Verifying Persistent Security Properties ⋆

Annalisa Bossi [a] Riccardo Focardi [a] Carla Piazza [a]
Sabina Rossi [a]

[a] *Dipartimento di Informatica, Università Ca' Foscari di Venezia*
{bossi,focardi,piazza,srossi}@dsi.unive.it

Abstract

We study bisimulation-based information flow security properties which are *persistent*, in the sense that if a system is secure then all of its reachable states are secure too. We show that such properties can be characterized in terms of *bisimulation-like* equivalence relations, between the full system and the system prevented from performing confidential actions. Moreover, we provide a characterization of such properties in terms of *unwinding conditions* which demand properties of individual actions. These two different characterizations naturally lead to efficient methods for the verification and construction of secure systems. We also prove several *compositionality* results, that allow us to check the security of a system by only verifying the security of its subcomponents.

1 Introduction

The protection of confidential data from undesired accesses is a typical security issue concerning both systems and networks. Inside a system, information is typically protected via some access control policy, limiting accesses of entities (such as users or processes) to data. There are different levels of flexibility of access control policies depending on the possibility for one entity to change the access rights of its own data. As an example, UNIX gives users complete control on the policy, i.e., every user may decide to make her own information either secret or public. On the other hand, there are *mandatory* policies in which entities have no control on the access rights. For example, *Multilevel*

⋆ This work is a revised and extended version of [1], and has been partially supported by MIUR project "Modelli formali per la sicurezza", the EU project MyThS (IST-2001-32617) and the FIRB project (RBAU018RCZ) "Interpretazione astratta e model checking per la verifica di sistemi embedded".

Security [2] imposes that entities and data are associated to (ordered) security levels and no access to data at higher levels is ever possible, even if the owner of the data is willing to reveal them. These strong mandatory security policies have been designed to avoid internal attacks performed by the so called *Trojan Horse* programs, i.e., malicious software that, once executed by a user, modifies the access rights of the data belonging to such a user. Unfortunately, even when direct access to data is forbidden by (strong) security policies, it might be the case that data are *indirectly* leaked by Trojan Horses which might exploit some observable system side-effects like, e.g., the CPU load or, more in general, the space/time availability of shared resources. (see, e.g., [3,4]).

The necessity of controlling information flow as a whole (both direct and indirect) motivated Goguen and Meseguer in introducing the notion of *Non-interference* [5,6]. Non-Interference formalizes the absence of information flow within deterministic systems. Given a system in which *confidential* (i.e., high level) and *public* (i.e., low level) information may coexist, *non-interference* requires that confidential inputs never affect the outputs on the public interface of the system, i.e., never interfere with the low level users. If such a property holds, one can conclude that no information flow is ever possible from high to low level.

A possibilistic security property can be regarded as an extension of non-interference to non-deterministic systems. Starting from Sutherland [7], various such extensions have been proposed, e.g., [8–18]. Most of these properties are based on *traces*, i.e., the behavior of systems is modelled through the set of their execution sequences. Examples are *non-inference* [15], *generalized non-interference* [11], *restrictiveness* [11], and the *perfect security property* [18].

In [8], Focardi and Gorrieri express the concept of non-interference in the *Security Process Algebra* (*SPA*) language, in terms of bisimulation semantics. In particular, inspired by [17], they introduce the notion of *Bisimulation-based non Deducibility on Compositions* (*BNDC*): a system E is *BNDC* if what a low level user sees of the system is not modified (in the sense of the bisimulation semantics) by composing any high level process Π with E. The main advantage of *BNDC* with respect to trace-based properties is that it is powerful enough to detect information flows due to the possibility, for a high level malicious process, to block or unblock a system. In particular, in [8,19], it is shown that a malicious process may build a channel from high to low, by suitably blocking and unblocking some system services accessible by low level users. The system used to build this covert channel turns out to be secure for trace-based properties. This motivates the use of more discriminating equivalences such as bisimulation.

Non-interference properties, like *BNDC*, provide formal definitions of information flow security and, as a consequence, are useful in order to well understand

and reason about system and network security. In this paper we also approach the problem of automatically checking *BNDC*-like properties, which is useful in many respects. First, as discussed in [20], having efficient automated checkers is useful to test a property against non-trivial system specifications. It is important to check on many examples that what is intuitively considered insecure is correctly rejected by the property. It is also crucial to verify that the property is not stronger than expected, and accepts as secure what is intuitively so. An automated tool is a good way for observing properties at work.

Moreover, there are some cases in which it is possible to analyze specifications that are strictly related to the "real world". An interesting example is the analysis of security protocols, i.e., simple distributed algorithms based on cryptography. They are simple to specify using process calculi like *SPA*, since they are characterized by little local computation and some message exchanges. In [21] it is shown how to use *BNDC*-like properties to check many different network security properties like, e.g., secrecy and authentication. The main idea is that *BNDC* allows us to check whether or not a malicious enemy is able to interfere on the correct (expected) protocol execution. In this setting, the automated verification of *BNDC* allows to either discover flaws on protocol or validate (finite instances of) them.

Although Martinelli [22] has shown that a class of *BNDC*-like properties is decidable over finite state processes, the problem of efficiently verifying *BNDC* is still open. Indeed, decidability of *BNDC* is still an open problem. The main difficulty consists of getting rid of the universal quantification on high level processes Π. A way to overcome this problems is to adopt sufficient conditions for *BNDC*. We recall from [19,23] two of them, named *Strong BNDC* (*SBNDC*, for short) and *Persistent_BNDC* (*P_BNDC*, for short) [1] . Indeed, *P_BNDC* is interesting *per se*, since it has been proposed for analysing systems in dynamic contexts. Intuitively, *P_BNDC* is a persistent version of *BNDC* in which every reachable state is (*BNDC*) secure. In [23] it is shown that this property is suitable when some abstract form of mobility is considered. If a process moves to a different execution environment (e.g., a different host) in the middle of its computation, then we have to be guaranteed that such an intermediate state is still secure. Requiring, from the beginning, that every reachable state is secure trivially guarantees that every possible migration will be done in a secure state.

In the literature there are two different characterizations of security properties that do not require the universal quantification over high level processes Π. They allow us to exploit two different verification techniques:

[1] In [23], *P_BNDC* is shown to be equivalent to the *SBSNNI* property of [19].

(*i*) *Bisimulation-based* characterizations are based on a bisimulation-like equivalence relation between the system E to be analysed and the low level view of the system itself, denoted by $E\backslash H$, i.e., the system E prevented from performing confidential actions. These characterizations allow us to exploit very efficient techniques for verifying the properties over finite-state processes, by using existing algorithms for the verification of strong bisimulation.

(*ii*) *Unwinding conditions* demand properties of individual actions. They aim at "distilling" the local effect of performing high level actions and are useful to define both proof systems (see, e.g., [24]) and *refinement* operators that preserve security properties, as done in [25]. Proof systems allow to incrementally build systems which are secure by construction. Similarly, refinement operators are useful in a stepwise development process, since properties which have been already investigated in some phase need not to be re-investigated in later phases.

In this paper, we start by considering the two characterizations above for *P_BNDC*, given in [24]. By studying the relation between such two characterizations, we generalize them to a parametric security property called *s_BNDC*, where parameter s specifies the way high level actions and internal actions are treated in the underlying bisimulation relation. We show that the *SBNDC* property, which was originally defined through unwinding conditions, is an instance of *s_BNDC*. This directly gives a new bisimulation-based characterization for *SBNDC* property. As a next step, we investigate the compositionality of *P_BNDC* and *SBNDC*. Compositionality is useful for both verification and synthesis: if a property is preserved when systems are composed, then the analysis may be performed on subsystems and, in case of success, the system as a whole can be proved to satisfy the desired property. We notice that both *P_BNDC* and *SBNDC* are compositional with respect to the parallel operator, but they are not *fully* compositional, since they are not preserved by the non-deterministic choice operator. In particular, when we build a system that may (non-deterministically) choose to behave as one of two secure subsystems, we could obtain an insecure system. As also observed in [26], this seems to be counterintuitive. We approach this issue by introducing a new security property, named *Progressing P_BNDC* (*PP_BNDC*), strictly stronger than *P_BNDC*, which is fully compositional, i.e., it is compositional also with respect to the non-deterministic choice. We show that *PP_BNDC* is an instance of the parametric property *s_BNDC* and can be thus expressed both in terms of a bisimulation-like equivalence and through unwinding conditions.

We also consider the specific problem of automatically checking our persistent security properties. In particular, we describe two methods for determining whether a system is *P_BNDC*, *SBNDC* or *PP_BNDC*. The first method is based on the derivation of *Characteristic Formulae* [27,28] in the language of modal μ-calculus [29] (see Section 6.1). The characteristic formulae can be automatically verified using model checkers for μ-calculus, such as NCSU

Concurrency Workbench [30]. Even if in the worst case this method has an exponential time complexity in the number of states of the process, it is still usable in many cases, and has the advantage of reducing the check of security properties to the standard problem of verifying a μ-calculus formula. The second method (see Section 6.2) is in the spirit of [28]: it is based on the computation of a sort of transitive closure (*Closure up to high level actions*) of the system and on the verification of a *Strong Bisimulation*. This allows us to use existing verification tools, since many different algorithms for computing the largest strong bisimulation between two processes (e.g, [31–34]) have been integrated in model checkers, such as NCSU Concurrency Workbench, XEVE [35], FDR2 [36]. In particular, this second approach improves on the polynomial time complexity of the Compositional Security Checker (CoSeC) presented in [20], since only one bisimulation test is necessary.

The paper is organized as follows. In Section 2, we introduce some basic notions on the *SPA* language and the security properties *BNDC* and *P_BNDC*. We recall the two characterizations of *P_BNDC* in terms of a bisimulation-like equivalence relation and an unwinding condition. In Section 3 we introduce a parametric security property named *s_BNDC* in terms of bisimulation and we prove that it can be equivalently characterized in terms of a parametric unwinding condition. *P_BNDC* is just an instance of *s_BNDC*. In Section 4, we show that property *SBNDC* is an instance of *s_BNDC* and provide a bisimulation-based characterization of it. In Section 5, we introduce the class of *PP_BNDC* processes, which is again an instance of *s_BNDC*, and prove that it is fully compositional. In Section 6, we propose two methods to prove our persistent security properties and we demonstrate some complexity results. Finally, in Section 7 we discuss related works and draw some conclusions.

2 Basic Notions

In this section we report the syntax and semantics of the *Security Process Algebra* (*SPA*, for short) [19] and the definition of the security properties *BNDC* [8] and *P_BNDC* [23] together with some main results [24].

2.1 The SPA Language

The *Security Process Algebra* [19] is a variation of Milner's CCS [37], where the set of visible actions is partitioned into high level actions and low level ones in order to specify multilevel systems. SPA syntax is based on the same elements as CCS that is: a set $\mathcal{L}$ of *visible* actions such that $\mathcal{L} = I \cup O$ where $I = \{a, b, \ldots\}$ is a set of *input* actions and $O = \{\bar{a}, \bar{b}, \ldots\}$ is a set of *output*

actions; a special action τ which models internal computations, i.e., not visible outside the system; a complementation function $\bar{\cdot} : \mathcal{L} \rightarrow \mathcal{L}$, such that $\bar{\bar{a}} = a$, for all $a \in \mathcal{L}$. $Act = \mathcal{L} \cup \{\tau\}$ is the set of all *actions*. The set of visible actions is partitioned into two sets, H and L, of high and low actions such that $\overline{H} = H$ and $\overline{L} = L$.

The syntax of SPA *terms* (or *processes*) is defined as follows:

$$E ::= \mathbf{0} \mid a.E \mid E + E \mid E|E \mid E \setminus v \mid E[f] \mid Z$$

where $a \in Act$, $v \subseteq \mathcal{L}$, $f : Act \rightarrow Act$ is such that $f(\bar{\alpha}) = \overline{f(\alpha)}$, $f(\tau) = \tau$, $f(H) \subseteq H \cup \{\tau\}$, and $f(L) \subseteq L \cup \{\tau\}$, and Z is a constant that must be associated with a definition $Z \stackrel{\text{def}}{=} E$.

Intuitively, $\mathbf{0}$ is the empty process that does nothing; $a.E$ is a process that can perform an action a and then behaves as E; $E_1 + E_2$ represents the nondeterministic choice between the two processes E_1 and E_2; $E_1|E_2$ is the parallel composition of E_1 and E_2, where executions are interleaved, possibly synchronized on complementary input/output actions, producing an internal action τ; $E \setminus v$ is a process E prevented from performing actions in v; [2]; $E[f]$ is the process E whose actions are renamed *via* the relabelling function f.

We denote by $\mathcal{E}$ the set of all SPA processes and by $\mathcal{E}_H$ the set of all high level processes, i.e., those constructed only using actions in $H \cup \{\tau\}$.

The operational semantics of SPA agents is given in terms of *Labelled Transition Systems* (*LTS*, for short). A LTS is a triple $(S, A, \rightarrow)$ where S is a set of states, A is a set of labels (actions), $\rightarrow \subseteq S \times A \times S$ is a set of labelled transitions. The notation $(S_1, a, S_2) \in \rightarrow$ (or equivalently $S_1 \stackrel{a}{\rightarrow} S_2$) means that the system can move from the state S_1 to the state S_2 through the action a. The operational semantics of SPA is the *LTS* $(\mathcal{E}, Act, \rightarrow)$, where the states are the terms of the algebra and the transition relation $\rightarrow \subseteq \mathcal{E} \times Act \times \mathcal{E}$ is defined by structural induction as the least relation generated by the inference rules depicted in Figure 1. In Section 6.2 we use also the notion of *rooted* labelled transition system which is a LTS augmented with a distinguish node, the root.

The concept of *observation equivalence* is used to establish equalities among processes and it is based on the idea that two systems have the same semantics if and only if they cannot be distinguished by an external observer. This is obtained by defining an equivalence relation over $\mathcal{E}$. The *weak bisimulation* relation [37] equates two processes if they are able to mutually simulate their behavior step by step. Weak bisimulation does not care about internal τ actions.

[2] In CCS the operator $\setminus$ requires that the actions of $E \setminus v$ do not belong to $v \cup \bar{v}$.

We will use the following auxiliary notations. If $t = a_1 \cdots a_n \in Act^*$ and $E \xrightarrow{a_1} \cdots \xrightarrow{a_n} E'$, then we write $E \xrightarrow{t} E'$. We also write $E \stackrel{t}{\Longrightarrow} E'$ if $E(\xrightarrow{\tau})^* \xrightarrow{a_1} (\xrightarrow{\tau})^* \cdots (\xrightarrow{\tau})^* \xrightarrow{a_n} (\xrightarrow{\tau})^* E'$ where $(\xrightarrow{\tau})^*$ denotes a (possibly empty) sequence of τ labelled transitions. If $t \in Act^*$, then $\hat{t} \in \mathcal{L}^*$ is the sequence gained by deleting all occurrences of τ from t. As a consequence, $E \stackrel{\hat{a}}{\Longrightarrow} E'$ stands for $E \stackrel{a}{\Longrightarrow} E'$ if $a \in \mathcal{L}$, and for $E(\xrightarrow{\tau})^* E'$ if $a = \tau$ (note that $\stackrel{\tau}{\Longrightarrow}$ requires at least one τ labelled transition while $\stackrel{\hat{\tau}}{\Longrightarrow}$ means zero or more τ labelled transitions). We say that E' is reachable from E when there exists t such that $E \xrightarrow{t} E'$.

The notion of *weak bisimulation* is defined as follows.

Definition 2.1 (Weak Bisimulation) *A binary relation $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ over agents is a* weak bisimulation *if $(E, F) \in \mathcal{R}$ implies, for all $a \in Act$,*

- *if $E \xrightarrow{a} E'$, then there exists F' such that $F \stackrel{\hat{a}}{\Longrightarrow} F'$ and $(E', F') \in \mathcal{R}$;*
- *if $F \xrightarrow{a} F'$, then there exists E' such that $E \stackrel{\hat{a}}{\Longrightarrow} E'$ and $(E', F') \in \mathcal{R}$.*

Two agents $E, F \in \mathcal{E}$ are weakly bisimilar*, denoted by $E \approx F$, if there exists a weak bisimulation $\mathcal{R}$ containing the pair (E, F).*

The relation $\approx$ is the largest weak bisimulation and it is an equivalence relation [37].

A *Rooted Labelled Transition System* is a LTS augmented with a distinguished node, the root. Given a process E we denote by $LTS(E) = (S_E, E, Act, \rightarrow)$ the rooted LTS constituted of the subpart of the SPA LTS reachable from E. E is a *finite-state* process if $LTS(E)$ has a finite number of nodes, that is S_E is finite.

2.2 Security Properties

The *BNDC* [8] security property aims at guaranteeing that no information flow from the high to the low level is possible, even in the presence of malicious processes. The main motivation is to protect a system also from internal attacks, which could be performed by the so called *Trojan Horse* programs, i.e., programs that are apparently honest but hide inside some malicious code. Property *BNDC* is based on the idea of checking the system against all high level potential interactions, representing every possible high level malicious program. In particular, a system E is *BNDC* if for every high level process Π a low level user cannot distinguish E from $(E|\Pi)$, i.e., if Π cannot interfere

with the low level execution of the system E. In other words, a system E is *BNDC* if what a low level user sees of the system is not modified by composing any high level process Π to E.

Definition 2.2 (BNDC) *Let $E \in \mathcal{E}$.*

$$E \in BNDC \quad \textit{iff} \quad \forall\, \Pi \in \mathcal{E}_H,\ E \setminus H \approx (E|\Pi) \setminus H.$$

Example 2.3 *The* BNDC *property is powerful enough to detect information flows due to the possibility for a high level malicious process to block or unblock a system. Let $H = \{h\}$, $L = \{l, j\}$ and $E_1 = l.h.j.\mathbf{0} + l.j.\mathbf{0}$. Consider the process $\Pi = \bar{h}.\mathbf{0}$. We have that $(E_1|\Pi) \setminus H \approx l.j.\mathbf{0}$, while $E_1 \setminus H \approx l.\mathbf{0} + l.j.\mathbf{0}$. Note that the latter may (nondeterministically) block after the l input. Having many instances of this process, a low level user could deduce if $\bar{h}$ is executed by observing whether the system always performs j or not. Process E_1 may be "repaired", by including the possibility of choosing to execute j or not inside the process. Indeed, process $E_2 = l.h.j.\mathbf{0} + l.(\tau.j.\mathbf{0} + \tau.\mathbf{0})$ is* BNDC. □

In [23], it is introduced a security property called *Persistent_BNDC* (*P_BNDC*, for short), which is suitable for analysing systems in dynamic execution environments. Intuitively, a system E is *P_BNDC* if it never reaches insecure states.

Definition 2.4 (P_BNDC) *Let $E \in \mathcal{E}$.*

$$E \in P\_BNDC \quad \textit{iff} \quad \forall\ E' \textit{ reachable from } E,\ E' \in BNDC.$$

We show the idea of *P_BNDC* through a simple example.

Example 2.5 *Consider the process E_2 of Example 2.3, i.e., $E_2 = l.h.j.\mathbf{0} + l.(\tau.j.\mathbf{0} + \tau.\mathbf{0})$ where $l, j \in L$ and $h \in H$. Suppose now that E_2 is moved in the middle of a computation. This might happen when it find itself in the state $h.j.\mathbf{0}$ (after the first l is executed). Now it is clear that this process is not secure, as a direct causality between h and j is present. In particular $h.j.\mathbf{0}$ is not* BNDC *and this gives evidence that E_2 is not* P_BNDC. *The process may be "repaired" as follows: $E_3 = l.(h.j.\mathbf{0} + \tau.j.\mathbf{0} + \tau.\mathbf{0}) + l.(\tau.j.\mathbf{0} + \tau.\mathbf{0})$. It may be proved that E_3 is* P_BNDC. *Note that, from this example it follows that* P_BNDC $\subset$ BNDC. □

In [23] it has been shown that even if the definition of *P_BNDC* introduces an universal quantification over all the possible reachable states, this can be avoided by including the idea of "being secure in every state" inside the bisimulation equivalence notion. This is done by defining an equivalence notion which just focus on observable actions which do not belong to H. More in details, it is defined an observation equivalence, named *weak bisimulation up to H* where

actions from H are allowed to be ignored, i.e., they are allowed to be matched by zero or more τ actions. To this aim, the following transition relation is used.

Definition 2.6 *Let $a \in Act$. We define the transition relation $\stackrel{\hat{a}}{\Longrightarrow}_{\setminus H}$ as follows:*

$$\stackrel{\hat{a}}{\Longrightarrow}_{\setminus H} = \begin{cases} \stackrel{\hat{a}}{\Longrightarrow} & \text{if } a \notin H \\ \stackrel{a}{\Longrightarrow} \text{ or } \stackrel{\hat{\tau}}{\Longrightarrow} & \text{if } a \in H \end{cases}$$

Note that the relation $\stackrel{\hat{a}}{\Longrightarrow}_{\setminus H}$ is a generalization of the relation $\stackrel{\hat{a}}{\Longrightarrow}$ used in the definition of weak bisimulation [37]. In fact, if $H = \emptyset$, then for all $a \in Act$, $E \stackrel{\hat{a}}{\Longrightarrow}_{\setminus H} E'$ coincides with $E \stackrel{\hat{a}}{\Longrightarrow} E'$.

The concept of *weak bisimulation up to H* is defined as follows.

Definition 2.7 (Weak Bisimulation up to H) *A binary relation $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ over agents is a* weak bisimulation up to H *if $(E, F) \in \mathcal{R}$ implies, for all $a \in Act$,*

- *if $E \stackrel{a}{\rightarrow} E'$, then there exists F' such that $F \stackrel{\hat{a}}{\Longrightarrow}_{\setminus H} F'$ and $(E', F') \in \mathcal{R}$;*
- *if $F \stackrel{a}{\rightarrow} F'$, then there exists E' such that $E \stackrel{\hat{a}}{\Longrightarrow}_{\setminus H} E'$ and $(E', F') \in \mathcal{R}$.*

Two agents $E, F \in \mathcal{E}$ are weakly bisimilar up to H, *written $E \approx_{\setminus H} F$, if $(E, F) \in \mathcal{R}$ for some weak bisimulation $\mathcal{R}$ up to H.*

The relation $\approx_{\setminus H}$ is the largest weak bisimulation up to H and it is an equivalence relation. In [23] *P_BNDC* has been characterized in terms of $\approx_{\setminus H}$ as stated below.

Theorem 2.8 (P_BNDC - Bisimulation [23]) *Let $E \in \mathcal{E}$. $E \in$ P_BNDC if and only if $E \approx_{\setminus H} E \setminus H$.*

In [24] we give a further characterization of *P_BNDC* processes in terms of *unwinding conditions*. This new characterization provides a better understanding of the operational semantics of *P_BNDC* processes. In practice, whenever a state E' of a *P_BNDC* process may execute a high level action moving to a state E'', then E' should be also able to simulate such high move through a τ sequence moving to a state E''' which is equivalent to E'' for a low level user.

Theorem 2.9 (P_BNDC - Unwinding [24]) *Let $E \in \mathcal{E}$ be a process. $E \in$ P_BNDC iff for all E' reachable from E, if $E' \stackrel{h}{\rightarrow} E''$, then $E' \stackrel{\hat{\tau}}{\Longrightarrow} E'''$ and $E'' \setminus H \approx E''' \setminus H$.*

Here we observe that there is a strict relation between the bisimulation-based characterization of *P_BNDC* given in Theorem 2.8 and the unwinding condition of Theorem 2.9: the equivalence $\approx_{\setminus H}$ between E and $E \setminus H$ in Theorem 2.8 states that high level actions of E are simulated by zero or more τ actions of $E \setminus H$, while the unwinding condition in Theorem 2.9 says that for every high level action there must exists a path of zero or more τ actions leading to equivalent states from the low level view. This suggests us that consistent changes in the way of dealing with high level actions in $\approx_{\setminus H}$ and in the corresponding unwinding condition, may lead to different bisimulation-like and unwinding characterizations of novel information flow security properties.

This idea will be exploited in the next sections when we study the properties *SBNDC* and *PP_BNDC*.

In [23] it is also proved that *P_BNDC* is compositional with respect to the parallel composition, restriction and low level prefix operators.

Proposition 2.10 ([23]) *Let $E, F \in \mathcal{E}$. If $E, F \in$ P_BNDC, then*

- *$a.E \in$ P_BNDC, for all $a \in L \cup \{\tau\}$;*
- *$(E|F) \in$ P_BNDC;*
- *$E \setminus v \in$ P_BNDC, for all $v \subseteq \mathcal{L}$;*
- *$E[f] \in$ P_BNDC* [3] *.*

Unfortunately, *P_BNDC* is not compositional with respect to the nondeterministic choice operator as illustrated below.

Example 2.11 *Let $E_4 = h.\mathbf{0}$ with $h \in H$ and $E_5 = l.\mathbf{0}$ with $l \in L$. It is easy to see that both E_4 and E_5 are* P_BNDC *but $E_4 + E_5$ is not* P_BNDC. *This example will be further illustrated in the next section.* □

3 A Generalization

In this section we generalize both the notion of weak bisimulation up to high level actions of Definition 2.7 and the unwinding condition expressed by Theorem 2.9, by making them parametric with respect to a parameter $s \in \{*, 0, +\}$. Then, we introduce a parametric security property, named *s_BNDC*, by generalizing the quantification-free characterization given by Theorem 2.8 for *P_BNDC* processes. Finally, we prove that *s_BNDC* processes can be equivalently defined by means of the generalized unwinding condition. This result is used in the next sections to provide a quantification-free characterization of

[3] This last item is not in [23], but it is immediate to prove it.

SBNDC [8] and of a novel, fully compositional property, named *Progressing P_BNDC*.

We introduce the following binary relations on processes which are parametric with respect to a parameter $s \in \{*, 0, +\}$.

Definition 3.1 *Let $s \in \{*, 0, +\}$. The transition relations $\stackrel{a}{\Longrightarrow}^s$ and $\stackrel{a}{\hookrightarrow}^s$ are defined as follows:*

$$\stackrel{a}{\Longrightarrow}^s = \begin{cases} \stackrel{\hat{a}}{\Longrightarrow} & \text{if either } s = * \text{ or } s = 0 \\ \stackrel{a}{\Longrightarrow} & \text{if } s = + \end{cases}$$

$$\stackrel{a}{\hookrightarrow}^s = \begin{cases} \stackrel{a}{\Longrightarrow}^s & \text{if } a \notin H \\ \stackrel{a}{\Longrightarrow} \text{ or } (\stackrel{\tau}{\longrightarrow})^s & \text{if } a \in H \end{cases}$$

where $(\stackrel{\tau}{\longrightarrow})^s$ stands for $\stackrel{\hat{\tau}}{\Longrightarrow}$ if $s = *$, for $\stackrel{\tau}{\Longrightarrow}$ if $s = +$, and for a sequence of zero actions [4] *, if $s = 0$.*

Since $\stackrel{\hat{a}}{\Longrightarrow}$ and $\stackrel{a}{\Longrightarrow}$ coincide for $a \in \mathcal{L}$, the various instances of $\stackrel{a}{\Longrightarrow}^s$ are different only when $a = \tau$. In this case both $\stackrel{\tau}{\Longrightarrow}^*$ and $\stackrel{\tau}{\Longrightarrow}^0$ represent a sequence of 0 or more τ actions while $\stackrel{\tau}{\Longrightarrow}^+$ represents a sequence with at least one τ.

Fact. The relation $\stackrel{a}{\hookrightarrow}^*$ coincides with $\stackrel{\hat{a}}{\Longrightarrow}_{\setminus H}$ of Definition 2.6.

Example 3.2 *Let $E \equiv \tau.h.\tau.l.\mathbf{0}$ with $h \in H$ and $l \in L$. We have $E \stackrel{h}{\Longrightarrow}^s l.\mathbf{0}$ for all $s \in \{*, 0, +\}$, $E \stackrel{\tau}{\Longrightarrow}^s h.\tau.l.\mathbf{0}$ for all $s \in \{*, 0, +\}$, $E \stackrel{\tau}{\Longrightarrow}^s \tau.h.\tau.l.\mathbf{0}$ for $s = *$ and $s = 0$, but $E \not\stackrel{\tau}{\Longrightarrow}^+ \tau.h.\tau.l.\mathbf{0}$. Moreover, $E \stackrel{h}{\hookrightarrow}^s l.\mathbf{0}$ for all $s \in \{*, 0, +\}$, $E \stackrel{h}{\hookrightarrow}^s \tau.h.\tau.l.\mathbf{0}$ for $s = *$ and $s = 0$ but $E \not\stackrel{h}{\hookrightarrow}^s \tau.h.\tau.l.\mathbf{0}$ for $s = +$, $E \stackrel{h}{\hookrightarrow}^s h.\tau.l.\mathbf{0}$ for $s = *$ and $s = +$ but $E \not\stackrel{h}{\hookrightarrow}^s h.\tau.l.\mathbf{0}$ for $s = 0$.* $\square$

The following definition generalizes the notion of weak bisimulation up to H.

Definition 3.3 (s-Weak Bisimulation up to H) *A binary relation $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ over agents is a s-*weak bisimulation up to H *if $(E, F) \in \mathcal{R}$ implies, for all $a \in Act$,*

- *if $E \stackrel{a}{\rightarrow} E'$, then there exists F' such that $F \stackrel{a}{\hookrightarrow}^s F'$ and $(E', F') \in \mathcal{R}$;*
- *if $F \stackrel{a}{\rightarrow} F'$, then there exists E' such that $E \stackrel{a}{\hookrightarrow}^s E'$ and $(E', F') \in \mathcal{R}$.*

[4] If $E(\stackrel{a}{\longrightarrow})^0 E'$ then E coincides with E'.

Two agents $E, F \in \mathcal{E}$ *are* s-weakly bisimilar up to H, *written* $E \approx^s_{\setminus H} F$, *if* $(E, F) \in \mathcal{R}$ *for some* s*-weak bisimulation* $\mathcal{R}$ *up to* H.

It is easy to see that all the three instances of $\approx^s_{\setminus H}$ are equivalence relations and that $\approx^s_{\setminus H}$ is the largest s-weak bisimulation.

The notion of s-weak bisimulation up to H can be used to define a parametric class of security properties as follows.

Definition 3.4 (s_BNDC) *Let* $E \in \mathcal{E}$. $E \in s\_BNDC$ *if and only if* $E \approx^s_{\setminus H} E \setminus H$.

Fact. The relation $\approx^*_{\setminus H}$ coincides with $\approx_{\setminus H}$ of Definition 2.7, hence $*\_BNDC$ is exactly P_BNDC.

In order to generalize also the unwinding condition we first introduce the following generalized notion of weak bisimulation.

Definition 3.5 (s-Weak Bisimulation) *A binary relation* $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ *over agents is a* s-weak bisimulation *if* $(E, F) \in \mathcal{R}$ *implies, for all* $a \in Act$,

- *if* $E \xrightarrow{a} E'$, *then there exists* F' *such that* $F \stackrel{a}{\Longrightarrow^s} F'$ *and* $(E', F') \in \mathcal{R}$,
- *if* $F \xrightarrow{a} F'$, *then there exists* E' *such that* $E \stackrel{a}{\Longrightarrow^s} E'$ *and* $(E', F') \in \mathcal{R}$.

Two agents $E, F \in \mathcal{E}$ *are* s-weakly bisimilar, *denoted by* $E \approx^s F$, *if there exists a* s*-weak bisimulation* $\mathcal{R}$ *containing the pair* (E, F).

Fact. The relations $\approx^*$ and $\approx^0$ coincide with the weak bisimulation relation $\approx$, while the relation $\approx^+$ coincides with the *progressing bisimulation* relation $\approx^p$ defined in [38].

Definition 3.6 (s-Unwinding Condition) *Let* $E \in \mathcal{E}$ *be a process.* E satisfies the s-unwinding condition *if for all* E' *reachable from* E, *if* $E' \xrightarrow{h} E''$, *then* $E'(\xrightarrow{\tau})^s E'''$ *and* $E'' \setminus H \approx^s E''' \setminus H$.

Also in this case, for $s = *$ we get back the unwinding condition of Theorem 2.9 which characterizes P_BNDC processes.

The following relationships between $\approx$, $\approx^s$, and $\approx^s_{\setminus H}$ hold.

Proposition 3.7 *Let* $E, F \in \mathcal{E}$.

(1) If $E \approx^s F$ *then* $E \approx F$;
(2) If $E \approx^s_{\setminus H} F$ *then* $E \setminus H \approx^s F \setminus H$;

(3) $E \setminus H \approx^s F \setminus H$ *is equivalent to* $E \setminus H \approx^s_{\setminus H} F \setminus H$*;*
(4) If $s \in \{*, 0\}$ *then* $E \approx F$ *is equivalent to* $E \approx^s F$*.*

Proof. Immediate by Definitions 3.5, 3.1 and 3.3. □

The next proposition shows that the following relationships between *s_BNDC* processes hold.

Proposition 3.8 *0_BNDC* $\subseteq$ **_BNDC and +_BNDC* $\subseteq$ **_BNDC.*

Proof. Immediate by Definitions 3.1, 3.3 and 3.4. □

A relevant property of *P_BNDC* processes is persistency. We show that it holds also for *s_BDNC* processes.

Proposition 3.9 *Let* $E \in \mathcal{E}$*. If* $E \in$ *s_BNDC then for all* E' *reachable from* E, $E' \in$ *s_BNDC.*

Proof. Let $E \in$ *s_BNDC*, i.e., $E \approx^s_{\setminus H} E \setminus H$, and E' be a process reachable from E. First, we prove that there exists $E'' \setminus H$ reachable from $E \setminus H$ such that $E' \approx^s_{\setminus H} E'' \setminus H$. This part of the proof follows by induction on the length l of the path which leads from E to E'.

- Base $l = 0$. We can choose E'' equal to E; then $E \equiv E' \equiv E''$ and we know that $E \approx^s_{\setminus H} E \setminus H$.
- Inductive step $l > 0$. Let F be reachable from E with a path of length $l - 1$ and $F \xrightarrow{a} E'$. By inductive hypothesis, there exists F' such that $F' \setminus H$ is reachable from $E \setminus H$ and $F \approx^s_{\setminus H} F' \setminus H$. From the fact that $F \approx^s_{\setminus H} F' \setminus H$, there exists $E'' \setminus H$ such that $F' \setminus H \hookrightarrow^a_s E'' \setminus H$ and $E' \approx^s_{\setminus H} E'' \setminus H$. Since $E'' \setminus H$ is reachable from $E \setminus H$ we have the thesis.

By Proposition 3.7 (2), we have that $E' \setminus H \approx^s E'' \setminus H$ and, by Proposition 3.7 (3), $E' \setminus H \approx^s_{\setminus H} E'' \setminus H$. Thus, $E' \approx^s_{\setminus H} E'' \setminus H \approx^s_{\setminus H} E' \setminus H$, i.e., by transitivity of $\approx^s_{\setminus H}$, $E' \approx^s_{\setminus H} E' \setminus H$. □

Finally, the following theorem shows the correspondence between *s_BDNC* and *s*-unwinding condition, thus generalizing the one obtained by Theorems 2.8 and 2.9 for *P_BDNC*. This result is used in the next sections to characterize different security properties.

Theorem 3.10 *Let* $E \in \mathcal{E}$ *be a process.* $E \in$ *s_BNDC if and only if* E *satisfies the s-unwinding condition.*

Proof. ($\Rightarrow$) Let $E \in$ *s_BNDC*. By Definition 3.4 and persistence of *s_BNDC*, for all E' reachable from E, $E' \approx^s_{\setminus H} E' \setminus H$. Let E' be a process reachable from E. Suppose that $E' \xrightarrow{h} E''$. By the fact that $E' \approx^s_{\setminus H} E' \setminus H$, it follows

that $E' \setminus H \stackrel{h}{\hookrightarrow}^s E''' \setminus H$ and $E'' \approx^s_{\setminus H} E''' \setminus H$. Since, by persistence of s_BNDC, $E'' \approx^s_{\setminus H} E'' \setminus H$, we have by transitivity that $E'' \setminus H \approx^s_{\setminus H} E''' \setminus H$, i.e., by Proposition 3.7 (2), $E'' \setminus H \approx^s E''' \setminus H$. Since $E' \setminus H$ does not perform any high level action, by Definition 3.1, $E' \setminus H \ (\stackrel{\tau}{\longrightarrow})^s E''' \setminus H$ and thus $E'(\stackrel{\tau}{\longrightarrow})^s E'''$, hence the thesis.

($\Leftarrow$) Let E be a process satisfying the s-unwinding condition, i.e.,

(1) $\forall$ E' reachable from E, if $E' \stackrel{h}{\rightarrow} E''$ then $E'(\stackrel{\tau}{\longrightarrow})^s E'''$ and $E'' \setminus H \approx^s E''' \setminus H$.

It is sufficient to prove that

$$\mathcal{S} = \{(E \setminus H, F) \mid F \text{ satisfy (1) and } E \setminus H \approx^s F \setminus H\}$$

is a s-bisimulation up to H. It follows from the following cases. Let $(E \setminus H, F) \in \mathcal{S}$. Then,

- $E \setminus H \stackrel{a}{\rightarrow} E' \setminus H$ with $a \notin H$. From the hypothesis that $E \setminus H \approx^s F \setminus H$, it follows that $F \setminus H \stackrel{a}{\Longrightarrow}^s F' \setminus H$ and $E' \setminus H \approx^s F' \setminus H$. Hence, since $a \notin H$, $F \stackrel{a}{\Longrightarrow}^s F'$, i.e., by Definition 3.1, $F \stackrel{a}{\hookrightarrow}^s F'$. Moreover, since property (1) is persistent, F' satisfy (1) and thus, by definition of $\mathcal{S}$, $(E' \setminus H, F') \in \mathcal{S}$.
- $F \stackrel{a}{\rightarrow} F'$ with $a \notin H$. Hence, $F \setminus H \stackrel{a}{\rightarrow} F' \setminus H$. From the hypothesis that $E \setminus H \approx^s F \setminus H$, it follows that $E \setminus H \stackrel{a}{\Longrightarrow}^s E' \setminus H$, i.e., by Definition 3.1, $E \setminus H \stackrel{a}{\hookrightarrow}^s E' \setminus H$, and $E' \setminus H \approx^s F' \setminus H$. Moreover, since property (1) is persistent, F' satisfy (1) and thus, by definition of $\mathcal{S}$, $(E' \setminus H, F') \in \mathcal{S}$.
- $F \stackrel{a}{\rightarrow} F'$ with $a \in H$. Since F satisfies property (1), there exists F'' such that $F(\stackrel{\tau}{\longrightarrow})^s F''$ and $F' \setminus H \approx^s F'' \setminus H$. We distinguish three cases corresponding to $s = *$, $s = 0$ and $s = +$.
  - Let $s = *$. From the hypothesis that $E \setminus H \approx^* F \setminus H$, it follows that $E \setminus H \stackrel{\hat{\tau}}{\Longrightarrow} E' \setminus H$, i.e., $E \setminus H \stackrel{a}{\hookrightarrow}^* E' \setminus H$ and $E' \setminus H \approx^* F'' \setminus H$. Hence, by transitivity of $\approx^*$, $E' \setminus H \approx^* F' \setminus H$. Moreover, since property (1) is persistent, F' satisfies (1) and thus, by definition of $\mathcal{S}$, $(E' \setminus H, F') \in \mathcal{S}$.
  - Let $s = 0$. Since $F(\stackrel{\tau}{\longrightarrow})^0 F''$, we have that $F \equiv F''$ and thus $F \setminus H \approx^0 F' \setminus H$. From the hypothesis that $E \setminus H \approx^0 F \setminus H$ and transitivity of $\approx^0$, it holds that $E \setminus H \approx^0 F' \setminus H$. By definition of $\stackrel{a}{\hookrightarrow}^s$, $E \setminus H \stackrel{h}{\hookrightarrow}^0 E \setminus H$. Moreover, since property (1) is persistent, F' satisfies (1) and thus, by definition of $\mathcal{S}$, $(E \setminus H, F') \in \mathcal{S}$.
  - Let $s = +$. From the hypothesis that $E \setminus H \approx^+ F \setminus H$, it follows that $E \setminus H \stackrel{\tau}{\Longrightarrow} E' \setminus H$, i.e., $E \setminus H \stackrel{h}{\hookrightarrow}^+ E' \setminus H$ and $E' \setminus H \approx^+ F'' \setminus H$. Hence, by transitivity of $\approx^+$, $E' \setminus H \approx^+ F' \setminus H$. Moreover, since property (1) is persistent, F' satisfies (1) and thus, by definition of $\mathcal{S}$, $(E' \setminus H, F') \in \mathcal{S}$.

□

4 Strong BNDC

The property *Strong BNDC* (*SBNDC*, for short) has been introduced in [8] as a sufficient condition for verifying *BNDC*. It just requires that before and after every high step, the system appears to be the same, from a low level perspective. It has been proved to be stronger than *SBSNNI* (and thus *P_BNDC*) and it has been defined as follows.

Definition 4.1 (SBNDC [8]) *Let $E \in \mathcal{E}$. $E \in SBNDC$ iff for all E' reachable from E, if $E' \xrightarrow{h} E''$, then $E' \setminus H \approx E'' \setminus H$.*

As a consequence of Proposition 3.7 item (4), we can immediately recognize that *SBNDC* is defined as the class of processes satisfying the 0-unwinding condition, which is obtained by instantiating s to 0 in Definition 3.6. Thus, by Proposition 3.8 we immediately obtain the relation among *SBNDC*, *P_BNDC* and *BNDC*.

Corollary 4.2 *$SBNDC \subseteq P\_BNDC \subseteq BNDC$.*

By exploiting Theorem 3.10 we can provide a quantification-free characterization of *SBNDC* as follows.

Theorem 4.3 (SBNDC - Bisimulation) *Let $E \in \mathcal{E}$ be a process. $E \in SBNDC$ if and only if $E \approx^0_{\setminus H} E \setminus H$.*

Proof. Immediate by Proposition 3.7 item (4) and Theorem 3.10. □

This theorem shows that we can avoid the universal quantification over all the possible reachable states in the definition of *SBNDC* by defining a suitable bisimulation equivalence notion. This property is particularly appealing since it suggests the effective computability of *SBNDC*.

Example 4.4 *Let us consider the process depicted below, modelling the use of a shared resource by a low level* producer *and an high level* consumer*, i.e.,* produce $\in L$ *and* consume $\in H$*.*

$$
\begin{aligned}
R_0 &= produce.R_1 \\
R_i &= produce.R_{i+1} + \overline{consume}.R_{i-1} \qquad for\ i \in [1, n-1] \\
R_n &= produce.R_n + \overline{consume}.R_{n-1}
\end{aligned}
$$

Note that the resource has a maximum capacity of n and the low level produce *action is ignored when such a limit is reached. This non-intuitive behavior is needed in order to avoid a potential flow from high to low level. In particular,*

if the low level producer could observe when the resource is full, this will be exploited to deduce how many high level consume *actions have been performed.*

It is easy to see that this process is SBNDC *by directly applying Definition 4.1. It is sufficient to observe that all the R_j states are equivalent when restricted on high level actions, as they may only perform a produce action moving to another restricted $R_{j'}$.* □

In [19] (see Theorem 4) it is proved that *SBNDC* is compositional with respect to the parallel and restriction operators. It is easy to extend the compositionality result by showing that *SBNDC* is also compositional with respect to low level prefix and relabelling.

Proposition 4.5 *Let $E, F \in \mathcal{E}$. If $E, F \in SBNDC$, then*

- *$a.E \in SBNDC$, for all $a \in L \cup \{\tau\}$;*
- *$(E|F) \in SBNDC$;*
- *$E \setminus v \in SBNDC$, for all $v \subseteq \mathcal{L}$;*
- *$E[f] \in SBNDC$.*

Similarly to *P_BNDC* also *SBNDC* is not compositional with respect to the nondeterministic choice operator. Let us reconsider Example 2.11: the same reasoning holds both for *P_BNDC* and for *SBNDC*.

Example 4.6 *Consider the processes $E_4 = h.\mathbf{0}$ with $h \in H$ and $E_5 = l.\mathbf{0}$ with $l \in L$. It is easy to see that both E_4 and E_5 are* SBNDC *but $E_4 + E_5$ is not* SBNDC. *In fact $E_4 + E_5 \stackrel{h}{\rightarrow} \mathbf{0}$ while $E_4 + E_5 \; (\stackrel{\tau}{\rightarrow})^0 \; E_4 + E_5 = h.\mathbf{0} + l.\mathbf{0}$, but $(h.\mathbf{0} + l.\mathbf{0}) \setminus H \not\approx \mathbf{0}$. The problem lies in the fact that while the high level action in E_4 is safely simulated by a sequence of zero τ in $E_4 \setminus H$, the same high level action in $E_4 + E_5$ is not safely simulated by a sequence of zero τ in $(E_4 + E_5) \setminus H$ due to the presence of the additional component E_5. This problem would not arise if h were be simulated by at least one τ action. This observation will be exploited in the next section to define a fully compositional security property.* □

5 Progressing P_BNDC

It is well-known that security properties are, in general, not preserved under composition [11]. We have seen in the previous sections that *P_BNDC* and *SBNDC* are both non-compositional with respect to the nondeterministic choice operator. However, compositionality results are crucial for making the development of large and complex systems feasible [13,39,40]. In this section we show that by instantiating s to $+$ in Definition 3.4 one obtains a prop-

erty which is *fully compositional* (i.e., it is compositional also with respect to the nondeterministic choice). We call such a class *Progressing P_BNDC* (*PP_BNDC*, for short). We define it in terms of the bisimulation-like relation $\approx^{+}_{\setminus H}$.

Definition 5.1 (PP_BNDC - Bisimulation) *Let $E \in \mathcal{E}$.*

$$E \in PP\_BNDC \quad \textit{iff} \quad E \approx^{+}_{\setminus H} E \setminus H.$$

By exploiting Theorem 3.10, *PP_BNDC* can be characterized also by an unwinding condition.

Theorem 5.2 (PP_BNDC - Unwinding) *Let $E \in \mathcal{E}$. $E \in PP\_BNDC$ iff for all E' reachable from E, if $E' \stackrel{h}{\rightarrow} E''$ then $E' \stackrel{\tau}{\Longrightarrow} E'''$ and $E''\setminus H \approx^{+} E'''\setminus H$.*

Proof. Immediate by Theorem 3.10 and the fact that $(\stackrel{\tau}{\rightarrow})^{+}$ and $\stackrel{\tau}{\Longrightarrow}$ coincide. □

In [1] and [41] a similar unwinding condition is introduced where weak bisimulation is used instead of $\approx^{+}$. The security property so defined is called [5] *CP_BNDC*. Since $E \approx^{+} F$ implies $E \approx F$, $PP\_BNDC \subseteq CP\_BNDC$. By Proposition 3.8 we immediately obtain the relation between *PP_BNDC*, *P_BNDC* and *BNDC*.

Corollary 5.3 *$PP\_BNDC \subseteq CP\_BNDC \subseteq P\_BNDC \subseteq BNDC$.*

Notice that neither *SBNDC* implies *PP_BNDC* nor *PP_BNDC* implies *SBNDC*. For example, process $h.\mathbf{0}$ is *SBNDC* but it is not *PP_BNDC*, as no τ transitions simulate the high level h. On the other hand, the process $E \equiv h.\mathbf{0} + l.\mathbf{0} + \tau.\mathbf{0}$ is *PP_BNDC* but not *SBNDC*. In fact $E \not\approx^{0}_{\setminus H} E \setminus H$ since $E \stackrel{h}{\rightarrow} \mathbf{0}$ but $E \setminus H$ is not weak bisimilar to $\mathbf{0}$. However, there are processes which are both *SBNDC* and *PP_BNDC*, e.g., processes which perform only low level actions. To be more precise, by putting together the two unwinding characterizations, we can say that a process E is both *SBNDC* and *PP_BNDC* if and only if for all E' reachable from E, if $E' \stackrel{h}{\rightarrow} E''$, then $E' \stackrel{\tau}{\Longrightarrow} E'''$ and $E' \setminus H \approx E'' \setminus H \approx^{+} E''' \setminus H$. Consider for instance $E \equiv h.\mathbf{0} + \tau.\mathbf{0}$. The situation is summarized in Fig. 2. Notice that all the inclusions are strict.

Example 5.4 *Consider the process C (channel) described through a value-passing extension of SPA by:*

[5] Note that in [1] the name *CP_BNDC* has been erroneously introduced by using a bisimulation-like relation

$$C = in(x).(\overline{out}(x).C + \tau.C).$$

C may accept a value x at the left-hand port, labelled in. When it holds a value, it either delivers it at the right-hand port, labelled $\overline{out}$, or resets itself performing an internal transition.

If the domain of x is $\{0,1\}$, then the channel C can be translated into SPA in a standard way by following [37] as:

$$C = in_0.(\overline{out}_0.C + \tau.C) + in_1.(\overline{out}_1.C + \tau.C).$$

Let us assume that C is used as communication channel from low to high level. This can be expressed as $in_0, in_1 \in L$ and $\overline{out}_0, \overline{out}_1 \in H$. Since, in correspondence of each high level action $(\overline{out}_0, \overline{out}_1)$ there is a τ transition leading to the same state, by Theorem 5.2 we can conclude that C is PP_BNDC. The τ transitions basically makes the channel a lossy one, as high level outputs may be non-deterministically lost. However, note that non-determinism is used to abstract away implementation details. For example, such τ's could correspond, at implementation time, to time-outs for the high level output actions, i.e., events that empty the channel and allow a new low level input, whenever high outputs are not accepted within a certain amount of time. Analogously, it is possible to see that C is also SBNDC. Note that process $C' = in(x).\overline{out}(x).C'$ with no τ's is neither PP_BNDC *nor SBNDC. Indeed, a high level user may block and unblock C' in order to transmit information to low level user.* □

Exploiting the unwinding characterization we are now ready to prove that *PP_BNDC* is compositional with respect to the nondeterministic choice operator. This is a consequence of the fact that $\approx^+$ coincides with the notion of progressing bisimulation introduced by Montanari and Sassone in [38] which is fully compositional. Notice that an analogous result holds also for *CP_BNDC* as proved in [1].

Proposition 5.5 *Let $E, F \in \mathcal{E}$. If $E, F \in PP\_BNDC$, then*

- *$a.E \in PP\_BNDC$, for all $a \in L \cup \{\tau\}$;*
- *$(E + F) \in PP\_BNDC$;*
- *$(E|F) \in PP\_BNDC$;*
- *$E \setminus v \in PP\_BNDC$, for all $v \subseteq \mathcal{L}$;*
- *$E[f] \in PP\_BNDC$.*

Proof. We show only the case $(E + F)$, since the other cases are similar to the ones of Theorem 4 in [19]. Let $E, F \in PP\_BNDC$. By Theorem 5.2, it is sufficient to show that

(1) $\forall\, G$ reachable from $E+F$, if $G \stackrel{h}{\rightarrow} G'$ then $G \stackrel{\tau}{\Longrightarrow} G''$ and $G'\backslash H \approx^{+} G''\backslash H$.

According to the operational semantics of the nondeterministic choice operator, G can be either a process reachable from E or a process reachable from F or the process $E + F$ itself. In the first two cases, the fact G satisfy (1) follows from the hypothesis that both E and $F \in PP\_BNDC$. Suppose that $E + F \stackrel{h}{\rightarrow} G'$. Then $E \stackrel{h}{\rightarrow} G'$ or $F \stackrel{h}{\rightarrow} G'$. Again, since both E and $F \in PP\_BNDC$, we have that $E \stackrel{\tau}{\Longrightarrow} G''$ (resp. $F \stackrel{\tau}{\Longrightarrow} G''$) and $G' \setminus H \approx^{+} G'' \setminus H$. Hence $E + F \stackrel{\tau}{\Longrightarrow} G''$ and $G' \setminus H \approx^{+} G' \setminus H$ satisfying (1). $\Box$

6 Automatic Verification and its Complexity

In this section we present two methods to determine whether $E \approx^{s}_{\setminus H} E \setminus H$, in the case that E is a finite-state process. Specifically, we tackle the problem of proving $E \approx^{s}_{\setminus H} F$, when E and F are finite-state processes. The first method consists of associating to any process E a modal μ-calculus formula $\phi_E^{\approx^{s}_{\setminus H}}$ such that F satisfies $\phi_E^{\approx^{s}_{\setminus H}}$ if and only if $E \approx^{s}_{\setminus H} F$. This method is obtained by applying the technique presented in [27]. The second method consists of transforming the LTS's of E and F into two LTS's that are strongly bisimilar if and only if $E \approx^{s}_{\setminus H} F$. The first method has the advantage that it directly exploits already existing model checkers for the μ-calculus. Unfortunately, it has an exponential time complexity with respect to the size of the LTS's of E and F. On the other hand, the second method requires the implementation of some ad-hoc transformations of the LTS's, but it has a polynomial time complexity.

6.1 Characteristic Formulae

The modal μ-calculus [29] is a small, yet expressive process logic. We consider modal positive μ-calculus formulae constructed according to the following grammar:

$$\phi ::= \mathbf{true} \mid \mathbf{false} \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \langle a \rangle \phi \mid [a]\phi \mid X \mid \mu X.\phi \mid \nu X.\phi$$

where X ranges over an infinite set of variables and a over a set of actions Act. The *fixpoint operators* μX and νX bind the respective variable X and we adopt the usual notion of closed formula. Notice that we give a syntax without using the negation operator $\neg$ or implication, i.e., we consider only formulae in positive normal form (see [29]).

Modal μ-calculus positive formulae are interpreted over processes, which are modelled by LTS's. Let E be a process and $LTS(E) = (S_E, E, Act, \rightarrow)$. An *environment* is a partial mapping $\rho : Var \rightarrow 2^{S_E}$ which interprets variables by subsets of S_E. Given a formula ϕ and an environment ρ defined on all the free variables of ϕ, the set of processes that satisfy ϕ with respect to ρ, denoted by $M_E(\phi)(\rho)$, is defined [6] in Fig. 3.

Intuitively, **true** (**false**) holds for all (no) states; $\wedge$ and $\vee$ are interpreted by conjunction and disjunction; $\langle a \rangle \phi$ holds in a state $E' \in S_E$ if there is a state E'' reachable from E' with an action a which satisfies ϕ; and $[a]\phi$ holds for E' if all states E'' reachable from E' with an action a satisfy ϕ. The interpretation of a variable X is as prescribed by the environment. The formula $\mu X.\phi$, called *least fixpoint formula*, is interpreted by the smallest subset x of S_E which interprets $\mu X.\phi$ when the environment associates x to X. Similarly, $\nu X.\phi$, called *greatest fixpoint formula*, is interpreted by the largest such set.

The set of processes *satisfying* a closed formula ϕ is $Proc(\phi) = \{F \mid F \in M_F(\phi)\}$.

We consider also *equation systems* of modal μ-calculus formulae in the form

$$Eqn : X_1 = \phi_1, \ \ldots, \ X_n = \phi_n$$

where $X_1, \ldots, X_n$ are mutually distinct variables and $\phi_1, \ldots, \phi_n$ are modal μ-calculus formulae having at most $X_1, \ldots, X_n$ as free variables.

An environment $\rho : \{X_1, \ldots, X_n\} \rightarrow 2^{S_E}$ is a *solution* of an equation system Eqn, if $\rho(X_i) = M_E(\phi_i)(\rho)$, for all $i = 1, \ldots, n$. By ordering environments defined on the same set $\{X_1, \ldots, X_n\}$ of variables with respect to componentwise inclusion:

$$\rho_1 \leq \rho_2 \iff \rho_1(X_i) \subseteq \rho_2(X_i), \ i = 1, \ldots, n,$$

we can determine the greatest of such solutions, which we denote by $M_E(Eqn)$. It interprets an equation system on the processes reachable from a given process E.

We can associate a set of processes to an equation system by saying that a process satisfies an equation system Eqn if it belongs to the greatest solution of the first equation. Thus the set of processes satisfying the system Eqn is $Proc(Eqn) = \{F \mid F \in M_F(Eqn)(X_1)\}$.

In order do derive a characteristic formula for a process E and a given property, we follow the approach described by Müller-Olm in [27] where he shows how to

[6] Given a set $x \subseteq S_E$ and a variable X, we write $\rho[x/X]$ for the environment that maps X to x and any other variable $Y \neq X$ into $\rho(Y)$.

derive μ-calculus formulae characterizing finite state processes up to strong or weak bisimulation directly from the greatest fix-point characterization of the bisimulation relation. As pointed out in [27] it is easy to extend the method to different bisimulation-like relations. The method consists in constructing first an appropriate system of equations of μ-calculus formulae and then a single characteristic formula by applying semantic preserving transformation rules on equation systems.

Before introducing our systems of equations, for any formula ϕ, any action a and $s \in \{*, 0, +\}$ we define the formulas $\langle\langle a\rangle\rangle^s \phi$ and $\langle\langle a\rangle\rangle^s_{\setminus H} \phi$ as follows:

$$\langle\langle a\rangle\rangle^s \phi = \begin{cases} \langle\langle \hat{a}\rangle\rangle \phi & \text{if } s = * \text{ or } s = 0 \\ \langle\langle a\rangle\rangle \phi & \text{if } s = + \end{cases}$$

where $\langle\langle \hat{\tau}\rangle\rangle \phi = \mu X.\phi \vee \langle\tau\rangle X$, with X not in ϕ, $\langle\langle a\rangle\rangle \phi = \langle\langle \hat{\tau}\rangle\rangle \langle a\rangle \langle\langle \hat{\tau}\rangle\rangle \phi$, and if $a \neq \tau$, $\langle\langle \hat{a}\rangle\rangle \phi = \langle\langle a\rangle\rangle \phi$.

$$\langle\langle a\rangle\rangle^s_{\setminus H} \phi = \begin{cases} \langle\langle a\rangle\rangle^s \phi & \text{if } a \notin H \\ \langle\langle a\rangle\rangle \phi \vee \langle\langle \hat{\tau}\rangle\rangle \phi & \text{if } s = * \text{ and } a \in H \\ \langle\langle a\rangle\rangle \phi \vee \phi & \text{if } s = 0 \text{ and } a \in H \\ \langle\langle a\rangle\rangle \phi \vee \langle\langle \tau\rangle\rangle \phi & \text{if } s = + \text{ and } a \in H \end{cases}$$

The operators $\langle\langle a\rangle\rangle^s_{\setminus H}$, $\langle\langle \hat{\tau}\rangle\rangle$ and $\langle\langle a\rangle\rangle$ model $\overset{a}{\hookrightarrow}^s$, $\overset{\hat{\tau}}{\Rightarrow}$ and $\overset{a}{\Rightarrow}$, respectively, since their semantics is given by

$$\begin{aligned} M_E(\langle\langle a\rangle\rangle^s_{\setminus H} \phi)(\rho) &= \{E' \mid \exists E'' \ : \ E' \overset{a}{\hookrightarrow}^s E'' \wedge E'' \in M_E(\phi)(\rho)\}, \\ M_E(\langle\langle \hat{\tau}\rangle\rangle \phi)(\rho) &= \{E' \mid \exists E'' \ : \ E' \overset{\hat{\tau}}{\Longrightarrow} E'' \wedge E'' \in M_E(\phi)(\rho)\}, \\ M_E(\langle\langle a\rangle\rangle \phi)(\rho) &= \{E' \mid \exists E'' \ : \ E' \overset{a}{\Longrightarrow} E'' \wedge E'' \in M_E(\phi)(\rho)\}. \end{aligned}$$

We construct a *characteristic equation system* as follows.

Definition 6.1 *Let E be a finite-state process, $S_E = \{E_1, \ldots, E_n\}$ and $E_1 = E$. For every $E_i \in S_E$ we define*

$$\begin{aligned} \phi^{\overset{s}{\approx}_{\setminus H}}_{E_i} = & \bigwedge_{a \in Act} \bigwedge_{E_i \overset{a}{\rightarrow} E_j} \langle\langle a\rangle\rangle^s_{\setminus H} X_{E_j} \wedge \\ & \bigwedge_{a \in Act} [a] \bigvee_{E_i \overset{a}{\hookrightarrow}^s E_j} X_{E_j}. \end{aligned}$$

The characteristic equation system *of the process E is:*

$$Eqn_E^{\approx^s_{\setminus H}} : X_{E_1} = \phi_{E_1}^{\approx^s_{\setminus H}}, \ldots, X_{E_n} = \phi_{E_n}^{\approx^s_{\setminus H}}.$$

The formulae $\phi_{E_i}^{\approx^s_{\setminus H}}$ have been defined in such a way that the largest solution $M_F(Eqn_E^{\approx^s_{\setminus H}})$ of $Eqn_E^{\approx^s_{\setminus H}}$ on an arbitrary process F associates to the variables X_{E_i} exactly the states F_j of F which are s-weakly bisimilar up to H to E_i. This is formalized by the following theorem whose proof is omitted since it is just an instance of the analogous result in [27].

Theorem 6.2 *Let E, F be a finite-state process, E_i be reachable from E, F_j be reachable from F, and $Eqn_E^{\approx^s_{\setminus H}}$ the characteristic equation system of Definition 6.1. Then $F_j \in M_F(Eqn_E^{\approx^s_{\setminus H}})(X_{E_i})$ iff $E_i \approx^s_{\setminus H} F_j$.*

Example 6.3 *Consider the process E_2 of Example 2.3. The characteristic equation system of E_2 is defined as follows:*

$$\begin{aligned}
X_{E_2} &= \langle\!\langle l \rangle\!\rangle^s_{\setminus H} X_{h.j.\mathbf{0}} \wedge \langle\!\langle l \rangle\!\rangle^s_{\setminus H} X_{\tau.j.\mathbf{0}+\tau.\mathbf{0}} \wedge \\
&\quad [l](X_{h.j.\mathbf{0}} \vee X_{\tau.j.\mathbf{0}+\tau.\mathbf{0}} \vee X_{j.\mathbf{0}} \vee X_{\mathbf{0}}) \wedge [\tau] X_{E_2} \wedge [h] X_{E_2} \\
X_{\tau.j.\mathbf{0}+\tau.\mathbf{0}} &= \langle\!\langle \tau \rangle\!\rangle^s_{\setminus H} X_{j.\mathbf{0}} \wedge \langle\!\langle \tau \rangle\!\rangle^s_{\setminus H} X_{\mathbf{0}} \wedge \\
&\quad [\tau](X_{\tau.j.\mathbf{0}+\tau.\mathbf{0}} \vee X_{\tau.j.\mathbf{0}} \vee X_{j.\mathbf{0}} \vee X_{\tau.\mathbf{0}} \vee X_{\mathbf{0}}) \wedge \\
&\quad [h](X_{\tau.j.\mathbf{0}+\tau.\mathbf{0}} \vee X_{\tau.j.\mathbf{0}} \vee X_{j.\mathbf{0}} \vee X_{\tau.\mathbf{0}} \vee X_{\mathbf{0}}) \\
X_{\tau.j.\mathbf{0}} &= \langle\!\langle \tau \rangle\!\rangle^s_{\setminus H} X_{j.\mathbf{0}} \wedge [\tau](X_{\tau.j.\mathbf{0}} \vee X_{j.\mathbf{0}}) \wedge [h](X_{\tau.j.\mathbf{0}} \vee X_{j.\mathbf{0}}) \\
X_{h.j.\mathbf{0}} &= \langle\!\langle h \rangle\!\rangle^s_{\setminus H} X_{j.\mathbf{0}} \wedge [\tau] X_{h.j.\mathbf{0}} \wedge [h](X_{h.j.\mathbf{0}} \vee X_{j.\mathbf{0}}) \\
X_{j.\mathbf{0}} &= \langle\!\langle j \rangle\!\rangle^s_{\setminus H} X_{\mathbf{0}} \wedge [h] X_{j.\mathbf{0}} \wedge [\tau] X_{j.\mathbf{0}} \wedge [j] X_{\mathbf{0}} \\
X_{\tau.\mathbf{0}} &= \langle\!\langle \tau \rangle\!\rangle^s_{\setminus H} X_{\mathbf{0}} \wedge [\tau](X_{\tau.\mathbf{0}} \vee X_{\mathbf{0}}) \wedge [h](X_{\tau.\mathbf{0}} \vee X_{\mathbf{0}}) \\
X_{\mathbf{0}} &= [h] X_{\mathbf{0}} \wedge [\tau] X_{\mathbf{0}}
\end{aligned}$$

□

Corollary 6.4 $Proc(Eqn_E^{\approx^s_{\setminus H}}) = \{F \mid E \approx^s_{\setminus H} F\}$.

This result holds for all processes F as $Eqn_E^{\approx^s_{\setminus H}}$ does not depend on F.

Characteristic formulae, i.e., *single* positive formulae characterizing processes can be constructed by applying simple semantics-preserving transformation rules on equation systems as described in [27]. We urge the reader to [27] for

a detailed description of such rules which recall Gaussian elimination process. Let ϕ be such that $Proc(Eqn) = Proc(\phi)$ (see [27]), we obtain that:

Theorem 6.5 *For all finite-state processes E and $s \in \{*, 0, +\}$ there is a modal μ-calculus formulae $\phi_E^{\approx^s_{\setminus H}}$ such that $Proc(\phi_E^{\approx^s_{\setminus H}}) = \{F \mid E \approx^s_{\setminus H} F\}$.*

Using this method we can for instance exploit the model checker NCSU Concurrency Workbench ([30]) to check whether $E \approx^s_{\setminus H} F$. Unfortunately, in the μ-calculus formula we obtain for a process E there are both μ and ν operators (see [27]). In the worst case the number of μ and ν alternations in $\phi_E^{\approx^s_{\setminus H}}$ is $2|S_E| + 1$ (when $LTS(E)$ has a unique strongly connected component) and in that case the complexity of model checking $\phi_E^{\approx^s_{\setminus H}}$ on $LTS(F)$ is $O(|S_F|^{(2|S_E|+1)/2})$ (see [42,43]).

This decidability result for *s_BNDC* properties differs from the one proved by Martinelli in [22], even if the underline approach is very similar. Both approached consider only finite-state systems and are based on the construction of characteristic formulae in modal μ-calculus. But we consider sufficient conditions for BNDC while Martinelli considers a necessary condition for BNDC. In fact he restricts also the class of attachers to finite-state processes.

6.2 Strong Bisimulation

We show now how to reduce the problem of testing whether two processes are s-weakly bisimilar up to H to a strong bisimulation problem.

The next property follows from the definition of $\stackrel{a}{\hookrightarrow^s}$.

Proposition 6.6 *Let $s \in \{*, 0, +\}$. A binary relation $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ over agents is a s-*weak bisimulation up to H *if and only if $(E, F) \in \mathcal{R}$ implies, for all $a \in Act$*

1. if $E \stackrel{a}{\hookrightarrow^s} E'$, there is $F' \in \mathcal{E}$ such that $F \stackrel{a}{\hookrightarrow^s} F'$ and $(E', F') \in \mathcal{R}$;

2. if $F \stackrel{a}{\hookrightarrow^s} F'$, there is $E' \in \mathcal{E}$ such that $E \stackrel{a}{\hookrightarrow^s} E'$ and $(E', F') \in \mathcal{R}$.

Proof. ($\Rightarrow$). We prove that if $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ is a s-weak bisimulation up to H, and $(E, F) \in \mathcal{R}$, then *1.* and *2.* hold for all $a \in Act$. We distinguish three cases.
Case 1. $a = \tau$. In this case $E \stackrel{a}{\hookrightarrow^s} E'$ coincides with $E \stackrel{\tau}{\Longrightarrow^s} E'$. The proof follows by induction on the number m of τ actions in $E \stackrel{\tau}{\Longrightarrow^s} E'$. There are two different base cases: $m = 0$ if $s \neq +$, and $m = 1$ if $s = +$. In the first case $E' = E$ and we can choose $F' = F$. The second case is immediate by definition

of s-weak bisimulation up to H. For the inductive step, let $E \stackrel{\tau}{\rightarrow} E'' \stackrel{\tau}{\Longrightarrow}^s E'$. Since, $(E,F) \in \mathcal{R}$, there exists $F'' \in \mathcal{E}$ such that $F \stackrel{\tau}{\Longrightarrow}^s F''$ and $(E'',F'') \in \mathcal{R}$. By the inductive hypothesis, there exists $F' \in \mathcal{E}$ such that $F'' \stackrel{\tau}{\Longrightarrow}^s F'$ and $(E',F') \in \mathcal{R}$. This proves the thesis since $F \stackrel{\tau}{\Longrightarrow}^s F''$ and $F'' \stackrel{\tau}{\Longrightarrow}^s F'$ implies $F \stackrel{\tau}{\Longrightarrow}^s F'$.
Case 2. $a \neq \tau$ and $a \notin H$. In this case $E \stackrel{a}{\hookrightarrow}^s E'$ coincides with $E \stackrel{a}{\Longrightarrow}^s E'$ and there exist E'' such that $E \stackrel{\tau}{\Longrightarrow}^* E'' \stackrel{a}{\rightarrow} E''' \stackrel{\tau}{\Longrightarrow}^* E'$. By Case 1.1 above, there exist $\bar{F}'' \in \mathcal{E}$ such that $F \stackrel{\tau}{\Longrightarrow}^* \bar{F}''$ and $(E'',\bar{F}'') \in \mathcal{R}$. By Definition 3.3 there exists $\bar{F}''' \in \mathcal{E}$ such that $\bar{F}'' \stackrel{a}{\hookrightarrow}^s \bar{F}'''$, i.e. $\bar{F}'' \stackrel{a}{\Longrightarrow}^s F'''$, and $(E''',\bar{F}''') \in \mathcal{R}$. Again by Case 1.1 above, there exists $F' \in \mathcal{E}$ such that $\bar{F}''' \stackrel{\tau}{\Longrightarrow}^* F'$ and $(E',F') \in \mathcal{R}$. This proves the thesis since $F \stackrel{\tau}{\Longrightarrow}^* \bar{F}'' \stackrel{a}{\Longrightarrow}^s \bar{F}''' \stackrel{\tau}{\Longrightarrow}^* F'$ implies $F \stackrel{a}{\Longrightarrow}^s F'$.
Case 3. $a \in H$. In this case $E \stackrel{a}{\hookrightarrow}^s E'$ coincides either with $E \stackrel{a}{\Longrightarrow}^s E'$ or with $E(\stackrel{\tau}{\rightarrow})^s E'$. If $E \stackrel{a}{\Longrightarrow}^s E'$ we proceed as for Case 2 above. If $E(\stackrel{\tau}{\rightarrow})^s E'$ and $s = *$ or $s = +$ then $(\stackrel{\tau}{\rightarrow})^s$ coincides with $\stackrel{\tau}{\Longrightarrow}^s$ and we proceed as for Case 1 above. Finally, if $E \stackrel{\tau}{\Longrightarrow}^0 E'$ then $E' = E$ and we can choose $F' = F$.

($\Leftarrow$). It is sufficient to observe that, by Definition 3.1, $E \stackrel{a}{\rightarrow} E'$ implies $E \stackrel{a}{\hookrightarrow}^s E'$ for each $E, E' \in \mathcal{E}$ and $a \in Act$. $\Box$

A direct consequence of this theorem is that two systems are s-weakly bisimilar up to H if and only if they are strongly bisimilar when in place of the transition relations $\stackrel{a}{\rightarrow}$ we consider the transition relations $\stackrel{a}{\hookrightarrow}^s$.

We can exploit this fact to determine whether $E \approx^s_{\setminus H} E \setminus H$ by: (i) translating the two labelled transition systems $LTS(E)$ and $LTS(E \setminus H)$, into $LTS^s_H(E)$ and $LTS^s_H(E \setminus H)$; (ii) computing the largest strong bisimulation $\sim$ between $LTS^s_H(E)$ and $LTS^s_H(E \setminus H)$. More formally $LTS^s_H(E)$ is:

Definition 6.7 (s-Closure up to H) *Let $E \in \mathcal{E}$ be a process such that $LTS(E) = (S_E, E, Act, \rightarrow\ )$. The* s-closure up to H *of E is the rooted labelled transition system $LTS^s_H(E) = (S_E, E, Act, \hookrightarrow^s)$.*

The notion of bisimulation on rooted labelled transition systems has been first introduced in the areas of modal logics (see [44]) and non-well-founded set theories (see [45]). Two rooted labelled transition systems are strongly bisimilar when, starting from the two roots, each step on the first transition system can be simulated on the second one and vice-versa.

Definition 6.8 (Strong Bisimulation on Rooted Labelled Transition Systems) *Let $G_1 = (S_1, n_1, Act, \hookrightarrow_1)$ and $G_2 = (S_2, n_2, Act, \hookrightarrow_2)$ be two rooted labelled transition systems. G_1 and G_2 are* strong bisimilar*, denoted by $G_1 \sim G_2$, if there exists a binary relation $\mathcal{R} \subseteq S_1 \times S_1$ such that $(n_1, n_2) \in \mathcal{R}$*

Let $E \in \mathcal{E}$ with $LTS(E) = (S_E, E, Act, \rightarrow)$. The *s-closure up to H* of E, $LTS^s_H(E) = (S_E, E, Act, \hookrightarrow^s)$, is computed as follows:

(1) calculate $(\stackrel{\tau}{\rightarrow})^+$ (transitive closure of $\stackrel{\tau}{\rightarrow}$) and $(\stackrel{\tau}{\rightarrow})^*$ (transitive and reflexive closure of $\stackrel{\tau}{\rightarrow}$);

(2) calculate part of $\stackrel{a}{\hookrightarrow^s}$ as:
  (1) $(\stackrel{\tau}{\rightarrow})^* \circ \stackrel{a}{\rightarrow} \circ (\stackrel{\tau}{\rightarrow})^*$, if $a \neq \tau$;
  (2) $(\stackrel{\tau}{\rightarrow})^*$, if $a = \tau$ and $s \neq +$;
  (3) $(\stackrel{\tau}{\rightarrow})^+$, if $a = \tau$ and $s = +$;

(3) calculate $(\stackrel{\tau}{\rightarrow})^s$ as
  (1) $(\stackrel{\tau}{\rightarrow})^*$, if $s = *$;
  (2) $(\stackrel{\tau}{\rightarrow})^+$, if $s = +$;
  (3) $E' \stackrel{\tau}{\rightarrow}^0 E'$, for each $E' \in S_E$, if $s = 0$;

(4) for each $a \in H$ and $E', E'' \in S_E$ add $E' \stackrel{a}{\hookrightarrow^s} E''$, every time $E'(\stackrel{\tau}{\rightarrow})^s E''$.

and $(n'_1, n'_2) \in \mathcal{R}$ *implies, for all* $a \in Act$

- *if* $n'_1 \stackrel{a}{\hookrightarrow}_1 n''_1$, *there is* $n''_2 \in S_2$ *such that* $n'_2 \stackrel{a}{\hookrightarrow}_2 n''_2$ *and* $(n''_1, n''_2) \in \mathcal{R}$;
- *if* $n'_2 \stackrel{a}{\hookrightarrow}_2 n''_2$, *there is* $n''_1 \in S_1$ *such that* $n'_1 \stackrel{a}{\hookrightarrow}_1 n''_1$ *and* $(n''_1, n''_2) \in \mathcal{R}$.

The next result is an immediate consequence of Proposition 6.6.

Corollary 6.9 *Let* $E, F \in \mathcal{E}$. *Then,* $E \approx^s_{\setminus H} F$ *iff* $LTS^s_H(E) \sim LTS^s_H(F)$.

Now, our first problem is to compute $LTS^s_H(E)$ from $LTS(E)$, using Definition 6.7. This can be immediately obtained with the following algorithm:

Correctness of algorithm above is trivially obtained by observing that: if $a \notin H$ and $a \neq \tau$, then $\stackrel{a}{\hookrightarrow^s}$ coincides with $\stackrel{\hat{a}}{\Longrightarrow}$, step 2(1); if $a \in H$, then $\stackrel{a}{\hookrightarrow^s}$ is the union of $\stackrel{\hat{a}}{\Longrightarrow}$ (which coincides with $\stackrel{a}{\Longrightarrow}$ and with $\stackrel{a}{\Longrightarrow^s}$), step 2(1), and of $(\stackrel{\tau}{\rightarrow})^s$, step 4; if $a = \tau$ and $s \neq +$, then $\stackrel{a}{\hookrightarrow^s}$ is $(\stackrel{\tau}{\rightarrow})^*$, step 2(2); if $a = \tau$ and $s = +$, then $\stackrel{a}{\hookrightarrow^s}$ is $(\stackrel{\tau}{\rightarrow})^+$, step 2(3). As far as time and space complexities are concerned, we notice that they depend on the algorithms used for computing the reflexive and transitive closure and the composition of relations. We start by fixing some notations. Let $n = |S_E|$ be the number of states in $LTS(E)$, for each $a \in Act$, let m_a be the number of $\stackrel{a}{\rightarrow}$ transitions in $LTS(E)$, and $m = \sum_{a \in Act} m_a$. Similarly, let $\hat{m}_a$ be the number of $\stackrel{a}{\hookrightarrow^s}$ transitions in $LTS^s_H(E)$, and $\hat{m} = \sum_{a \in Act} \hat{m}_a$.

The next lemma shows that $LTS^s_H(E)$ can be computed in polynomial time with respect to the number of nodes and edges in $LTS(E)$.

Lemma 6.10 *Let* $s \in \{*, 0, +\}$. *Algorithm 6.2 can be executed in time* $O(n\hat{m}_\tau + n^w)$ *and space* $O(n^2)$, *where* w *denotes the exponent in the running time of the matrix multiplication algorithm used. If* $\hat{m} \leq n$, *then it is*

possible to work in time $O(n\hat{m})$ and space $O(n).$

Proof. First of all we have to determine the transitive closure of $\xrightarrow{\tau}$ (step 1). The algorithm proposed in [46] computes the transitive closure of a graph represented with adjacency-lists in time $O(m_\tau + ne)$, where e is the number of edges in the transitive closure of the graph of the strongly connected components. Since $m_\tau, e \leq \hat{m}_\tau$, an upper bound to the cost of the computation of $(\xrightarrow{\tau})^+$ and $(\xrightarrow{\tau})^*$ is $O(n\hat{m}_\tau)$.

Let us consider the computation of the composition $(\xrightarrow{\tau})^* \circ \xrightarrow{a} \circ (\xrightarrow{\tau})^*$ for $a \neq \tau$ (step 2(1)). Given two transition relations $\rightarrow_1$ and $\rightarrow_2$ on a set of n nodes, the problem of determining the composition $\rightarrow_1 \circ \rightarrow_2$ is known to be equivalent to the $n \times n$ Boolean matrix multiplication problem (see [47]). In particular, if A_i is the adjacency-matrix defined by $\rightarrow_i$, for $i = 1, 2$, then the adjacency-matrix of $\rightarrow_1 \circ \rightarrow_2$ is the matrix $A_1 \cdot A_2$. Hence, in our case, we have to: (i) determine the adjacency-matrixes $A_{\tau *}$ and A_a associated to $(\xrightarrow{\tau})^*$ and $\xrightarrow{a}$ respectively; (ii) compute the product $(A_{\tau *} \cdot A_a) \cdot A_{\tau *}$; (iii) rebuild the adjacency-list representation (in the computation of the strong bisimulation it is important to use the adjacency-list representation). Starting from the adjacency-list representations of $(\xrightarrow{\tau})^*$ and $\xrightarrow{a}$ in time $O(n^2)$ we obtain their adjacency-matrix representations $A_{\tau *}$ and A_a. The matrix product $(A_{\tau *} \cdot A_a) \cdot A_{\tau *}$ can be determined in time $O(n^{2.376})$ using twice the algorithm in [48]. Then, again in time $O(n^2)$, we rebuild the adjacency-list representation. So, the global cost of the computation of $(\xrightarrow{\tau})^* \circ \xrightarrow{a} \circ (\xrightarrow{\tau})^*$ is $O(n^{2.376})$. We have to perform this step once for each $a \in \mathcal{L}$, assuming that $|\mathcal{L}|$ is a constant with respect to n. Notice that we could work using only 2 matrix multiplications, instead of $2|\mathcal{L}|$ matrix multiplications, but in this case we would have to use matrixes in which each element is an array of length $\mathcal{L}$ of bits, hence also in this way it is not possible to drop the assumption that $|\mathcal{L}|$ is a constant with respect to n.

The complexity of the computation of $\overset{a}{\hookrightarrow}^s$ for $a = \tau$ (steps 2(2) and 2(3)) has already been considered above (see step 1).

Consider now the computation of $(\xrightarrow{\tau})^s$ (step 3) and the addition of the edges $\overset{a}{\hookrightarrow}^s$ with $a \in H$ of step 4:

- if $s = 0$, then the computation consists in the addition of all the caps with label $a \in H$, hence it costs $O(n)$ (we are assuming that $|\mathcal{L}|$ is a constant);
- if $s \neq 0$, then see the first part of this proof (step 1);

Hence, we have described a procedure which maps E into $LTS^s_H(E)$ in time $O(n\hat{m}_\tau + n^w)$ and space $O(n^2)$, where w is the exponent in the running time of the matrix multiplication algorithm used ($w = 2.376$ using [48]).

In the procedure just described we use the adjacency-matrix representation to compute the relation $\xrightarrow{a} \circ (\xrightarrow{\tau})^*$. If we know that $\hat{m} \leq n$, then using the

adjacency-list representation and a naïve algorithm (two iterations of the naïve algorithm for the transitive closure [47]) we can perform this step in time $O(n\hat{m})$. Thus, when $\hat{m} \leq n$, we determine $LTS^s_H(E)$ in time $O(n\hat{m})$ and space $O(n + \hat{m}) = O(n)$. $\Box$

From the above lemma, since $LTS(E \setminus H)$ and $LTS^s_H(E \setminus H)$ have at most the same size of $LTS(E)$ and $LTS^s_H(E)$, respectively, we obtain the following complexity result.

Theorem 6.11 *Let $s \in \{0, *, +\}$. The test $E \approx^s_{\setminus H} E \setminus H$ can be performed in time $O(n\hat{m}_\tau + n^w + \hat{m} \log n)$ and space $O(n^2)$, where w denotes the exponent in the running time of the matrix multiplication algorithm used.*[7] *If $\hat{m} \leq n$, then it is possible to work in time $O(n\hat{m})$ and space $O(n)$.*

Notice that in the complexity result $\hat{m} \log n$ comes from the fact that we use the algorithm by Paige and Tarjan ([31]) to compute the maximum bisimulation.

Example 6.12 *Consider again process $E_2 = l.h.j.\mathbf{0} + l.(\tau.j.\mathbf{0} + \tau.\mathbf{0})$ of Example 2.3. In Fig. 4 we show $LTS(E_2)$ and $LTS(E_2 \setminus H)$. By performing the closure up to H (Algorithm 6.2) we obtain the transformed labelled transition systems $LTS^*_H(E_2)$ and $LTS^*_H(E_2 \setminus H)$ reported in Fig. 5. In particular, the first step just adds the τ-loops in every state; the second one, adds two transitions labelled with l corresponding to $l.\tau$ and one transition labelled with j corresponding to $\tau.j$; finally, step 4 adds a h-labelled transition every time there is a τ transition. The two transformed transition systems are not strongly bisimilar: the leftmost node after l in $LTS^*_H(E_2)$ is not bisimilar to any node in $LTS^*_H(E_2 \setminus H)$, since in $LTS^*_H(E_2 \setminus H)$ all the nodes are either "sink-nodes" (which only executes τ and h loops) or they have at least one outgoing edge with label j or l. Indeed, that node in $LTS^*_H(E_2)$ may execute only h and τ actions and could thus be simulated only by sink-nodes in $LTS^*_H(E_2 \setminus H)$. However, differently from sink-nodes, after one h, it is also able to execute a j. This proves that $LTS^*_H(E_2) \not\sim LTS^*_H((E_2 \setminus H))$, thus, by Corollary 6.9, $E_2 \notin P\_BNDC$.* $\Box$

7 Related Works and Conclusions

In this paper we study three persistent information flow security properties based on the bisimulation semantics model. For these properties we provide two characterizations: one in terms of a bisimulation-like equivalence relation

[7] In the algorithm in [48], which is at the moment the fastest in literature, we have that $w = 2.376$.

and another one in terms of unwinding conditions.

The first characterization allows us to perform the verification of the properties for finite state processes in polynomial time with respect to the number of states of the system, also improving on the polynomial time complexity required by the Compositional Security Checker (CoSeC) presented in [20].

The second characterization is based on unwinding conditions. This kind of conditions for possibilistic security properties have been already explored in many works like, e.g., [49–51,25]. However, such unwinding conditions, have been all proposed for traces-based models and represent, in most of the cases, only sufficient conditions for their respective security properties. Our work contributes significantly in this research field, by proposing new unwinding conditions for bisimulation-based security properties, which are both necessary and sufficient.

Moreover, unwinding gives new interesting perspectives on the characterized properties, and is also useful for verification. In [24] we show how unwinding conditions can be exploited for defining a proof system which provides a very efficient technique for the verification and the development of *P_BNDC* secure processes. Indeed, the proof system allows us to verify whether a process is secure just by inspecting its syntax, and thus avoiding the state-explosion problem. In particular, it allows us to deal with recursive processes which may perform unbounded sequences of actions, possibly reaching an infinite number of states. Moreover, the system allows us to build processes which are *P_BNDC* by construction in an incremental way. Such a proof system could be easily adapted to deal with the *PP_BNDC* and *SBNDC* properties studied in this paper. In [52], Mantel shows how one can easily define refinement operators which preserve security, starting from unwinding conditions. In [1], we give some preliminary results about refinement operators which preserves our persistent security properties. This is the topic of our current research.

Finally, in this paper we also deal with compositionality issues. The development of large and complex systems strongly depends on the ability of dividing the task of the system into subtasks that are solved by system components. Thus, it is essential to know how properties of the components behave under composition. We show that *P_BNDC* and *SBNDC* are compositional with respect to all the operators of SPA language, except the non-deterministic choice. Moreover, we prove that the new property named *PP_BNDC* is fully compositional. Compositionality of possibilistic security properties has been widely studied in the literature. There are several information flow properties, based on the traces model, which have been proved to be fully compositional like, e.g., restrictiveness [11], forward correctability [53] or separability [13]. In [13,39] it has been studied how to restrict composition in order to preserve certain security properties which are not preserved by (more general) com-

position. In [54], it has been studied how restricting the class of the running environments makes security properties compositional.

References

[1] A. Bossi, R. Focardi, C. Piazza, S. Rossi, Bisimulation and Unwinding for Verifying Possibilistic Security Properties, in: L. D. Zuck, P. C. Attie, A. Cortesi, S. Mukhopadhyay (Eds.), Proc. of Int. Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'03), Vol. 2575 of LNCS, Springer-Verlag, 2003, pp. 223–237.

[2] D. E. Bell, L. J. L. Padula, "Secure Computer Systems: Unified Exposition and Multics Interpretation", ESD-TR-75-306, MITRE MTR-2997.

[3] C. R. Tsai, V. D. Gligor, C. S. Chandersekaran, "On the Identification of Covert Storage Channels in Secure Systems", IEEE Transactions on Software Engineering (1990) 569–580.

[4] J. K. Millen, "Finite-State Noiseless Covert Channels", in: Proceedings of the Computer Security Foundations Workshop II, the MITRE Corporation, IEEE Computer Society Press, 1989, pp. 81–86.

[5] J. A. Goguen, J. Meseguer, Security Policies and Security Models, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'82), IEEE Computer Society Press, 1982, pp. 11–20.

[6] J. A. Goguen, J. Meseguer, Inference Control and Unwinding, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'84), IEEE Computer Society Press, 1984, pp. 75–86.

[7] D. Sutherland, A Model of Information, in: Proc. of the 9th National Computer Security Conference, 1986, pp. 175–183.

[8] R. Focardi, R. Gorrieri, A Classification of Security Properties for Process Algebras, Journal of Computer Security 3 (1) (1994/1995) 5–33.

[9] S. N. Foley, A Universal Theory of Information Flow, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'87), IEEE Computer Society Press, 1987, pp. 116–122.

[10] H. Mantel, Possibilistic Definitions of Security - An Assebly Kit -, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'00), IEEE Computer Society Press, 2000, pp. 185–199.

[11] D. McCullough, Specifications for Multi-Level Security and a Hook-Up Property, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'87), IEEE Computer Society Press, 1987, pp. 161–166.

[12] J. McLean, Security Models and Information Flow, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'90), IEEE Computer Society Press, 1990, pp. 180–187.

[13] J. McLean, A General Theory of Composition for Trace Sets Closed under Selective Interleaving Functions, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'94), IEEE Computer Society Press, 1994, pp. 79–93.

[14] J. McLean, Security Models, Encyclopedia of Software Engineering.

[15] C. O'Halloran, A Calculus of Information Flow, in: Proc. of the European Symposium on Research in Security and Privacy, AFCET, 1990, pp. 180–187.

[16] S. Schneider, May Testing, Non-Interference, and Compositionality, Electronic Notes in Theoretical Computer Science 40.

[17] J. T. Wittbold, D. M. Johnson, "Information Flow in Nondeterministic Systems", in: Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press, 1990, pp. 144–161.

[18] A. Zakinthinos, E. S. Lee, A General Theory of Security Properties, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'97), IEEE Computer Society Press, 1997, pp. 74–102.

[19] R. Focardi, R. Gorrieri, Classification of Security Properties (Part I: Information Flow), in: R. Focardi, R. Gorrieri (Eds.), Foundations of Security Analysis and Design, Vol. 2171 of LNCS, Springer-Verlag, 2001, pp. 331–396.

[20] R. Focardi, R. Gorrieri, The Compositional Security Checker: A Tool for the Verification of Information Flow Security Properties, IEEE Transactions on Software Engineering 23 (9) (1997) 550–571.

[21] A. Durante, R. Focardi, R. Gorrieri, A compiler for analysing cryptographic protocols using non-interference, ACM Transactions on Software Engineering and Methodology (TOSEM) 9 (4) (2000) 488–528.

[22] F. Martinelli, Partial Model Checking and Theorem Proving for Ensuring Security Properties, in: Proc. of the IEEE Computer Security Foundations Workshop (CSFW'98), IEEE Computer Society Press, 1998, pp. 44–52.

[23] R. Focardi, S. Rossi, Information Flow Security in Dynamic Contexts, in: Proc. of the 15th IEEE Computer Security Foundations Workshop (CSFW'02), IEEE Computer Society Press, 2002, pp. 307–319.

[24] A. Bossi, R. Focardi, C. Piazza, S. Rossi, A Proof System for Information Flow Security, in: M. Leuschel (Ed.), Proc. of Int. Workshop on Logic Based Program Development and Transformation (LOPSTR'02), Vol. 2264 of LNCS, Springer-Verlag, 2002, pp. 199–218.

[25] H. Mantel, Unwinding Possibilistic Security Properties, in: Proc. of the European Symposium on Research in Computer Security, Vol. 2895 of LNCS, Springer-Verlag, 2000, pp. 238–254.

[26] R. Forster, Non-Interference Properties for Nondeterministic Processes, Ph.D. thesis, Oxford University Computing Laboratory (1999).

[27] M. Müller-Olm, Derivation of Characteristic Formulae, Electronic Notes in Theoretical Computer Science 18.

[28] B. Steffen, A. Ingòlfsdòttir, Characteristic Formulae for Processes with Divergence, Information and Computation 110 (1) (1994) 149–163.

[29] D. Kozen, Results on the Propositional μ-calculus, Theoretical Computer Science 27 (1983) 333–354.

[30] R. Cleaveland, S. Sims, The NCSU Concurrency Workbench, in: R. Alur, T. Henzinger (Eds.), Proc. of Int. Conference on Computer Aided Verification (CAV'96), Vol. 1102 of LNCS, Springer-Verlag, 1996, pp. 394–397.

[31] R. Paige, R. E. Tarjan, Three Partition Refinement Algorithms, SIAM Journal on Computing 16 (6) (1987) 973–989.

[32] A. Bouali, R. de Simone, Symbolic Bisimulation Minimization, in: G. von Bochmann, D. K. Probst (Eds.), Proc. of Int. Conference on Computer Aided Verification (CAV'92), Vol. 663 of LNCS, Springer-Verlag, 1992, pp. 96–108.

[33] D. Lee, M. Yannakakis, Online Minimization of Transition Systems, in: Proc. of 24th ACM Symposium on Theory of Computing (STOC'92), ACM Press, 1992, pp. 264–274.

[34] A. Dovier, C. Piazza, A. Policriti, A Fast Bisimulation Algorithm, in: G. Berry, H. Comon, A. Finkel (Eds.), Proc. of Int. Conference on Computer Aided Verification (CAV'01), Vol. 2102 of LNCS, Springer-Verlag, 2001, pp. 79–90.

[35] A. Bouali, XEVE, an ESTEREL verification environment, in: A. J. Hu, M. Y. Vardi (Eds.), Proc. of Int. Conference on Computer Aided Verification (CAV'98), Vol. 1427 of LNCS, Springer-Verlag, 1998, pp. 500–504.

[36] A. W. Roscoe, The Theory and Practice of Concurrency, Series in Computer Science, Prentice Hall, 1998.

[37] R. Milner, Communication and Concurrency, Prentice-Hall, 1989.

[38] U. Montanari, V. Sassone, CCS Dynamic Bisimulation is Progressing, in: Proc. of the 16th International Symposium on Mathematical Foundations of Computer Science (MFCS'91), Vol. 520 of LNCS, Springer-Verlag, 1991, pp. 346–356.

[39] J. McLean, A General Theory of Composition for a Class of "Possibilistic" Security Properties, IEEE Trabsactions on Software Engineering 22 (1) (1996) 53–67.

[40] H. Mantel, On the Composition of Secure Systems, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'02), IEEE Computer Society Press, 2002, pp. 88–101.

[41] A. Bossi, R. Focardi, C. Piazza, S. Rossi, Refinement Operators and Information Flow Security, in: Proc. of the 1st IEEE Int. Conference on Software Engineering and Formal Methods (SEFM'03), IEEE Computer Society Press, 2003, pp. 44–53.

[42] D. Long, A. Browne, E. Clarke, S. Jha, W. Marrero, An improved Algorithm for the Evaluation of Fixpoint expressions, in: D. L. Dill (Ed.), Proc. of Int. Conference on Computer Aided Verification (CAV'94), Vol. 818 of LNCS, Springer-Verlag, 1994, pp. 338–350.

[43] E. M. Clarke, O. Grumberg, D. A. Peled, Model Checking, The MIT Press, 1999.

[44] J. v. Benthem, Modal Correspondence Theory, Ph.D. thesis, Universiteit van Amsterdam, Instituut voor Logica en Grondslagenonderzoek van Exacte Wetenschappen (1976).

[45] P. Aczel, Non-well-founded Sets, Vol. 14 of CSLI Lecture Notes, Stanford University Press, 1988.

[46] A. Goralcikova, V. Koubek, A reduct and closure algorithm for graphs, in: Proc. of Mathematical Foundations of Computer Science (MFCS'79), Vol. 74 of LNCS, Springer-Verlag, 1979, pp. 301–307.

[47] T. H. Cormen, C. E. Leiserson, R. L. Rivest, Introduction to Algorithms, The MIT Press, 1990.

[48] D. Coppersmith, S. Winograd, Matrix multiplication via arithmetic progression, in: Proc. of the 19th Symposium on Theory of Computing, 1987, pp. 1–6.

[49] J. Graham-Cumming, J. W. Sanders, On the Refinement of Non-Interference, in: Proc. of the IEEE Computer Security Foundations Workshop (CSFW'91), IEEE Computer Society Press, 1991, pp. 35–42.

[50] P. Y. A. Ryan, A CSP Formulation of Non-Interference and Unwinding, Cipher (1991) 19–27.

[51] J. K. Millen, Unwinding Forward Correctability, in: Proc. of the IEEE Computer Security Foundations Workshop (CSFW'94), IEEE Computer Society Press, 1994, pp. 2–10.

[52] H. Mantel, Preserving Information Flow Properties under Refinement, in: Proc. of the IEEE Symposium on Security and Privacy (SSP'01), IEEE Computer Society Press, 2001, pp. 78–91.

[53] D. M. Johnson, F. J. Thayer, Security and the Composition of Machines, in: Proc. of of the IEEE Computer Security Foundations Workshop (CSFW'88), IEEE Computer Society Press, 1988, pp. 72–89.

[54] A. Bossi, D. Macedonio, C. Piazza, S. Rossi, Secure Contexts for Confidential Data, in: Proc. of the 16th IEEE Computer Security Foundations Workshop (CSFW'03), IEEE Computer Society Press, 2003, pp. 14–25.

Prefix	$\dfrac{-}{a.E \xrightarrow{a} E}$
Sum	$\dfrac{E_1 \xrightarrow{a} E_1'}{E_1 + E_2 \xrightarrow{a} E_1'} \quad \dfrac{E_2 \xrightarrow{a} E_2'}{E_1 + E_2 \xrightarrow{a} E_2'}$
Parallel	$\dfrac{E_1 \xrightarrow{a} E_1'}{E_1 \mid E_2 \xrightarrow{a} E_1' \mid E_2} \quad \dfrac{E_2 \xrightarrow{a} E_2'}{E_1 \mid E_2 \xrightarrow{a} E_1 \mid E_2'}$ $\dfrac{E_1 \xrightarrow{a} E_1' \quad E_2 \xrightarrow{\bar{a}} E_2'}{E_1 \mid E_2 \xrightarrow{\tau} E_1' \mid E_2'}\ a \in \mathcal{L}$
Restriction	$\dfrac{E \xrightarrow{a} E'}{E \setminus v \xrightarrow{a} E' \setminus v}$ if $a \notin v$
Relabelling	$\dfrac{E \xrightarrow{a} E'}{E[f] \xrightarrow{f(a)} E'[f]}$
Constant	$\dfrac{E \xrightarrow{a} E'}{Z \xrightarrow{a} E'}$ if $Z \stackrel{\text{def}}{=} E$

Fig. 1. The operational rules for SPA

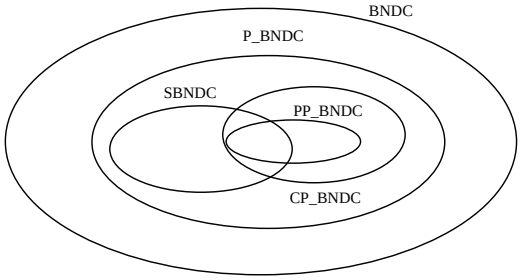

Fig. 2. Security Properties.

$$
\begin{aligned}
M_E(\mathbf{true})(\rho) &= S_E \\
M_E(\mathbf{false})(\rho) &= \emptyset \\
M_E(\phi_1 \wedge \phi_2)(\rho) &= M_E(\phi_1)(\rho) \cap M_E(\phi_2)(\rho) \\
M_E(\phi_1 \vee \phi_2)(\rho) &= M_E(\phi_1)(\rho) \cup M_E(\phi_2)(\rho) \\
M_E(\langle a \rangle \phi)(\rho) &= \{E' \mid \exists E'' :\ E' \xrightarrow{a} E''\ \wedge\ E'' \in M_E(\phi)(\rho)\} \\
M_E([a]\phi)(\rho) &= \{E' \mid \forall E'' :\ E' \xrightarrow{a} E''\ \Rightarrow\ E'' \in M_E(\phi)(\rho)\} \\
M_E(X)(\rho) &= \rho(X) \\
M_E(\mu X.\phi)(\rho) &= \bigcap\{x \subseteq S_E \mid M_E(\phi)(\rho[x/X]) \subseteq x\} \\
M_E(\nu X.\phi)(\rho) &= \bigcup\{x \subseteq S_E \mid M_E(\phi)(\rho[x/X]) \supseteq x\}
\end{aligned}
$$

Fig. 3. Semantics of modal μ-calculus

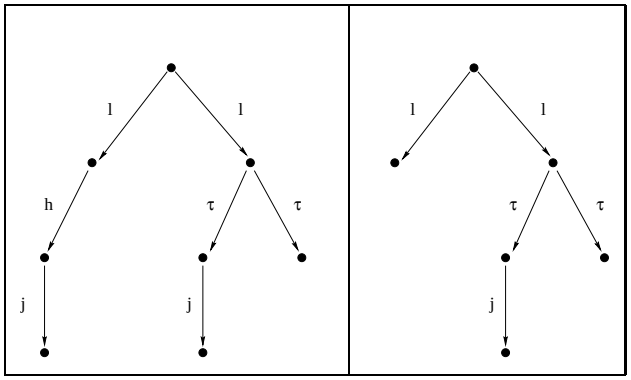

Fig. 4. The labelled transition systems of E_2 and $E_2 \setminus H$.

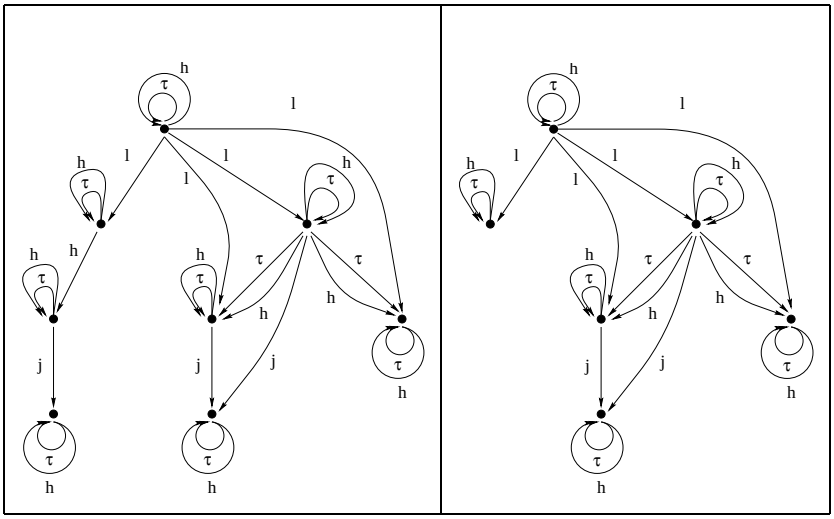

Fig. 5. The labelled transition systems $LTS^*_H(E_2)$ and $LTS^*_H(E_2 \setminus H)$.