; that is, a node can transmit and receive at the same time. As a consequence of full-duplex channels and global broadcasting, interferences between two simultaneous transmissions are immediately detected by transmitters. Thus transmitters know that they have to retransmit their messages, and they do so after a randomly-chosen period of time. This means that interferences in Ethernet-like systems are easy to repair. In a model of these systems it is therefore reasonable to abstract from interferences, i.e., to assume that they do not exist. By contrast, in wireless systems channels are *half-duplex*: a device can either transmit or receive, but cannot do both at the same time. Hence, an interference between two transmissions is only possibly detected by receivers located in the intersection of the cells of the two transmitters. Interference is thus a delicate aspect of wireless systems that has to be handled by means of ad-hoc protocols (e.g., CDMA/CA). Interference is also one of the key aspects to be described by a model of these systems. A third, but semantically less relevant, difference between Ethernet and wireless broadcast is that only the latter uses explicit communication channels.

Wireless systems have also features of *synchrony* that remind us of synchronous languages (e.g., Esterel [6], Statecharts [16], SCCS [25]). Indeed, in a single time unit of a wireless system multiple events can happen; such a simultaneous execution of the events is different from an interleaving of them: only in the former case, for instance, interferences can appear.

Wireless devices locally try to remain synchronized on a common clock so that, for instance, transmissions start at the same time. This is due to the fact that each device incorporates a hardware clock that is employed for physical transmission; however, each hardware clock is generally characterized by a different clock drift with respect to the nominal frequency. Hence, to reduce the effect of the clock drift, each device re-synchronizes its own clock on the transmissions which are performed from the other devices. However, the physical distance among the devices and the partial area covered by each transmission may prevent an exact global agreement on clocks. As a consequence, the clocks of distant devices usually do not coincide. This introduces an additional element of *asynchrony* that affects the way interferences appear and are detected, as we discuss below.

Another – though less important – feature of wireless systems (that a semantics should show) is the fact that a transmitter, before initiating a transmission, checks that, locally, the communication channel (i.e. a specific frequency band) is not presently employed for performing other transmissions. This is imposed in order to reduce the possibilities of interferences.

Our calculus is called *Calculus of Wireless Systems* (CWS). It has nodes, which represent the devices of the system, that can be composed in parallel. Inside a node there is a sequential process, which models the behavior of that device. Each node has a location and a radius that define the cell over which that node can transmit. When developing the semantics for CWS we tried hard to adhere to the standard operational semantics of concurrent systems (broadcast systems as in CBS [28], point-to-point systems as CCS or CSP) in which each transmission of a message (or communication) is represented precisely by a single transition (or reduction) in the semantic. For instance, in the labeled transition semantics of CCS an event  $P \xrightarrow{\bar{a}v} P'$  means that an output process in  $P$  transmits value  $v$  on channel  $a$  and then the process evolves into  $P'$ . Only one among the input processes executing in parallel with  $P$  receives that value. Similarly, in CBS an event  $P \xrightarrow{!v} P'$  means that an output process in  $P$  transmits value  $v$  and then the process evolves into  $P'$ . All the input processes executing in parallel with  $P$  catch that transmission.

Our attempt at following this approach was unsuccessful due to the physical aspects of wireless systems outlined above. In particular, we could not maintain transmission atomicity and, at the same time, express all the possible modes in which interferences may occur.

A natural alternative would have been to follow the approach of synchronous languages. In these languages an event in the semantics represents a set of transmissions, namely all those that occurred during the same time unit. Unfortunately, we failed also with this approach. The reason is that – as explained earlier – the synchrony among the devices of a wireless system is not perfect: the transmission of two distant devices may span time units that are only partially overlapping, with strong consequences on the interference that these devices can cause with each other and with other devices.

We thus decided to refine the view on transmissions and observe, for each node, the change of state between transmission and reception (and vice versa), rather than single transmissions. Further, in accordance with physical wireless devices, we assume that when a device is not performing a transmission its antenna is in reception mode. We call *event* the state change that occurs in the network when a device changes the function of its antenna. Specifically, we call *begin transmission* the event which corresponds to a device which initiates a transmission; and, we call *end transmission* the event which corresponds to a transmitter which finishes its transmission. This choice has also physical justifications: wireless communication protocols require different signal sequences for indicating begin and end of transmissions.

In concurrency theory, *Labelled Transition Systems* (LTS) are the most widely adopted means of giving operational semantics. An LTS can be a basis for defining powerful proof techniques: both inductive techniques, because, for instance, the derivation of a transition in the LTS is driven by the structure of a term; and co-inductive techniques, because, for instance, the transitions of an LTS expose the full behavior of a system (its internal activities as well as the interactions with the environment)

which is needed for defining behavioral equivalences according to the notion of bisimulation. However, sometimes the rules of the LTS can be hard to understand. This happens, for instance, in calculi for mobility such as the  $\pi$ -calculus or in higher-order calculi such as Ambients. Thus, a different form of operational semantics, called *Reduction Semantics* (RS) has been introduced. An RS only gives meaning to the internal activities of a system. For this, the structural operational semantics style is combined with an auxiliary relation of structural congruence that allows manipulation of the term structure so as to bring potential interacting components in contiguous positions. An RS can make the meaning of the calculus or language easier to grasp; and it can be used to check the correctness of an LTS, by proving a consistency with respect to such LTS. On the other hand, the LTS remains superior for defining reasoning techniques on the processes.

For reasons similar to those above, we define both RS and LTS semantics. Besides the different flavor (the fact of being RS or LTS), the two semantics also differ in the approach we followed to check the interferences. In the LTS, the derivation of a transition takes a set of active transmitters (i.e., a set of devices that are currently engaged in a transmission) as a parameter; then, the various possibilities of interferences are checked against such a parameter. In the RS, by contrast, such parameter is absent; the checks for interferences are “hardwired” into the rules of the semantics itself. As a consequence, however, the derivation of a reduction has to be decomposed into three separate sub-derivations, which are defined using some auxiliary relations on an extension of the calculus. Each one of such auxiliary relations implements a logical distinct sub-component of the mechanism with which transmissions are performed in wireless systems (e.g., individuation of the transmitter; individuation of its cell; for each receiver in the cell, the individuation of possible interferences). The main extensions of the calculus are given by markers that are placed on the network nodes and that represent a partial state of the node within the whole reduction.

Our main technical result is the equivalence between the two semantics. The proof differs from existing results of this kind in the literature both because of the differences between standard LTSs and RSs and ours, and because of the different approaches used in our LTS and RS concerning the identification of the active nodes.

### Sensoria relevance and future work

The work described here addresses a specific goal of Task 3.1.

As a matter of fact, the semantics that we have defined for our calculus of wireless systems is fairly complex. For instance, each single rule by itself is very simple, but the number of rules is very high. Also, in the Reduction Semantics a transition is computed by composing four relations, representing intermediate steps of the transition, and each relation is defined by means of separate rules. The most urgent work for the future is to try to simplify the semantics, especially the Reduction Semantics. Such simplifications should also reduce the complexity of the proofs.

Another important work for the future would be the addition of movement. In the present calculus, devices are static – they cannot move. Movement could be added as a separate feature, orthogonal to communication. In other words, a viable approach to movement could consist in adding primitives for expressing movement of devices (the calculus could even be parametric with respect to this); during communication, however, the network would be considered static, and therefore communication could be described using the machinery for static systems.

The long-term objective of the work is to transplant onto wireless systems the techniques for the analysis of security and trust developed within Sensoria.

### References to official Sensoria publications

**Paper delivered:** [24] N. Mezzetti, D. Sangiorgi, *Towards a calculus for Wireless Systems* Electr. Notes Theor. Comput. Sci., 158, Elsevier, 2006

## 6 Secure service composition

### The overall framework

Security plays a crucial role in Service Oriented Computing. In this scenario, applications are built by assembling stand-alone components distributed over a network, called services.

Web Services, built upon XML technologies, are possibly the most illustrative and well developed example of this paradigm. The problem of discovering and composing Web Services by taking advantage of semantic information has been the subject of a considerable amount of research and development. The idea is to extend the primitives of service description languages with constructs for specifying properties of the published interface.

Secure composition and coordination of services may however require specialized mechanisms: indeed, services are open, i.e. built with little or no knowledge about their operating environment, their clients, and further services.

Authentication mechanisms are not enough to guarantee security in the Service-Oriented Computing paradigm. Actually, security may be breached even by trusted services, either because of unintentional behavior (e.g. bugs), or because the composition of the client and the services exhibits some behavior unwanted by the client (e.g. leakage of information). We advocate language-based techniques to recover security in the world of Service Oriented Computing. In particular, we are interested in studying linguistic primitives for the protection of both clients and services.

### Work done

We proposed in [3, 5, 4] a distributed calculus for describing networks of services. We modeled service interaction through a call-by-property invocation mechanism, by specifying the security constraints that make their composition safe. A static approach was then proposed to determine how to compose services and guarantee that their execution is always secure, without resorting to any dynamic check.

In our approach, clients may protect from their callers by wrapping security-critical portions of their own code into *safety framings*. These framings enforce the given security policy on the execution of the wrapped piece of code, aborting it whenever about to violate the policy, thus offering additional flexibility with respect to monolithic *global* policies, and relieving the programmer of guarding each use of security-critical resources.

On their side, callers may constrain the behavior of the called services, by supplying a security policy at the moment of invocation. We push further this invocation mechanism, by allowing callers to request services that not only do obey the imposed security constraints, but that also respect a given contract on their functional behavior. The implementation of this *call-by-property* invocation mechanism requires that services are published together with a *certified* abstraction of their behavior.

To construct this certified abstraction, we have introduced a type and effect system for our calculus. The type of a service describes its functional behavior, while the effect is a *history expression*, representing those histories of events relevant to security. History expressions extend regular expressions with information about the selection of services, coupled with their corresponding effect. To reconstruct the type of a service we exploit the partial visibility that the service has about the other services in the network, and so no global knowledge is assumed.

Another major problem is how to orchestrate a service-based application while guaranteeing that no executions will abort because of some action attempting to violate security. This is called the planning problem, and it amounts to selecting from the network those services that accomplish the requested task, while respecting the security constraints on demand. Those services that locally obey the property imposed by a request are not always good candidates, because their behavior may affect security of the whole composition.

We have devised a way of extracting from a history expression all the *viable* plans, i.e. those that successfully drive secure executions. This is a two-stage construction. A first transformation of history

expressions makes them model-checkable for validity. Valid history expressions guarantee that the services they come from never go wrong at run-time. From valid histories it is then immediate to obtain the viable plans, that make any execution monitor unneeded.

Our approach therefore extends and complements those based on behavioral descriptions, while still fully automating the process of discovering services and planning their composition in a secure way.

An important aspect is that the selection of services should not be based on *names* (e.g. the identity of the service deployer), but on the properties the selected services can be proven to enjoy. The trusted computing base can then be dramatically reduced, because we only have to trust the entity that certifies services, instead of the myriad of entities that deploy services over the web. A further step towards security consists in assessing the trustiness of such a certification entity through formal methods (e.g. type systems).

### Sensoria relevance and future work

This work represents a successful integration of the activities in WP2 on behavioral types, once adapted to the specific needs and goals in WP3 about security techniques for services.

We plan to extend this work in several directions.

We plan to further develop the call-by-property invocation mechanism in [5], by considering other security properties (e.g. information-flow), and non-functional aspects (e.g. transactions, QoS). For instance, we would discard a service if it might leak some client private data.

We also wish to extend our calculus and static machinery so to allow for the creation of new resources at run-time. In the current approach, the set of security-relevant resources is fixed at the time of static analysis, while we wish to be able to analyze services that may dynamically create new resources and pass them around the network.

We also plan to give a more general solution to the planning problem, by also considering dynamic scenarios where services may be published on-the-fly and may become temporarily unavailable - and still devise an efficient, incremental orchestration.

Another research direction is the combination of semantic-based service composition with security protocols and trust relationships. E.g., this would allow for guaranteeing that certified services will not arbitrarily change their code on the fly.

### References to official Sensoria publications

**Paper delivered:** [3] M. Bartoletti, P. Degano, and G. L. Ferrari. Plans for service composition. In *Workshop on Issues in the Theory of Security (WITS)*, 2006.

**Paper delivered:** [4] M. Bartoletti, P. Degano, and G. L. Ferrari. Security issues in service composition. In *The 8th IFIP International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS)*, volume LNCS 4037, 2006. Invited talk.

**Paper delivered:** [5] M. Bartoletti, P. Degano, and G. L. Ferrari. Types and effects for secure service orchestration. In *Proc. 19th Computer Security Foundations Workshop (CSFW)*, 2006.

## 7 Ensuring constraints on interfaces between services

### The overall framework

A central part in service-oriented architectures are the interfaces between services. Services typically impose constraints on how and what they communicate. In this work we develop techniques for ensuring that constraints on interfaces between services correctly describe the operation of the code implementing

the service interface. We further develop analysis on top of combinations of constraints for ensuring relevant safety and security properties.

## The work done

The Flow Logic approach to static analysis amounts to specifying the admissibility of solutions to analysis problems; when specified using formulae in stratified alternation-free least fixed point logic, one may use efficient algorithms for computing the least admissible solutions. We extend this scenario to validate the fulfilment of safety and security constraints on admissible solutions; the modified development produces a least solution together with a boolean value indicating whether or not the constraints are validated or violated.

The main contribution is the development of a deterministic heuristics for obtaining a solution that is close to the least solution, while enforcing the safety or security constraints. We illustrate it on the Bell-LaPadula mandatory access control policy where the heuristics is used to suggest modifications to the security annotations of entities in order for the security policy to hold.

To illustrate our approach, assume a simple language with mandatory access control based on Bell-LaPadula. The operations are `read( $s, o$ )` for indicating that a subject  $s$  reads the object  $o$ , and similarly `write( $s, o$ )` for indicating that subject  $s$  writes to object  $o$ . The discretionary part of the access control policy is specified by statements of the form `readable( $o : s_1, \dots, s_n$ )` and `writable( $o : s_1, \dots, s_n$ )` for indicating that the object  $o$  may read (written) by subjects  $s_1$  to  $s_n$ . The mandatory part is specified by `subject( $s : \phi$ )` for indicating that subject  $s$  has clearance  $\phi$ , and `object( $o : \phi$ )` for indicating that object  $o$  may be accessed at security level  $\phi$ .

Consider a simple program with one subject `sub` and two objects `ob1` and `ob2`. An interface specification might consist of a list of subjects and objects known to the interface, along with constraints imposed on them, e.g.

```
subject(sub:H);
object(ob1:L); readable(ob1:sub); writable(ob1:sub);
object(ob2:L); readable(ob2:sub); writable(ob2:sub);
```

In addition to the explicit constraints there are implicit ones based on the access rights. The interface implementation then works on the elements specified, e.g.

```
read(sub,ob2); write(sub,ob1);
```

As presented in detail in [26], the logical representation of the semantics works on configuration  $(S, O, M, B)$ , where  $S$  records the last security level assigned to subjects,  $O$  records for objects the last security level required to access them,  $M$  records access rights that a subject  $s$  has with respect to an object  $o$ , and  $B$  records access operations initiated by a subject  $s$  on an object  $o$ . Part of the formalism developed in [26] is a formula that needs to be fulfilled in order to for the interface implementation to fulfill the constraints imposed. For the example above the formula does not hold, as `sub`, which has security level `H`, writes a value to an object that has security level `L`.

If the formula component is *not* fulfilled for the least solution of the analysis, [26] develops a heuristic for suggesting remedies. In the example, there are two ways for fulfilling the constraints—either downgrading `sub` to `L`, or upgrading `ob2` to `H`. Since in the security context it usually is preferred to upgrade objects rather than downgrading subjects, for the example the heuristics should allow spurious elements in the  $O$  component of configurations. To ensure this in a flexible way, each component in configurations is associated with a *rank* that represents the acceptability of modifications in that component.

The extension of Flow Logic and the heuristics have been integrated into the Succinct Solver [27], where the heuristics are used to decide how far to backtrack in case a found solution does not fulfill the formula. This allows to efficiently compute close to least solutions that fulfill constraints.

## Sensoria relevance and future work

The work is central for the WP3 task on security and trust in that the techniques developed allow to check that the code implementing service interfaces is correct with respect to the safety and security constraints describing the interface. The techniques developed work on Flow Logic specifications and it is planned to develop these specifications for the calculi developed in the SENSORIA project to check (constraints on) interfaces between services.

## References to official Sensoria publications

**Paper delivered:** [26] F. Nielson and H. R. Nielson. Heuristics for safety and security constraints. *Festschrift dedicated to Gordon Plotkin's 60. birthday, 2006.*

# 8 Autonomic security mechanisms

## The overall framework

Autonomic communication and computing is the new paradigm for dynamic service integration over a network. An autonomic network crosses organizational and management boundaries and is provided by entities that see each other as partners that need to collaborate with little known or even unknown parties. For many services no autonomic partner may guess a priori what will be sent by clients and clients may not know a priori what credentials are demanded for completing a service.

To solve this problem we discuss in this section a novel interactive access control model: servers should be able to get back to clients asking for missing or exceeding credentials, whereas the clients may supply or decline the requested credentials. The process iterates until a final decision of grant or deny is taken.

This model can then be used to monitor the behavior of applications in a Java Virtual Machine (JVM) where we implemented an architecture for fine grained control of computational services based on this framework and on a variant of the cryptoCCS languages. In this way, when the application encounters a security exceptions can instead asks for additional credentials to different services.

## The work done

This proposal is grounded in a formal model on policy-based access control. It identifies the automated reasoning services of deduction and abduction. Based on them, it proposes a comprehensive access control framework for stateless and statefull autonomic systems. The framework describes two interactive access control algorithms for stateless and statefull services and shows the technical guarantees they provide.

On top of the interactive access control, we have developed an interactive trust negotiation model. The negotiation model protects privacy and security interests with respect to information disclosure and access control on client and server sides. It allows two entities in a network to automatically negotiate the requirements to access a service.

We have further developed a proof-of-concept implementation of the entire negotiation framework. The X.509 standard for public key certificates is used for describing participants identities and access rights, the SAML standard for access control specification is used for authorization statements among participants and the DLV datalog system is used as a back-end engine for the basic functionalities of deduction and abduction.

As we said we have then integrated the negotiation mechanism into a framework for monitoring Java applications and their access to system/network resources through *syscall* instructions. The architecture consists of a policy enforcement point, which detects the system call performed by the JVM and an integrated policy evaluation point.

The policy language used for negotiation has been extended to a process-algebra based language to take into account the notion of parallel or sequential constraints that cannot be captured by a purely declarative policy. The complete language is based on the cryptoCCS process algebra has been developed. This language is used to model allowed sequence of system call from JVM. The formal semantics provided by the process algebra and inference systems have been very useful for developing a simple and clear framework whose policies can be formally analyzed.

### Sensoria relevance and future work

The work directly addresses an item of the WP3 (this topic has been moved from Task 3.2 to Task 3.1.). In addition, it also represents a nice integration of the formal specification techniques described in Section 2 developed by ISTI with the work performed at UNITN on access control techniques.

Building upon the good results of the first year, we plan to propose and analyse algorithms for full fledged negotiation between clients and servers and investigate the interoperability of the SENSORIA proposals with other approaches.

### References to official Sensoria publications

**Paper delivered:** [19] H. Koshutanski and F. Massacci. Interactive access control and trust negotiation for autonomic systems. In *Book on Advances in Enterprise Information Technology Security*. IDEA, In revision.

**Paper delivered:** [18] H. Koshutanski, F. Martinelli, P. Mori, and A. Vaccarelli. Fine-grained and history-based access control with trust management for autonomic services. 2006. In proc. of the International Conference on Autonomic Systems, IEEE press. (Best paper award.)

## 9 Relevant Sensoria Publications and Reports

Below we give the list of scientific contributions supported by the SENSORIA project.

1. M. Bartoletti, P. Degano, and G. L. Ferrari. Plans for service composition. In *Workshop on Issues in the Theory of Security (WITS)*, 2006.
2. M. Bartoletti, P. Degano, and G. L. Ferrari. Security issues in service composition. In *The 8th IFIP International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS)*, volume LNCS 4037, 2006. Invited talk.
3. M. Bartoletti, P. Degano, and G. L. Ferrari. Types and effects for secure service orchestration. In *Proc. 19th Computer Security Foundations Workshop (CSFW)*, 2006.
4. R. Focardi, R. Lucchi, and G. Zavattaro. Secure shared data-space Coordination Languages: a Process Algebraic survey. *Science of Computer Programming*, 63:3–15, November 2006.
5. R. Gorrieri, F. Martinelli, and M. Petrocchi. A formalization of credit and responsibility within the gndc schema. *Electr. Notes Theor. Comput. Sci.*, 157(3):61–78, 2006.
6. H. Koshutanski, F. Martinelli, P. Mori, and A. Vaccarelli. Fine-grained and history-based access control with trust management for autonomic services. 2006. In proc. of the International Conference on Autonomic Systems, IEEE press. (Best paper award.)
7. H. Koshutanski and F. Massacci. Interactive access control and trust negotiation for autonomic systems. In *Book on Advances in Enterprise Information Technology Security*. IDEA, In revision.

8. F. Martinelli and M. Petrocchi. On relating and integrating two trust management languages. *Electr. Notes Theor. Comput. Sci.* In proc. of the 2<sup>nd</sup> International Workshop on Views On Designing Complex Architectures (VODCA'06) 2006. To appear in ENTCS.
9. F. Martinelli and M. Petrocchi. A uniform framework for security and trust modeling and analysis with cryptoccs. *Electr. Notes Theor. Comput. Sci.* In proc. of the International Workshop on Computer Security (ICS 06). Timisoara, Romania. September 29-30 2006. To appear in ENTCS. (Invited talk.)
10. N. Mezzetti and D. Sangiorgi. Towards a calculus for wireless systems. In *Proc. MFPS '06*, volume 158 of *Electr. Notes Theor. Comput. Sci.*, pages 331–354. Elsevier, 2006.
11. F. Nielson and H. R. Nielson. Heuristics for safety and security constraints. *Festschrift dedicated to Gordon Plotkin's 60. birthday*, 2006.
12. R. Zunino. *Models for Cryptographic Protocol Analysis*. PhD thesis, University of Pisa, 2006.
13. R. Zunino and P. Degano. Handling  $\exp$ ,  $\times$  (and timestamps) in protocol analysis. In *9th International Conference on Foundations of Software Science and Computation Structures FoSSaCS*, volume LNCS 3921, pages 413–427, 2006.

## 10 Conclusion and future work

The results described nicely address most of the topics relative to Task 3.1. For the future we will continue these lines of research. In particular, we will focus on languages and architectures that are being developed in other work-packages, in particular the core calculi developed in WP 2.

We will continue to study techniques for the *integrated specification of security and trust* for services described by means of languages using features for SOC as developed in the project. The idea is to be able to model security protocols using trust relationships as well as mechanisms to build such trust relationships.

In the area of *security protocol analysis techniques* we will continue the application of static analysis techniques to security protocols. In particular, we will develop static analysis for validating cryptographic protocols expressed in one or more of the SENSORIA calculi (including extensions of Klaim). Specific analysis techniques will be developed for information flow and trust properties (also by considering timing issues).

In addition, we shall develop techniques for verifying cryptographic protocols, with possible causality based semantics. We shall also investigate the automatic generation of proofs of security properties for protocols. We shall extend our analysis tools to provide actual machine-checkable correctness proofs for protocols. Techniques will be adapted to make this feasible. Such proofs will greatly improve the confidence of the security results obtained through the tools.

We will study *secure composition of services*, from a language-based perspective. In particular, we will focus on static mechanisms for enabling call-by-contract invocation primitives, and for enforcing information-flow security policies.

*Autonomic* issues on security are of most importance for the project. We thus plan to continue our research activities for building refined algorithms for security negotiations for autonomic web services. As a matter of fact, we plan to propose and analyse algorithms for full fledged negotiation between clients and servers and investigate the interoperability of the current proposal with others such as *Traust* (e.g., see [20]). This will be an extension of previous work on the our prototype for interactive access control to actually prove that the negotiation algorithm that we have developed can interoperate under some mild conditions on the security policies with other competitive approaches with a less strong logical foundations as our.

## References

- [1] M. Abadi. Two facets of authentication. In *CSFW1998*, pages 27–33. IEEE Press, 1998.
- [2] M. Abadi. Security protocols and specifications. In *Proc. Foundations of Software Science and Computation Structures*, volume 1578 of *Lecture Notes in Computer Science*, pages 1–13, 1999.
- [3] M. Bartoletti, P. Degano, and G. L. Ferrari. Plans for service composition. In *Workshop on Issues in the Theory of Security (WITS)*, 2006.
- [4] M. Bartoletti, P. Degano, and G. L. Ferrari. Security issues in service composition. In *The 8th IFIP International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS)*, volume LNCS 4037, 2006. Invited talk.
- [5] M. Bartoletti, P. Degano, and G. L. Ferrari. Types and effects for secure service orchestration. In *Proc. 19th Computer Security Foundations Workshop (CSFW)*, 2006.
- [6] G. Berry and G. Gonthier. The estereel synchronous programming language: Design, semantics, implementation. *Science of Computer Programming*, 19(2):87–152, 1992.
- [7] P. A. Bonatti and P. Samarati. Logics for authorizations and security. In *Logics for Emerging Applications of Databases*, LNCS. Springer-Verlag, 2003.
- [8] D. Dolev and A. Yao. On the security of public key protocols. In *Proceedings of the IEEE 22nd Annual Symposium on Foundations of Computer Science*, pages 350–357, 1981.
- [9] R. Focardi, R. Gorrieri, and F. Martinelli. Non interference for the analysis of cryptographic protocols. In *Proceedings of 27th International Colloquium in Automata, Languages and Programming*, volume 1853 of *Lecture Notes in Computer Science*, pages 354–372, 2000.
- [10] R. Focardi, R. Gorrieri, and F. Martinelli. A comparison of three authentication properties. *Theoretical Computer Science*, 291(3):285–327, 2003.
- [11] R. Focardi, R. Lucchi, and G. Zavattaro. Secure shared data-space Coordination Languages: a Process Algebraic survey. *Science of Computer Programming*, 63:3–15, November 2006.
- [12] R. Focardi and F. Martinelli. A uniform approach for the definition of security properties. In *Proceedings of World Congress on Formal Methods (FM'99)*, volume 1708 of *Lecture Notes in Computer Science*, pages 794–813, 1999.
- [13] D. Gelernter. Generative Communication in Linda. *ACM Transactions on Programming Languages and Systems*, 7(1):80–112, 1985.
- [14] R. Gorrieri, F. Martinelli, and M. Petrocchi. A formalization of credit and responsibility within the gndc schema. *Electr. Notes Theor. Comput. Sci.*, 157(3):61–78, 2006.
- [15] J. Gosling, B. Joy, and G. Steele. *The Java Language Specification*. Addison-Wesley, 1996.
- [16] D. Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, June 1987.
- [17] A. Josang, E. Gray, and M. Kinateder. Analysing topologies of transitive trust. In *Proc. of the 1<sup>st</sup> workshop on Formal Aspects in Security and Trust (FAST2003)*, 2003.
- [18] H. Koshutanski, F. Martinelli, P. Mori, and A. Vaccarelli. Fine-grained and history-based access control with trust management for autonomic services. 2006. In *proc. of the International Conference on Autonomic Systems*, IEEE press.

- [19] H. Koshutanski and F. Massacci. Interactive access control and trust negotiation for autonomic systems. In *Book on Advances in Enterprise Information Technology Security*. IDEA, In revision.
- [20] A. J. Lee, M. Winslett, J. Basney, and V. Welch. Traust: a trust negotiation-based authorization service for open systems. In *SACMAT '06: Proceedings of the eleventh ACM symposium on Access control models and technologies*, pages 39–48, New York, NY, USA, 2006. ACM Press.
- [21] N. Li, W. H. Winsborough, and J. C. Mitchell. Distributed credential chain discovery in trust management. *Journal of Computer Security*, 1:35–86, 2003.
- [22] F. Martinelli and M. Petrocchi. On relating and integrating two trust management languages. *Electr. Notes Theor. Comput. Sci.* In proc. of the 2<sup>nd</sup> International Workshop on Views On Designing Complex Architectures (VODCA'06) 2006. To appear in ENTCS.
- [23] F. Martinelli and M. Petrocchi. A uniform framework for security and trust modeling and analysis with cryptoccs. *Electr. Notes Theor. Comput. Sci.* In proc. of the International Workshop on Computer Security (ICS 06). Timisoara, Romania. September 29-30 2006. To appear in ENTCS.
- [24] N. Mezzetti and D. Sangiorgi. Towards a calculus for wireless systems. In *Proc. MFPS '06*, volume 158 of *Electr. Notes Theor. Comput. Sci.*, pages 331–354. Elsevier, 2006.
- [25] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, 25(3):267–310, 1983.
- [26] F. Nielson and H. R. Nielson. Heuristics for safety and security constraints. *Festschrift dedicated to Gordon Plotkin's 60. birthday*, 2006.
- [27] F. Nielson, H. R. Nielson, and H. Seidl. A succinct solver for alfp. *Nordic J. of Computing*, 9(4):335–372, 2002.
- [28] K. Ostrovsky, K. V. S. Prasad, and W. Taha. Towards a primitive higher order calculus of broadcasting systems. In *PPDP '02: Proceedings of the 4th ACM SIGPLAN international conference on Principles and practice of declarative programming*, pages 2–13, 2002.
- [29] K. V. S. Prasad. A calculus of broadcasting systems. *Sci. Comput. Program.*, 25(2-3):285–327, 1995.
- [30] P. Ryan, S. Schneider, M. Goldsmith, G. Lowe, and B. Roscoe. *The Modelling and Analysis of Security Protocols: the CSP Approach*. Addison-Wesley, 2001.
- [31] B. Schneier. *Applied Cryptography*. Wiley, 1996.
- [32] Sun Microsystems, Inc. *JavaSpaces<sup>TM</sup> Service Specification*, 2002. <http://www.sun.com/jini/specs/>.
- [33] W. H. Winsborough and N. Li. Safety in automated trust negotiation. In *IEEE Symposium on Security and Privacy*. 2004.
- [34] P. Wyckoff, S. McLaughry, and D. Ford. TSpaces. *IBM System Journal*, August 1998.
- [35] R. Zunino. *Models for Cryptographic Protocol Analysis*. PhD thesis, University of Pisa, 2006.
- [36] R. Zunino and P. Degano. Handling exp, \* (and timestamps) in protocol analysis. In *9th International Conference on Foundations of Software Science and Computation Structures FoSSaCS*, volume LNCS 3921, pages 413–427, 2006.