Security issues in contract-based computing

Massimo Bartoletti[1] and Roberto Zunino[2]

[1] Dipartimento di Matematica e Informatica, Università degli Studi di Cagliari, Italy
[2] Dipartimento di Ingegneria e Scienza dell'Informazione, Università di Trento, Italy

Abstract. We propose a computational paradigm for service-oriented applications, where the interactions among services are driven by *contracts*. A contract is a commitment between two or more parties, which specifies the duties and the rights of the parties involved therein. We study the logical foundations of contracts, through an intuitionistic logic extended with a "contractual" form of implication. This logic is decidable, so we can mechanically infer the consequences deriving from any set of contracts. Several security issues can be explored, among which: how to detect when a contract is violated, how to single out the responsible of a violation, how to take countermeasures against violations. New research directions are then proposed to cope with these issues.

1 Introduction

A crucial aspect of service-oriented applications is how to regulate the interaction between clients and services, so to guarantee to each party that it will obtain the desired behaviour from the other parties. Typical service infrastructures are focussed on protecting services from undesired interactions, while little effort is devoted to protecting clients. Ideally, client and services should agree on a common protocol, making explicit their duties and expectations. This can be done by making each party advertise a *contract*, that subordinates the behaviour promised by a client (e.g. "I will pay for a service X") to the behaviour promised by a service (e.g. "I will provide you with a service Y"), and *vice versa*. Contracts are then first-order citizens in this paradigm: they can be exchanged between services, used to decide which actions to take, inspected to detect violations, and possibly contested to invoke third parties for taking recovery actions.

A foundational problem is then how to formalise contracts. First, this would enable parties to exchange non-repudiable, digitally signed promises. Second, formalising contracts would allow us to answer the question "are these contracts sufficient to guarantee the property X?". Third, when a violation occurs, we could inspect the contracts and single out the actual responsible party.

To give the intuition about contracts, suppose there are two kids, Alice and Bob, who want to play together. Alice has a toy airplane, while Bob has a bike. Both Alice and Bob wish to play with each other's toy. Before sharing their toys, Alice and Bob stipulate the following "gentlemen's agreement":

Alice: I will lend my airplane to you, Bob, provided that I borrow your bike.
Bob: I will lend my bike to you, Alice, provided that I borrow your airplane.

We want to formally deduce that Alice and Bob will indeed share their toys, provided they are real "gentlemen" who always respect their promises. Let us write a for the atomic proposition "Alice lends her airplane" and b for "Bob lends his bike". A (wrong) formalisation of the above commitments in classical propositional logic could be the following, using implication $\rightarrow$. Alice's commitment A is represented as $\mathsf{b} \rightarrow \mathsf{a}$ and Bob's commitment as $\mathsf{a} \rightarrow \mathsf{b}$. While the above commitments agree with our intuition, they are not enough to deduce that Alice will lend her airplane and Bob will lend his bike. Formally, it is possible to make true the formula $A \wedge B$ by assigning false to both propositions a and b.

The failure to represent scenarios like the one above seems related to the the Modus Ponens rule: to deduce b from $\mathsf{a} \rightarrow \mathsf{b}$, we need to prove a. That is, we could deduce that Bob lends his bike, but only *after* Alice has lent Bob her airplane. So, one of the two parties must "take the first step". In a logic for mutual agreements, we would like our logic able to deduce $\mathsf{a} \wedge \mathsf{b}$ whenever $A \wedge B$ is true, without requiring any party to take the first step. To this aim, we introduce a new form of "contractual" implication, which we denote with the symbol $\twoheadrightarrow$. For instance, the contract declared by Alice, "I will lend my airplane to Bob provided that Bob lends his bike to me", will be written $\mathsf{b} \twoheadrightarrow \mathsf{a}$. Actually, the following formula is a theorem of our logic:

$$(\mathsf{b} \twoheadrightarrow \mathsf{a}) \ \wedge \ (\mathsf{a} \twoheadrightarrow \mathsf{b}) \ \rightarrow \ \mathsf{a} \wedge \mathsf{b} \tag{1}$$

In other words, from the "gentlemen's agreement" stipulated by Alice and Bob, we can deduce that the two kids will indeed share their toys. In Section 2 we will briefly present our logic and some of its main properties.

Our core logic for contracts does not make explicit the identity of the participant who is advertising a contract. E.g., in (1) the contract $\mathsf{a} \twoheadrightarrow \mathsf{b}$ does not mention Bob, but simply states the promise, implicitly modelling the fact that Bob is authoritative for that contract (Bob can do b). In more complex scenarios, we would like to write *Bob says* $\mathsf{a} \twoheadrightarrow \mathsf{b}$, to make explicit the name of who is issuing a contract. In Section 3 we will further discuss this issue, as well as some other issues that require further investigation in contract-based computing.

2 A logic for contracts

We propose an extension of intuitionistic propositional logic IPC, called propositional contract logic (PCL). PCL features a new form of implication, which we denote with the symbol $\twoheadrightarrow$. The proof system of PCL comprises the axioms of IPC, the Modus Ponens rule, and the following additional axioms:

$\top \twoheadrightarrow \top$	Zero
$(p \twoheadrightarrow p) \rightarrow p$	Fix
$(p' \rightarrow p) \rightarrow (p \twoheadrightarrow q) \rightarrow (q \rightarrow q') \rightarrow (p' \twoheadrightarrow q')$	PrePost

Back to the example of Sect. 1, the axioms of PCL allow us to deduce the agreement between Alice and Bob, i.e. (1) is a theorem of PCL. Some generalisations of this "handshaking" are also provable. For instance, a sort of "greedy" handshaking holds, where a party promises p_i only provided that *all* the other parties promise their duties, i.e. $p_1, \ldots, p_{i-1}, p_{i+1}, \ldots, p_n$:

$$\vdash \bigwedge_{i \in 1..n} \Big((p_1 \wedge \ldots \wedge p_{i-1} \wedge p_{i+1} \wedge \ldots \wedge p_n) \twoheadrightarrow p_i \Big) \;\rightarrow\; p_1 \wedge \cdots \wedge p_n$$

We can also prove a "circular" handshaking, where the i-th party promises p_i only provided that the (circularly) preceding party promises p_{i-1}:

$$\vdash (p_1 \twoheadrightarrow p_2) \wedge \cdots \wedge (p_{n-1} \twoheadrightarrow p_n) \wedge (p_n \twoheadrightarrow p_1) \;\rightarrow\; p_1 \wedge \cdots \wedge p_n$$

Several interesting properties follow from the axioms of PCL, among which:

$$\begin{array}{ll}
\vdash (p \twoheadrightarrow q) \wedge (q \twoheadrightarrow r) \rightarrow (p \twoheadrightarrow r) & \vdash (p \twoheadrightarrow q) \rightarrow (p \rightarrow q) \\
\vdash (p \twoheadrightarrow q) \wedge (q \rightarrow q') \rightarrow (p \twoheadrightarrow q') & \vdash q \rightarrow (p \twoheadrightarrow q) \\
\vdash (p' \rightarrow p) \wedge (p \twoheadrightarrow q) \rightarrow (p' \twoheadrightarrow q) & \vdash (p \twoheadrightarrow q) \rightarrow ((q \rightarrow p) \rightarrow q) \\
\vdash (p \twoheadrightarrow q) \wedge (q \twoheadrightarrow r) \rightarrow (p \twoheadrightarrow (q \wedge r)) & \vdash (p \twoheadrightarrow (q \wedge r)) \rightarrow (p \twoheadrightarrow q) \wedge (p \twoheadrightarrow r) \\
\vdash (p \twoheadrightarrow q) \vee (p \twoheadrightarrow r) \rightarrow (p \twoheadrightarrow (q \vee r)) & \vdash p \twoheadrightarrow \top
\end{array}$$

Theorem 1. *The logic* PCL *is consistent, i.e.* $\nvdash \bot$.

The following formulae are *not* tautologies of PCL :

$$\begin{array}{ll}
\nvdash (p \rightarrow q) \rightarrow (p \twoheadrightarrow q) & \nvdash (p \twoheadrightarrow q) \rightarrow q \\
\nvdash \bot \twoheadrightarrow p & \nvdash ((q \rightarrow p) \rightarrow q) \rightarrow (p \twoheadrightarrow q)
\end{array}$$

Note that if we augment our logic with the axiom of excluded middle, then $(p \twoheadrightarrow q) \leftrightarrow q$ becomes a theorem, so making contractual implication trivial. For this reason we use IPC, instead of classical logic, as the basis of PCL .

A main result about PCL is its decidability. To prove that, we have devised a Gentzen-style sequent calculus, which is equivalent to the Hilbert-style axiomatisation. In particular, we have extended the sequent calculus for IPC presented in [4], with the following rules to deal with the connective $\twoheadrightarrow$:

$$\frac{\Gamma \vdash q}{\Gamma \vdash p \twoheadrightarrow q}\textsc{Zero} \qquad \frac{\Gamma, p \twoheadrightarrow q, r \vdash p \quad \Gamma, p \twoheadrightarrow q, q \vdash r}{\Gamma, p \twoheadrightarrow q \vdash r}\textsc{Fix} \qquad \frac{\Gamma, p \twoheadrightarrow q, a \vdash p \quad \Gamma, p \twoheadrightarrow q, q \vdash b}{\Gamma, p \twoheadrightarrow q \vdash a \twoheadrightarrow b}\textsc{PrePost}$$

Cut elimination holds for PCL; we have proved this in full details in [1].

Theorem 2. *If p is provable in* PCL *, then there exists a proof of p which does not use the* Cut *rule.*

Decidability then follows from the subformula property, which is enjoyed by our Gentzen rules, and by the cut elimination theorem:

Theorem 3. *The logic* PCL *is decidable.*

As a further support to our logic, we have implemented a proof search algorithm, which decides if any given formula is a tautology or not. In [1] we have proved further properties of PCL , among which equivalence of the Hilbert and the Gentzen systems, the subformula property, and some relations between PCL and IPC, the modal logic S4, and propositional lax logic. Also, we have explored further interesting properties and application scenarios for our logic.

3 Future Research Directions

Our investigation on contracts is still at its beginnings, and in future work we plan to study, along with logics for contracts, programming languages that exploit their features. In particular, we will develop process calculi to describe the behaviour of services in the presence of contracts and attackers. The main features of these calculi will be the possibility of publishing and stipulating contracts, deciding whether a given formula is on duty, and taking recovery actions in the case a contract is not respected. We plan to develop analysis techniques to formally and automatically prove the correctness of the service infrastructure, i.e. that the contracts are always respected, without the need for resorting to third parties external to the model.

We expect that many useful features can be added to our logic, to make it more suitable for modelling complex scenarios. First, we could introduce predicates and quantifiers. This will allow us to model more accurately several scenarios, where a party issues a "generic" contract that can be matched by many parties. While this first order extension shall force us to drop the decidability result, we expect to find interesting decidable fragments of the logic, through which modelling many relevant situations.

We will consider extending our logic with a *says* modality, similarly to [3]. This will enable us to write, e.g. *Alice says* $(\mathsf{b} \twoheadrightarrow \mathsf{a})$ to represent the fact that Alice has issued that contract. Back to our example of Sect. 1, one could expect a handshaking of the following form:

$$\textit{Alice says}\ (\mathsf{b} \twoheadrightarrow \mathsf{a}) \wedge \textit{Bob says}\ (\mathsf{a} \twoheadrightarrow \mathsf{b}) \ \rightarrow\ \textit{Alice says}\ \mathsf{a} \ \wedge\ \textit{Bob says}\ \mathsf{b}$$

in which the duties of Alice and Bob are made clear. This additional information can be exploited by a third party (a sort of "automated" judge) which has to investigate the responsibilities of various parties, in the unfortunate case that a contract is not respected. For instance, if our automated judge is given the evidence that Alice's airplane has never been lent to Bob, from the above he will infer that $(\textit{Alice says}\ \mathsf{a}) \wedge \neg\mathsf{a}$, hence *Alice says* $\bot$, meaning that Alice has not respected her contract and can be prosecuted for that. We now model an attack, where an adversary maliciously issues a "fake" contract, making a promise that he cannot actually implement. Consider e.g. the following buyer-seller scenario:

$$\begin{aligned} \mathit{Seller} &= \forall \mathit{item}, \mathit{cust}, \mathit{addr} : \mathsf{pay}(\mathit{item}, \mathit{cust}, \mathit{addr}) \twoheadrightarrow \mathsf{ship}(\mathit{item}, \mathit{addr}) \\ \mathit{Bob} &= \mathsf{ship}(\mathsf{drill}, \mathsf{bobAddress}) \twoheadrightarrow \mathsf{pay}(\mathsf{drill}, \mathsf{Bob}, \mathsf{bobAddress}) \end{aligned}$$

Assume now that the adversary wants to maliciously exploit the seller contract, in order to receive a free item, and make the unaware customer Bob pay for it:

$$\mathit{FakeBob} \;=\; \mathsf{ship}(\mathsf{10Kdiamond}, \mathsf{fakeAddress}) \twoheadrightarrow \mathsf{pay}(\mathsf{10Kdiamond}, \mathsf{Bob}, \mathsf{fakeAddress})$$

Joining the seller and the attacker contracts will then cause an unwelcome situation for Bob, who is due to pay for a 10K diamond, shipped to the adversary:

$$\mathit{Seller} \wedge \mathit{FakeBob} \rightarrow \mathsf{pay}(\mathsf{10Kdiamond}, \mathsf{Bob}, \mathsf{fakeAddress}) \wedge \mathsf{ship}(\mathsf{10Kdiamond}, \mathsf{fakeAddress})$$

Revisiting our example with the *says* modality, we would deduce:

$$\mathit{Seller} \wedge \mathit{Bob} \;\rightarrow\; \mathit{Bob\ says}\ \mathsf{pay}(\mathsf{drill}, \mathsf{Bob}, \mathsf{bobAddress})$$

In this case, we have a successful transaction, because Bob is stating that he will pay for his drill. Instead, joining the seller and the attacker contracts produces:

$$\mathit{Seller} \wedge \mathit{FakeBob} \;\rightarrow\; \mathit{FakeBob\ says}\ \mathsf{pay}(\mathsf{10Kdiamond}, \mathsf{Bob}, \mathsf{fakeAddress})$$

Now, it is easy to realize that someone has attempted a fraud, because the principal who has signed the contract (FakeBob) is different from that who is due to pay (Bob).

Another possible future direction for our logic would be that of extending its axioms with those of propositional lax logic [2]. This would allow for establishing further properties of contracts, which are not implied by the current PCL axioms, e.g. $(a \twoheadrightarrow c) \wedge (b \twoheadrightarrow d) \;\rightarrow\; (a \wedge b \;\twoheadrightarrow\; c \wedge d)$.

Time is another useful feature that may arise while modelling real-world scenarios. For instance, in an e-commerce transaction, a contract may state that if the customer returns the purchased item within 10 days from the purchase date, then she will have a full refund within 21 days from then. We would like to model such a contract in a temporal extension of our logic, so to reason about the obligations that arise when the deadlines expire. There are a number of techniques aimed at dealing with time in logical systems, so we expect to be able to reuse some of them for extending PCL .

Acknowledgements. Work partially supported by EU-FETPI Global Computing Project IST-2005-16004 SENSORIA and by the MIUR-PRIN project SOFT.

References

1. Massimo Bartoletti and Roberto Zunino. A logic for contracts. Technical Report DISI-09-034, DISI - Università di Trento, 2009.
2. Matt Fairtlough and Michael Mendler. Propositional lax logic. *Information and Computation*, 137(1):1–33, 1997.
3. Deepak Garg and Martín Abadi. A modal deconstruction of access control logics. In *Proc. FoSSaCS*, pages 216–230, 2008.
4. Frank Pfenning. Structural cut elimination - I. intuitionistic and classical logic. *Information and Computation*, 157(1/2):84–141, 2000.