

Information Flow Security and Recursive Systems^{*}

Annalisa Bossi, Damiano Macedonio, Carla Piazza, and Sabina Rossi

Dipartimento di Informatica, Università Ca' Foscari di Venezia
{bossi,mace,piazza,srossi}@dsi.unive.it

Abstract. Information flow security in a multilevel system aims at guaranteeing that no high level information is revealed to low level users, even in the presence of any possible malicious process. *Persistent_BNDC* (*P_BNDC*, for short) is an information-flow security property which is suitable to deal with processes in dynamic contexts. In this work we show that *P_BNDC* is compositional with respect to the replication operator. Then, by exploiting the compositionality properties of the class of *P_BNDC* processes, we define a proof system which provides a very efficient technique for the stepwise development and the verification of recursively defined *P_BNDC* processes.

1 Introduction

The design of large and complex systems that satisfy a given property strongly depends on the ability of dividing the task of the system into subtasks that are solved by system components. It is the classical divide-and-conquer approach, at the basis of any systematic development of complex systems. When security is the property of interest, difficulties can be encountered in applying this approach since secure systems might not be composed by secure components only. Nevertheless it is essential to know how properties of the components behave under composition. General theories of compositionality exist for properties like safety and liveness [25, 1] and compositionality results for information-flow based confidentiality properties have also been developed [18, 26, 15].

The problem of protecting confidential data in a multilevel system is one of the relevant issues in computer security. Information flow security assures confidentiality since it guarantees that no high level (confidential) information is revealed to users running at low levels [12, 17, 9, 22], even in the presence of any possible malicious process. To establish that information does not flow from high to low it is sufficient to establish that high behavior has no effect on what low level users can observe, i.e., the low level view of the system is independent of high behavior. This notion of information flow security, known

^{*} This work has been partially supported by the MURST project “Modelli formali per la sicurezza” and the EU Contract IST-2001-32617 “Models and Types for Security in Mobile Distributed Systems” (MyThS).

as Non-Interference, has been introduced in [13], and subsequently developed by many authors in many different settings [9, 10, 21, 23, 14].

In this paper we consider the security property *Persistent_BNDC* (*P_BNDC*, for short), proposed in [11], and further studied in [4]. *P_BNDC* is a security property based on Non-Interference suitable to analyze processes in completely dynamic hostile environments. In [4] we exploit the compositionality properties of *P_BNDC* processes to define a proof system which allows us to *statically prove* that a process is *P_BNDC* by just inspecting its syntax. The proof system consists of two layers, a kernel which deals only with non-recursive processes and a second layer where a rather complex rule involving many expensive checks is devoted to the treatment of recursive processes. The system is correct but not complete, for instance it does not deal with recursive processes involving the parallel operator. The incompleteness and the complexity of the system is due to the lack of a compositionality result for constant definitions, which is the only way recursion is expressed in the considered language, SPA, that is a variant of Milner's CCS [19].

In this paper we consider another form of recursion, namely the replication operator (!) which is used by many authors instead of constant definitions. The two approaches have the same expressive power in π -calculus [20, 24], but as recently proved in [7], replication cannot supplant recursion in CCS. In this paper we show that the class of *P_BNDC* processes is compositional with respect to the replication operator. This allows us to extend the kernel of the proof system in [4] with a new inference rule for the replication, thus allowing us to deal also with recursive processes involving the parallel operator. Moreover, we prove a partial compositionality of *P_BNDC* with respect to constant definitions, i.e., we identify a class of constant definitions which can be safely added to our language and treated by the extended proof system.

The paper is organized as follows. In Section 2 we introduce the language, and recall the definition of *P_BNDC* process and its properties. In Section 3 we prove that *P_BNDC* is compositional with respect to the replication operator, and then present a proof system which, by exploiting the new compositionality result, extends the kernel presented in [4] by adding recursion through replication in a very simple way. Section 4 is devoted to the (re)-introduction of constant definitions. Finally, in Section 5 we draw some conclusions. All the proofs can be found in [5]; we also add an Appendix with the proofs of the principal results.

2 Basic Notions

2.1 The Language

In this section we report the syntax and semantics of the process algebra we consider in this paper. It is a variation of Milner's CCS [19], similar to the SPA [9], where the set of visible actions is partitioned into high level actions and low level ones in order to specify multilevel systems. Differently from [19], we use the *replication* (!) operator instead of the constant definitions. Intuitively, the process

$!E$ (bang E) means $E|E|\dots$, i.e., the parallel composition of as many copy as needed of the process E . In Section 4 we will reintroduce constant definitions.

The syntax of our process algebra 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 $I = \{a, b, \dots\}$ is a set of *input* actions and $O = \{\bar{a}, \bar{b}, \dots\}$ 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 $\overline{L} = L$. A *process* E is a term built using the following productions:

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

where $a \in Act$, $v \subseteq \mathcal{L}$, $f : Act \rightarrow Act$ is such that $f(L) \subseteq L \cup \{\tau\}$, $f(H) \subseteq H \cup \{\tau\}$, $f(\bar{a}) = \overline{f(a)}$ and $f(\tau) = \tau$.

Given a fixed set \mathcal{L} we denote by $\mathcal{E}^!$ the set of all processes, by $\mathcal{E}_H^!$ the set of all high level processes, i.e., those constructed over $H \cup \{\tau\}$, and by $\mathcal{E}_L^!$ the set of all low level processes, i.e., those constructed over $L \cup \{\tau\}$.

The operational semantics of processes is given in terms of a *Labelled Transition System* (LTS). In particular, the operational semantics of our language 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 reported in Figure 1.

In the paper we use the following notations. If $t = a_1 \cdots a_n \in Act^*$ and $E \xrightarrow{a_1} \cdots \xrightarrow{a_n} E'$, then we say that E' is reachable from E and write $E \xrightarrow{t} E'$, or simply $E \rightsquigarrow E'$. We also write $E \xRightarrow{t} E'$ 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, $E \xrightarrow{\hat{a}} E'$ stands for $E \xrightarrow{a} E'$ if $a \in \mathcal{L}$, and for $E(\xrightarrow{\tau})^* E'$ if $a = \tau$ (note that $\xrightarrow{\tau}$ requires at least one τ labelled transition while $\xrightarrow{\hat{\tau}}$ means zero or more τ labelled transitions). Given two processes E, F we write $E \equiv F$ when E and F are syntactically equal.

The concept of *observation equivalence* between two processes 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 $\mathcal{E}^!$. In the Appendix we report the definitions of *weak bisimulation* and *strong bisimulation* [19]. Intuitively, weak bisimulation equates two processes if they mutually simulate their behavior step by step, but it does not care about internal τ actions. So, when P simulates an action of Q , it can also execute some τ actions before or after that action. Strong bisimulation is stronger than weak bisimulation, since it considers the τ actions as any other action. We denote by \approx the largest weak bisimulation and by \sim is the largest strong bisimulation. Both of them are equivalence relations [19]. Moreover, two strongly bisimilar processes are also weakly bisimilar.

Prefix	$a.E \xrightarrow{a} E$		
Sum	$E_1 \xrightarrow{a} E'_1$	$E_2 \xrightarrow{a} E'_2$	
Parallel	$E_1 + E_2 \xrightarrow{a} E'_1$	$E_1 + E_2 \xrightarrow{a} E'_2$	$E_1 \xrightarrow{\ell} E'_1 \quad E_2 \xrightarrow{\bar{\ell}} E'_2$
	$E_1 \xrightarrow{a} E'_1$	$E_2 \xrightarrow{a} E'_2$	$E_1 \xrightarrow{\ell} E'_1 \quad E_2 \xrightarrow{\bar{\ell}} E'_2$
	$E_1 E_2 \xrightarrow{a} E'_1 E_2$	$E_1 E_2 \xrightarrow{a} E_1 E'_2$	$E_1 E_2 \xrightarrow{\tau} E'_1 E'_2$
			$\ell \in \mathcal{L}$
Restriction	$E \xrightarrow{a} E'$		
	if $a \notin v$		
	$E \setminus v \xrightarrow{a} E' \setminus v$		
Relabelling	$E \xrightarrow{a} E'$		
	$E[f] \xrightarrow{f(a)} E'[f]$		
Replication	$E \xrightarrow{a} E'$		
	$E \xrightarrow{\ell} E' \quad E \xrightarrow{\bar{\ell}} E''$		
	$!E \xrightarrow{a} E' E$	$!E \xrightarrow{\tau} E' E'' E$	$\ell \in \mathcal{L}$

Fig. 1. The operational rules

2.2 The P_BNDC Security Property

In this section we recall the *Persistent Bisimulation-based Non Deducibility on Compositions* (P_BNDC , for short) security property (see [11]). We start by introducing an equivalence relation on low actions that is a sort of weak bisimulation which considers only the low actions. Hence, when two processes are weakly bisimilar on low actions they cannot be distinguished by a low level user.

Definition 1 (Weak Bisimulation on Low Actions). *A symmetric binary relation $\mathcal{R} \subseteq \mathcal{E}^! \times \mathcal{E}^!$ over processes is a weak bisimulation on low actions, if $(E, F) \in \mathcal{R}$ implies, for all $a \in L \cup \{\tau\}$,*

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

Two processes $E, F \in \mathcal{E}^!$ are weakly bisimilar on low actions, denoted by $E \approx_l F$, if there exists a weak bisimulation on low actions \mathcal{R} containing the pair (E, F) .

The relation \approx_l is the largest weak bisimulation on low actions and it is an equivalence relation. Moreover, it holds $E \approx_l F$ if and only if $E \setminus H \approx F \setminus H$.

Using weak bisimulation on low actions we recall the notion of *Bisimulation-based Non Deducibility on Compositions* ($BNDC$, for short) [9] which is at the basis of P_BNDC . The $BNDC$ 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. A system E is *BNDC* if for every high process Π a low user cannot distinguish E from $(E|\Pi)$, i.e., if Π cannot interfere [13] with the low level execution of E .

Definition 2 (BNDC). *Let $E \in \mathcal{E}^!$. $E \in \text{BNDC}$ iff $\forall \Pi \in \mathcal{E}_H^!$, $E \approx_l (E|\Pi)$.*

In [11] it is shown that *BNDC* is not strong enough for systems in dynamic environments. To deal with these situations, the property *P_BNDC* is introduced. Intuitively, a system E is *P_BNDC* if it never reaches insecure states.

Definition 3 (P_BNDC). *Let $E \in \mathcal{E}^!$. $E \in \text{P_BNDC}$ iff $E \rightsquigarrow E'$ implies $E' \in \text{BNDC}$.*

In [4] it has been proved that the class of *P_BNDC* processes enjoys the following compositionality properties.

Lemma 1. *The class of P_BNDC processes contains all the processes in $\mathcal{E}_L^! \cup \mathcal{E}_H^!$ and is closed with respect to restriction, renaming, and parallel composition. Moreover, if $E_i, F_j \in \text{P_BNDC}$, $a_i \in L$ and $h_j \in H$, $i \in I$ and $j \in J$, then $\sum_{i \in I} a_i.E_i + \sum_{j \in J} (h_j.F_j + \tau.F_j) \in \text{P_BNDC}$.*

3 *P_BNDC* and Replications

In this section we first extend the compositionality result of Lemma 1 by proving that *P_BNDC* is closed also with respect to the replication operator. Then we present a proof system for *P_BNDC* processes. The proofs of the results presented in Section 3.1 are reported in the Appendix.

3.1 Compositionality of *P_BNDC* wrt !

We start by observing that the processes reachable from $!E$ have the form of a parallel composition of a finite number of processes reachable from E and $!E$.

Lemma 2. *Let $E \in \mathcal{E}^!$ be a process. If $!E \rightsquigarrow E'$, then there exist $n \geq 0$ and E_1, \dots, E_n such that $E \rightsquigarrow E_i$, for $i = 1, \dots, n$ and $E' \equiv E_1|E_2|\dots|E_n|!E$.*

Hence the set $\{E_1, \dots, E_n\}$ of processes reachable from E characterizes the process $E_1|E_2|\dots|E_n|!E$ reachable from $!E$.

There is an interesting connection between the processes reachable from E and the processes reachable from $!E$ when E is *P_BNDC*: if the sets $\{F_1, \dots, F_n\}$ and $\{G_1, \dots, G_n\}$ of processes reachable from E are pairwise weakly bisimilar on low actions, i.e., $F_i \approx_l G_i$, this relation is preserved also on the processes reachable from $!E$ that they characterize.

Lemma 3. *Let E be a P_BNDC process and $\forall i \in \{1, \dots, n\}$ F_i, G_i be reachable from E . If $\forall i \in \{1, \dots, n\}$ $F_i \approx_l G_i$ then $F_1|F_2|\dots|F_n|!E \approx_l G_1|G_2|\dots|G_n|!E$.*

The two previous lemmas, together with the unwinding condition (see Theorem 4 in the Appendix), allow us to prove that *P_BNDC* is compositional with respect to the replication operator.

Theorem 1. *Let $E \in \mathcal{E}^!$ be a process. If $E \in \text{P_BNDC}$, then $!E \in \text{P_BNDC}$.*

$$\begin{array}{c}
\frac{}{E \in \mathcal{HP}\{\{E\}\}} \quad E \text{ is a process} \quad (Proc) \\
\\
\frac{}{E \in \mathcal{HP}\{\emptyset\}} \quad E \in \mathcal{E}_L^! \quad (Low) \qquad \frac{}{E \in \mathcal{HP}\{\emptyset\}} \quad E \in \mathcal{E}_H^! \quad (High) \\
\\
\frac{E \in \mathcal{HP}[A]}{E \setminus v \in \mathcal{HP}[A]} \quad (Rest) \qquad \frac{E \in \mathcal{HP}[A]}{E[f] \in \mathcal{HP}[A]} \quad (Label) \\
\\
\frac{E \in \mathcal{HP}[A] \quad F \in \mathcal{HP}[B]}{E|F \in \mathcal{HP}[A \cup B]} \quad (Par) \\
\\
\frac{E_i \in \mathcal{HP}[A_i] \quad F_j \in \mathcal{HP}[B_j]}{\sum_{i \in I} a_i.E_i + \sum_{j \in J} (h_j.F_j + \tau.F_j) \in \mathcal{HP}[\cup_i A_i \cup \cup_j B_j]} \quad \begin{array}{l} a_i \in L \cup \{\tau\}, h_j \in H \\ (Choice) \end{array} \\
\\
\frac{E \in \mathcal{HP}[A]}{!E \in \mathcal{HP}[A]} \quad (Repl) \qquad \frac{E[G] \in \mathcal{HP}[A] \quad F \in \mathcal{HP}[B]}{E[F/G] \in \mathcal{HP}[(A \setminus \{F\}) \cup B]} \quad (Subst)
\end{array}$$

Fig. 2. The proof system $System^1$

3.2 A Proof System for Processes with Replications

In [4] it has been presented a proof system which allows us to build P_BNDC processes in an incremental way. The proof system is composed by a set of rules whose conclusions are in the form $E \in \mathcal{HP}[A]$, where A is a set of constants. The intended meaning of the judgment is that E is a P_BNDC process provided that all the constants in A are P_BNDC . The set A plays the role of a set of assumptions: if it is empty then E is P_BNDC otherwise we are still working on our construction under open hypothesis. It is immediate to observe that the system described in [4] is correct also using set of processes, instead of set of constants, as assumptions. Hence, in this section the meaning of $E \in \mathcal{HP}[A]$ is that E is a P_BNDC process provided that all the processes in A are P_BNDC . We show how to exploit Lemma 1 and Theorem 1 in order to extend the system to the case of processes with replication. In particular, let us consider the proof system $System^1$ whose rules are shown in Figure 2¹.

¹ We use $E[F/G]$ to denote the process we obtain by replacing all the occurrences of G in E with F , where G denotes a process whose occurrences in E can be syntactically and unambiguously identified.

Theorem 2 (Correctness). *System[!] is correct, i.e., if there exists a proof in System[!] which ends with $E \in \mathcal{HP}[A]$, then E is P_BNDC provided that all the processes in A are P_BNDC .*

Corollary 1. *Let $E \in \mathcal{E}^!$. If there exists a proof of $E \in \mathcal{HP}[\emptyset]$, then E is P_BNDC .*

Example 1. Consider the process CH defined as

$$CH \equiv ((in_0.(\overline{out_0}.\overline{\sigma}.\mathbf{0} + \tau.\overline{\sigma}.\mathbf{0}) + in_1.(\overline{out_1}.\overline{\sigma}.\mathbf{0} + \tau.\overline{\sigma}.\mathbf{0})) | \\ !(\sigma.(in_0.(\overline{out_0}.\overline{\sigma}.\mathbf{0} + \tau.\overline{\sigma}.\mathbf{0}) + in_1.(\overline{out_1}.\overline{\sigma}.\mathbf{0} + \tau.\overline{\sigma}.\mathbf{0})))) \setminus \{\sigma, \overline{\sigma}\}$$

where $in_0, in_1, \sigma, \overline{\sigma} \in L$ and $\overline{out_0}, \overline{out_1} \in H$. This process CH is a channel which may accept a value 0 (or 1) through the low level input in_0 (or in_1). When it holds a value, it may deliver it through a high level output $\overline{out_0}$ (or $\overline{out_1}$). The channel can transmit values infinitely many times. In fact, when the $\overline{\sigma}$ action is reached the process resets itself and recursively repeats the sequence of actions.

This process is a variation of the channel described in [19]. It is easy to see that we can derive the judgement $CH \in \mathcal{HP}[\emptyset]$ in *System[!]*.

4 Adding Constant Definitions

In this section we add some constant definitions to our language. Then, exploiting the compositionality of P_BNDC with respect to the replication operator, we prove a compositionality result for P_BNDC with respect to the constant definitions we consider. We do not add all constant definitions, since in CCS, differently from π -calculus [24], replication is not expressive enough to represent all constant definitions [7].

4.1 Definitions using Replications

In standard CCS [19] complex recursive systems are defined parametrically, as $Z \stackrel{\text{def}}{=} E[Z]$, where Z is a process identifier and $E[Z]$ a process expression which may contain “calls” to Z as well as to other parametric processes.

Example 2. Consider the process Z recursively defined as $Z \stackrel{\text{def}}{=} a.Z + b.\mathbf{0}$. Intuitively this process can perform either an action a and return in its initial state or an action b and terminate. Similarly it is possible to consider two mutually defined processes X and Y where X performs an action a and then calls Y ; while Y performs an action b and calls X . Their definitions are

$$X \stackrel{\text{def}}{=} a.Y \qquad Y \stackrel{\text{def}}{=} b.X$$

This way of defining recursive processes was taken as basic in [9] and in other previous works on P_BNDC (see [4]). In the context of the π -calculus in [20], an encoding is defined which eliminates a finite number of constant definitions

using replication. As already noticed in [24], the same encoding applied to full CCS does not work (see also Remark 1). In what follows we identify a fragment of CCS on which the encoding is correct.

Let $Act = \mathcal{L} \cup \{\tau\}$ be a set of actions, with \mathcal{L} partitioned into the two sets H and L , as described in Section 2.1. Let \mathcal{C} be a finite set of constants. Consider all the processes D which can be obtained using the following productions:

$$D ::= \mathbf{0} \mid a.D \mid D + D \mid D|D \mid Z$$

where $Z \in \mathcal{C}$ is a constant which must be associated to a definition $Z \stackrel{\text{def}}{=} D$. Let \mathcal{E}^{def} be the set of processes defined with this syntax. Given a process D , $\text{const}(D)$ denotes all the constants which occur in D . We say that a process D is *constant-free* if $\text{const}(D) = \emptyset$.

In order to define the semantics of the processes in \mathcal{E}^{def} we add to the rules of Figure 1 the following rule to deal with constant definitions.

$$\text{Constant} \frac{}{Z \xrightarrow{\tau} D} \text{ if } Z \stackrel{\text{def}}{=} D$$

This rule tells us that if $Z \stackrel{\text{def}}{=} D$ then Z performs a τ transition and then behaves as D .

Example 3. Let Z be the constant defined in Example 2. By applying once the rule Constant we obtain that $Z \xrightarrow{\tau} a.Z + b.\mathbf{0}$, then either $a.Z + b.\mathbf{0} \xrightarrow{b} \mathbf{0}$ or $a.Z + b.\mathbf{0} \xrightarrow{a} Z$. In the second case we can apply again the rule Constant.

All the processes in \mathcal{E}^{def} can be translated into an equivalent (bisimilar) process of the language $\mathcal{E}^!$ presented in Section 2.1 (i.e., into a process with restriction and replication and without constant definition).

We briefly recall how the encoding which removes the constant definitions works. Let Z_1, \dots, Z_n be n constants defined as $Z_i \stackrel{\text{def}}{=} D_i$, where for all $i = 1, \dots, n$ $\text{const}(D_i) \subseteq \{Z_1, \dots, Z_n\}$. Let $S = \{\sigma_1, \overline{\sigma}_1, \dots, \sigma_n, \overline{\sigma}_n\}$ be a new set of actions disjoint from Act . We associate to the constant Z_i the actions σ_i and $\overline{\sigma}_i$ and we introduce the notation²:

$$\widehat{Z}_i \equiv !(\sigma_i.D_i[\overline{\sigma}_1.\mathbf{0}/Z_1, \dots, \overline{\sigma}_n.\mathbf{0}/Z_n]),$$

where in D_i each constant Z_j is replaced by the constant-free expression $\overline{\sigma}_j.\mathbf{0}$. Since $\text{const}(D_i) \subseteq \{Z_1, \dots, Z_n\}$, \widehat{Z}_i is a constant-free expression.

Definition 4 (Encoding of \mathcal{E}^{def}). Let $D \in \mathcal{E}^{\text{def}}$ be a process with $\text{const}(D) \subseteq \{Z_1, \dots, Z_n\}$. Its encoding $\llbracket D \rrbracket$ is the constant-free process

$$\llbracket D \rrbracket \equiv (D[\overline{\sigma}_1.\mathbf{0}/Z_1, \dots, \overline{\sigma}_n.\mathbf{0}/Z_n]|\widehat{Z}_1|\dots|\widehat{Z}_n) \setminus S.$$

² We use the notation $D[Z_1, \dots, Z_n]$ when we want to stress the fact that the constants Z_1, \dots, Z_n can occur in D .

In particular, when D is one of the Z_i 's we obtain

$$\llbracket Z_i \rrbracket \equiv (\overline{\sigma}_i.\mathbf{0}|\widehat{Z}_1|\dots|\widehat{Z}_n) \setminus S.$$

Example 4. Let Z be the constant defined in Example 2. We obtain that the encoding of Z is $\llbracket Z \rrbracket \equiv (\overline{\sigma}.\mathbf{0}|!(\sigma.\mathbf{0}.(a.\overline{\sigma}.\mathbf{0} + b.\mathbf{0}))) \setminus S$. Notice that \widehat{Z} , which is $\widehat{Z} \equiv !(\sigma.\mathbf{0}.(a.\overline{\sigma}.\mathbf{0} + b.\mathbf{0}))$, is different from $\llbracket Z \rrbracket$.

Remark 1. In the encoding, the action $\overline{\sigma}_i$ is used to make a ‘‘call to the procedure’’ Z_i which is represented by \widehat{Z}_i . The encoding does not work in the full CCS, since the scope of the restrictions and renamings is not enlarged to the \widehat{Z}_i . Consider for instance a constant Z defined as $Z \stackrel{\text{def}}{=} a.Z$ and the process $E \equiv (Z) \setminus \{a\}$. The process E can only perform a τ action, then it terminates. If we apply our encoding to E we obtain $\llbracket E \rrbracket \equiv ((\overline{\sigma}.\mathbf{0}) \setminus \{a\})|!(\sigma.a.\overline{\sigma}.\mathbf{0}) \setminus S$. Differently from E , the process $\llbracket E \rrbracket$ performs a τ , and then it is able to perform an action a , since in \widehat{Z} the action a is allowed. Actually, we can overcome this problem and define a correct translation for E (see Definition 5). Another process which cannot be translated can be obtained using two mutual recursive constant definitions

$$X \stackrel{\text{def}}{=} (a.X|b.\overline{a}.Y) \setminus \{a, \overline{a}\} \qquad Y \stackrel{\text{def}}{=} (b.Y|a.\overline{b}.X) \setminus \{b, \overline{b}\}$$

The process $F \equiv X$ can perform only b and τ actions. Its encoding would be the process $\llbracket F \rrbracket$ defined as

$$(\overline{\sigma}_X.\mathbf{0}|!(\sigma_X.((a.\overline{\sigma}_X.\mathbf{0}|b.\overline{a}.\overline{\sigma}_Y.\mathbf{0}) \setminus \{a, \overline{a}\}))|!(\sigma_Y.((b.\overline{\sigma}_Y.\mathbf{0}|a.\overline{b}.\overline{\sigma}_X.\mathbf{0}) \setminus \{b, \overline{b}\}))) \setminus S.$$

The process $\llbracket F \rrbracket$ can perform also a actions, since the restriction on a is not applied to \widehat{Y} . The solution we will apply later to enlarge the encoding cannot be applied to this process.

The following theorem states the observational equivalence between D and $\llbracket D \rrbracket$ when D belongs to \mathcal{E}^{def} . Since $D \in \mathcal{E}^{\text{def}}$ and $\llbracket D \rrbracket \in \mathcal{E}^!$ the bisimulation we establish is a relation on $\mathcal{E}^{\text{def}} \times \mathcal{E}^!$.

Theorem 3. *For each $D \in \mathcal{E}^{\text{def}}$ it holds $D \sim \llbracket D \rrbracket$.*

The actions σ_i 's introduced in the encoding are neither high nor low level actions. They are used only in the encoding, in order to obtain constant free-processes, but they are not visible outside because of the outmost restriction. Indeed, they are introduced only to *fire* infinitely many times the actions of the D_i 's. Nevertheless, we have to decide how to treat them in the definition of the attackers and in the definition of the low level observational equivalence. We consider this issue in the next section.

Before moving to our security property we show how to apply the encoding to a richer language in which restriction and renaming can be used ‘‘outside’’ the recursive definitions. In particular, consider all the processes E defined by the following productions:

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

where $Z \in \mathcal{C}$ is a constant which must be associated to a definition $Z \stackrel{\text{def}}{=} D$, with $D \in \mathcal{E}^{\text{def}}$. Let $\mathcal{E}^{\text{def!}}$ be the set of processes defined with this syntax.

Since the constants are defined using processes in \mathcal{E}^{def} , by Theorem 3, we have that $Z \sim \llbracket Z \rrbracket$. Observing that \sim is a congruence on our language we immediately get that the following encoding can be applied to the processes in $\mathcal{E}^{\text{def!}}$.

Definition 5 (Encoding of $\mathcal{E}^{\text{def!}}$). Let $E \in \mathcal{E}^{\text{def!}}$ be a process with $\text{const}(E) \subseteq \{Z_1, \dots, Z_n\}$ its encoding $\{\!\{E\}\!\}$ is the constant-free process

$$\{\!\{E\}\!\} \equiv E[\llbracket Z_1 \rrbracket / Z_1, \dots, \llbracket Z_n \rrbracket / Z_n].$$

Corollary 2. For each $E \in \mathcal{E}^{\text{def!}}$ it holds $E \sim \{\!\{E\}\!\}$.

Example 5. Consider the constant Z and the process E defined in Remark 1. The process E is in $\mathcal{E}^{\text{def!}}$. Its encoding is $\{\!\{E\}\!\} \equiv ((\bar{\sigma}.\mathbf{0}!(\sigma.a.\bar{\sigma}.\mathbf{0})) \setminus S) \setminus \{a\}$. Now, we correctly obtain that E performs a τ transitions, then it terminates.

The constants X and Y of Remark 1 do not belong to $\mathcal{E}^{\text{def!}}$. In fact, in order to translate X we would need a correct encoding of Y , and this is not possible without a correct encoding of X , i.e., we enter in a loop.

4.2 P_BNDC and Definitions

Let $Act = L \cup H \cup \{\tau\}$ as defined in Section 2.1. Let S be a new set of (synchronization) actions such that $S \cap Act = \emptyset$ and $\bar{S} = S$, i.e., S is closed with respect to the complementation operation. In what follows we consider as set of actions $Act = L \cup H \cup \{\tau\} \cup S$. Moreover, we require that if f is a relabelling function, then $\forall \sigma \in S, f(\sigma) = \sigma$. As previously observed the actions of S do not represent ‘real’ actions, but they are only instrumental for the encoding. The processes we start with have no actions in S , while their encodings do. For this reason it is necessary to decide how to treat S with respect to our security notions. In order to keep the compositionality of P_BNDC it is convenient to assimilate them to low level actions. Therefore, the high level attacker cannot perform them and the low level user can observe them. In this way we can treat in a compositional way also processes in which these actions occur. In particular, we extend the concept of weak bisimulation on low actions considering the actions in S as if they were actions in L . With a slight abuse of notation from now on we say that two processes $E, F \in \mathcal{E}^{\text{def!}}$ (built also using actions in S) are *weakly bisimilar on low actions*, denoted by $E \approx_l F$, if there exists a symmetric binary relation $\mathcal{R} \subseteq \mathcal{E}^{\text{def!}} \times \mathcal{E}^{\text{def!}}$ such that if $(E, F) \in \mathcal{R}$, then for all $a \in L \cup S \cup \{\tau\}$,

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

Clearly \approx_l is still the largest weak bisimulation on low actions and it is an equivalence relation. Moreover it is still true that $E \approx_l F$ iff $E \setminus H \approx F \setminus H$.

Using this definition of \approx_l the notions of $BNDC$ and P_BNDC can be consistently transposed. Notice that using these extended definitions Theorem 4

and Theorem 1 continue to hold. As far as Lemma 1 is concerned some trivial changes are necessary. In particular, let $\mathcal{E}_{HS}^{\text{def!}}$ ($\mathcal{E}_{HS}^{\text{def}}$) be the set of all processes in $\mathcal{E}^{\text{def!}}$ (\mathcal{E}^{def}) constructed over $H \cup S \cup \{\tau\}$. Similarly, let $\mathcal{E}_{LS}^{\text{def!}}$ ($\mathcal{E}_{LS}^{\text{def}}$) be the set of all processes constructed over $L \cup S \cup \{\tau\}$ and $\mathcal{E}_{HL}^{\text{def!}}$ ($\mathcal{E}_{HL}^{\text{def}}$) be the set of all processes constructed over $L \cup H \cup \{\tau\}$. In the first sentence of Lemma 1 it is necessary to consider constant-free processes in $\mathcal{E}_{LS}^{\text{def!}} \cup \mathcal{E}_H^{\text{def!}}$. In the third sentence the actions a_i 's can range over $L \cup S \cup \{\tau\}$. Moreover, from Theorem 3 we immediately get the following result.

Corollary 3. *Let Z_1, \dots, Z_n be constants defined as $Z_i \stackrel{\text{def}}{=} D_i$, with $D_i \in \mathcal{E}_{HL}^{\text{def}}$ for $i = 1, \dots, n$. If for all $i = 1, \dots, n$ it holds $\text{const}(D_i) \subseteq \{Z_1, \dots, Z_n\}$ and $\llbracket Z_i \rrbracket \in P\text{-BNDC}$, then all the Z_i 's are $P\text{-BNDC}$.*

4.3 Extension of the Proof System to Processes with Definitions

In order to deal with the language extended with the actions in S and with the constant definitions we have to modify some of the rules of the proof system described in Section 3.2 and to add new rules to deal with constant definitions. In particular, we change the rules (*Low*) and (*Choice*) by considering $L \cup S$ instead of L and by adding “ E is constant-free” to the rules (*Low*) and (*High*).

Then we add the following rules to deal with constant definitions

$$\frac{}{E \setminus S \in \mathcal{HP}[\emptyset]} \quad E \in \mathcal{E}_{HS}^{\text{def!}}, E \text{ is constant-free} \quad (\text{High2})$$

$$\frac{\llbracket X_i \rrbracket \in \mathcal{HP}[A]}{X_i \in \mathcal{HP}[A]} \quad (X_i \stackrel{\text{def}}{=} D_i)_{i=1}^n, D_i \in \mathcal{E}_{HL}^{\text{def}} \quad (\text{Const})$$

where $\llbracket X_i \rrbracket$ is a constant-free process.

We call $\text{System}^{\text{def!}}$ the modified system. Corollary 3 ensures its correctness.

Example 6. Consider the channel C as defined in [3] (see [19]) and its encoding.

$$C = in_0.(\overline{out_0}.C + \tau.C) + in_1.(\overline{out_1}.C + \tau.C)$$

$$\llbracket C \rrbracket \equiv (\overline{\sigma}.\mathbf{0} \mid !(\sigma.(in_0.(\overline{out_0}.\overline{\sigma}.\mathbf{0} + \tau.\overline{\sigma}.\mathbf{0}) + in_1.(\overline{out_1}.\overline{\sigma}.\mathbf{0} + \tau.\overline{\sigma}.\mathbf{0})))) \setminus S$$

It is easy to see that we can derive $C \in \mathcal{HP}[\emptyset]$ in our extended proof system. Notice that the process CH described in Example 1 is exactly the process we obtain after a τ transition of $\llbracket C \rrbracket$.

Corollary 4. *Let $E \in \mathcal{E}^{\text{def!}}$ be a process. If there exists a proof of $E \in \mathcal{HP}[\emptyset]$ in $\text{System}^{\text{def!}}$, then E is $P\text{-BNDC}$.*

By exploiting the result of Corollary 2 we can add the derived rule below, which can be used to shorten derivations involving constant definitions:

$$\frac{\{\{E\}\} \in \mathcal{HP}[A]}{E \in \mathcal{HP}[A]} \quad E \in \mathcal{E}_{HL}^{\text{def!}}(\text{Trans})$$

Example 7. Let Z be defined as $Z \stackrel{\text{def}}{=} l.Z + h.l.\mathbf{0} + \tau.l.\mathbf{0}$ and consider the process $E \equiv l.Z$. By applying rule (*Trans*) we can directly prove that E is P_BNDC without explicitly prove that $\llbracket Z \rrbracket$ is P_BNDC .

5 Conclusions

In this paper we study the class of P_BNDC processes written in a variant of SPA language where recursive processes are defined by means of replications instead of constant definitions. The modified language is slightly less powerful than the original one, but the loss of expressive power is largely compensated by the compositionality result we obtain. In fact, we proved that the class of P_BNDC processes is compositional with respect to replication. This result allows us to define a proof system which provides a very efficient technique for the stepwise development and the verification of recursively defined P_BNDC processes. We also identify a class of constants definitions which can be safely added to our language and treated by an extended proof system.

As already noticed in [4], there are many other approaches to the verification of information flow properties. In the literature we found only another example of a proof system for security proposed by Martinelli [16] which deals only with finite processes. Other verification techniques for information flow security are based on types (see, e.g., [23,14]) and control flow analysis (see, e.g., [2,6]). However, most of them are concerned with different models, e.g., trace semantics (see, e.g., [8,18]).

References

1. M. Abadi and L. Lamport. Conjoining Specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 17(3):507–535, May 1995.
2. C. Bodei, P. Degano, F. Nielson, and H. Nielson. Static Analysis for the π -calculus with Applications to Security. *Information and Computation*, 168(1):68–92, 2001.
3. A. Bossi, R. Focardi, C. Piazza, and S. Rossi. Transforming Processes to Ensure and Check Information Flow Security. In H. Kirchner and C. Ringeissen, editors, *Int. Conference on Algebraic Methodology and Software Technology (AMAST'02)*, volume 2422 of *LNCS*, pages 271–286. Springer-Verlag, 2002.
4. A. Bossi, R. Focardi, C. Piazza, and S. Rossi. A Proof System for Information Flow Security. In M. Leuschel, editor, *Logic Based Program Development and Transformation*, volume 2664 of *LNCS*. Springer-Verlag, 2003. To appear.
5. A. Bossi, D. Macedonio, C. Piazza, and S. Rossi. P_BNDC and Replication. Technical Report CS-2003-6, Dipartimento di Informatica, Università Ca' Foscari di Venezia, Italy, 2003. <http://www.dsi.unive.it/~mace/TechRep2003-6.ps>.
6. C. Braghin, A. Cortesi, and R. Focardi. Control Flow Analysis of Mobile Ambients with Security Boundaries. In *Proc. Int. Conference on Formal Methods for Open Object-Based Distributed Systems (IFIPM'02)*, pages 197–212. Kluwer, 2002.
7. N. Busi, M. Gabbrielli, and G. Zavattaro. Replication vs. Recursive Definitions in Channel Based Calculi. In *Proc. of Int. Colloquium on Automata, Languages and Programming (ICALP'03)*. LNCS, 2003. To appear.

8. McCullough. D. A Hookup Theorem for Multilevel Security. *IEEE Transactions on Software Engineering*, 16(6):563–568, 1990.
9. R. Focardi and R. Gorrieri. Classification of Security Properties (Part I: Information Flow). In R. Focardi and R. Gorrieri, editors, *Foundations of Security Analysis and Design*, volume 2171 of *LNCS*. Springer-Verlag, 2001.
10. R. Focardi, R. Gorrieri, and F. Martinelli. Non Interference for the Analysis of Cryptographic Protocols. In U. Montanari, J. D. P. Rolim, and E. Welzl, editors, *Proc. of Int. Colloquium on Automata, Languages and Programming (ICALP'00)*, volume 1853 of *LNCS*, pages 744–755. Springer-Verlag, 2000.
11. R. Focardi and S. Rossi. Information Flow Security in Dynamic Contexts. In *Proc. of the IEEE Computer Security Foundations Workshop (CSFW'02)*, pages 307–319. IEEE Comp. Soc. Press, 2002.
12. S. N. Foley. A Universal Theory of Information Flow. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 116–122. IEEE Comp. Soc. Press, 1987.
13. J. A. Goguen and J. Meseguer. Security Policies and Security Models. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 11–20. IEEE Comp. Soc. Press, 1982.
14. M. Hennessy and J. Riely. Information Flow vs. Resource Access in the Asynchronous Pi-calculus. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 24(5):566–591, 2002.
15. H. Mantel. On the Composition of Secure Systems. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 88–101. IEEE Comp. Soc. Press, 2002.
16. F. Martinelli. Partial Model Checking and Theorem Proving for Ensuring Security Properties. In *Proc. of the IEEE Computer Security Foundations Workshop (CSFW'98)*, pages 44–52. IEEE Comp. Soc. Press, 1998.
17. J. McLean. Security Models and Information Flow. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 180–187. IEEE Comp. Soc. Press, 1990.
18. J. McLean. A General Theory of Composition for a Class of “Possibilistic” Security Properties. *IEEE Transactions on Software Engineering*, 22(1):53–67, 1996.
19. R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.
20. R. Milner. The Polyadic Pi-calculus: a tutorial. In F. L. Bauer, W. Brauer, and H. Schwichtenberg, editors, *Logic and Algebra of Specification*, pages 203–246. Springer-Verlag, 1993.
21. P. Ryan and S. Schneider. Process Algebra and Non-Interference. *Journal of Computer Security*, 9(1/2):75–103, 2001.
22. A. Sabelfeld and A. C. Myers. Language-Based Information-Flow Security. *IEEE Journal on Selected Areas in Communication*, 21(1):5–19, 2003.
23. A. Sabelfeld and D. Sands. Probabilistic Noninterference for Multi-threaded Programs. In *Proc. of the IEEE Computer Security Foundations Workshop (CSFW'00)*, pages 200–215. IEEE Comp. Soc. Press, 2000.
24. D. Sangiorgi and D. Walker. *The π -calculus*. Cambridge University Press, 2001.
25. J. Widom, D. Gries, and F. B. Schneider. Trace-based Network Proof Systems: Expressiveness and Completeness. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 14(3):396–416, 1992.
26. A. Zakinthinos and E. S. Lee. A General Theory of Security Properties. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 74–102. IEEE Comp. Soc. Press, 1997.

Appendix

Basic Definitions

Definition 6 (Weak Bisimulation). A symmetric binary relation $\mathcal{R} \subseteq \mathcal{E}^! \times \mathcal{E}^!$ over processes is a weak bisimulation if $(E, F) \in \mathcal{R}$ implies, for all $a \in \text{Act}$,

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

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

Definition 7 (Strong Bisimulation). A symmetric binary relation $\mathcal{R} \subseteq \mathcal{E}^! \times \mathcal{E}^!$ over processes is a strong bisimulation if $(E, F) \in \mathcal{R}$ implies, for all $a \in \text{Act}$,

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

Two processes $E, F \in \mathcal{E}^!$ are strong bisimilar, denoted by $E \sim F$, if there exists a strong bisimulation \mathcal{R} containing the pair (E, F) .

Proofs

Although the decidability of *BNDC* is still an open problem, *P-BNDC* is decidable (in polynomial time) as shown in [11]. In [4] another decidable characterization of *P-BNDC* processes has been proposed. It allows us to express *P-BNDC* in terms of a local property of high level actions and it recalls the unwinding conditions proposed in other settings. Also if we are using a variation of the SPA, with replications instead of constant definitions, the characterization presented in [4] holds.

Theorem 4 (Unwinding). Let $E \in \mathcal{E}^!$. $E \in P\text{-BNDC}$ iff if $E \rightsquigarrow E_i \xrightarrow{b} E_j$, then $E_i \xrightarrow{\hat{\tau}} E_k$ and $E_j \approx_l E_k$.

The proof of Theorem 1, which is the main result of Section 3, strongly depends on the previous unwinding condition and the subsequent two characterization lemmas.

Lemma 2. Let $E \in \mathcal{E}^!$ be a process. If $!E \rightsquigarrow E'$, then there exist $n \geq 0$ and E_1, \dots, E_n such that $E \rightsquigarrow E_i$, for $i = 1, \dots, n$ and $E' \equiv E_1 | E_2 | \dots | E_n | !E$.

Proof. By induction on the length ln of \rightsquigarrow .

If $ln = 0$, then $E' \equiv !E$, hence we have the thesis with $n = 0$.

Let us assume that we have proved the thesis for all the $ln \leq m$. Let $ln = m + 1$. This means that there exists E'' such that $!E \rightsquigarrow E''$ with m steps and $E'' \xrightarrow{a} E'$. By inductive hypothesis there exist $n \geq 0$ and E_1, \dots, E_n such that $E \rightsquigarrow E_i$, for $i = 1, \dots, n$ and $E'' \equiv E_1 | E_2 | \dots | E_n | !E$. If the action a is performed by one of the E_i 's, say E_1 , we have the thesis, since $E \rightsquigarrow E_1 \xrightarrow{a} E'_1$ and $E' \equiv E'_1 | E_2 | \dots | E_n | !E$. Similarly we obtain the thesis if $a = \tau$ is a synchronization between two of the E_i 's. If the action a is performed by $!E$ applying the first rule of Replication, then $E \xrightarrow{a} E_{n+1}$ and $E' \equiv E_1 | E_2 | \dots | E_n | E_{n+1} | !E$. Similarly we

obtain the thesis in the remaining two cases, i.e. if a is performed by $!E$ applying the second rule of Replication or if a is a synchronization between one of the E_i 's and $!E$. \square

Lemma 3. Let E be a P_BNDC process and $\forall i \in \{1, \dots, n\} F_i, G_i$ be reachable from E . If $\forall i \in \{1, \dots, n\} F_i \approx_l G_i$ then $F_1|F_2 \dots |F_n|!E \approx_l G_1|G_2 \dots |G_n|!E$.

Proof. Let E be a P_BNDC process and

$$\mathcal{R} = \{(F_1|F_2 \dots |F_n|!E, G_1|G_2 \dots |G_n|!P) \mid n \geq 0, \forall i \in \{1, \dots, n\} F_i, G_i \text{ are reachable from } E \text{ and } F_i \approx_l G_i\}.$$

We prove that \mathcal{R} is a weak bisimulation on low actions. We have to consider the following three cases.

- No synchronization. Let $a \in L \cup \{\tau\}$ and assume $F_1|F_2 \dots |F_n|!E \xrightarrow{a} E'$. There are three subcases.
 - The action is performed by one of the F_i 's, say F_1 . Hence $E' \equiv F_1'|F_2 \dots |F_n|!E$ where $F_i \xrightarrow{a} F_i'$. Since $F_1 \approx_l G_1$ there exists G_1' such that $G_1 \xrightarrow{\hat{a}} G_1'$ and $G_1' \approx_l G_1$. Thus $G_1|G_2 \dots |G_n|!E \xrightarrow{\hat{a}} G_1'|G_2 \dots |G_n|!E$ and $(F_1'|F_2 \dots |F_n|!E, G_1'|G_2 \dots |G_n|!E) \in \mathcal{R}$.
 - The action is performed by $!E$. If $a \in L$, then $E' \equiv F_1|F_2 \dots |F_n|F_{n+1}|!E$ where $E \xrightarrow{a} F_{n+1}$. Clearly $G_1|G_2 \dots |G_n|!E \xrightarrow{\hat{a}} G_1|G_2 \dots |G_n|F_{n+1}|!E$ and $(F_1|F_2 \dots |F_n|F_{n+1}|!E, G_1|G_2 \dots |G_n|F_{n+1}|!E) \in \mathcal{R}$. If $a = \tau$, we similarly get the thesis applying the second rule for the semantics of the replication.
- Synchronization on low actions. There are two subcases.
 - Two of the F_i 's synchronize. Without loss of generality we can assume that $F_1|F_2 \dots |F_n|!E \xrightarrow{\tau} F_1'|F_2' \dots |F_n|!E$ where $F_1 \xrightarrow{a} F_1'$ and $F_2 \xrightarrow{\bar{a}} F_2'$. Since $F_1 \approx_l G_1$ and $F_2 \approx_l G_2$ there exist G_1', G_2' such that $G_1 \xrightarrow{\hat{a}} G_1', G_1' \approx_l F_1', G_2 \xrightarrow{\hat{\bar{a}}} G_2', G_2' \approx_l F_2'$. Then $G_1|G_2 \dots |G_n|!E \xrightarrow{\hat{\tau}} G_1'|G_2' \dots |G_n|!E$ and $(F_1'|F_2' \dots |F_n|!E, G_1'|G_2' \dots |G_n|!E) \in \mathcal{R}$.
 - One of the F_i 's synchronizes with $!E$. Similar to the previous case.
- Synchronization on high actions. There are two subcases.
 - Two of the F_i 's synchronize. Without loss of generality we can assume that $F_1|F_2 \dots |F_n|!E \xrightarrow{\tau} F_1'|F_2' \dots |F_n|!E$ where $F_1 \xrightarrow{h} F_1'$ and $F_2 \xrightarrow{\bar{h}} F_2', h \in H$. Since E is P_BNDC and both F_1 and F_2 are reachable from E , by the unwinding Theorem 4 there exist G_1'' and G_2'' such that

$$F_i \xrightarrow{\hat{\tau}} G_i'', F_i' \approx_l G_i'', i = 1, 2.$$

Since $F_i \approx_l G_i, i = 1, 2$, there exist G_1' and G_2' such that

$$G_i \xrightarrow{\hat{\tau}} G_i', G_i'' \approx_l G_i', i = 1, 2.$$

Hence $F_i' \approx_l G_i', i = 1, 2$, and $G_1|G_2 \dots |G_n|!E \xrightarrow{\hat{\tau}} G_1'|G_2' \dots |G_n|!E$ where $(F_1'|F_2' \dots |F_n|!E, G_1'|G_2' \dots |G_n|!E) \in \mathcal{R}$.

- one of the F_i 's synchronizes with $!E$. Without loss of generality we can assume that $F_1|F_2 \dots |F_n|!E \xrightarrow{\tau} F'_1|F_2 \dots |F_n|E'|!E$ where $F_1 \xrightarrow{h} F'_1$ and $!E \xrightarrow{\bar{h}} E'|!E$, $h \in H$. Hence $E \xrightarrow{\bar{h}} E'$. Since E is P_BNDC and both F_1 and E' are reachable from E , by the unwinding Theorem 4 there exist G''_1 and E'' such that $F_1 \xrightarrow{\hat{\tau}} G''_1$, $F'_1 \approx_l G''_1$ and $E \xrightarrow{\hat{\tau}} E''$, $E' \approx_l E''$. Since $F_1 \approx_l G_1$, there exists G'_1 such that $G_1 \xrightarrow{\hat{\tau}} G'_1$, $G''_1 \approx_l G'_1$. Hence $F'_1 \approx_l G'_1$, and $G_1|G_2 \dots |G_n|!E \xrightarrow{\hat{\tau}} G'_1|G_2 \dots |E''|!E$ where it holds $(F'_1|F_2 \dots |F_n|E'|!E, G'_1|G_2 \dots |E''|!E) \in \mathcal{R}$.

□

Theorem 1. Let $E \in \mathcal{E}^!$ be a process. If $E \in P_BNDC$, then $!E \in P_BNDC$.

Proof. We prove that if F is reachable from $!E$, then F is P_BNDC . We prove that F satisfies the unwinding condition (see Theorem 4). By Lemma 2 we have that $F \equiv F_1|F_2 \dots |F_n|!E$. Let $F_1|F_2 \dots |F_n|!E \xrightarrow{h} F'$. If the action is performed by one of the F_i 's, say F_1 , then $F' \equiv F'_1|F_2 \dots |F_n|!E$ where $F_1 \xrightarrow{h} F'_1$. By the unwinding condition on F_1 , there exists G'_1 such that $F_1 \xrightarrow{\hat{\tau}} G'_1$, $G'_1 \approx_l F'_1$. Hence $F_1|F_2 \dots |F_n|!E \xrightarrow{\hat{\tau}} G'_1|F_2 \dots |F_n|!E$ and, by Lemma 3, $F'_1|F_2 \dots |F_n|!E \approx_l G'_1|F_2 \dots |F_n|!E$. If the action is performed by $!E$, then $F' \equiv F_1|F_2 \dots |F_n|E'|!E$ and $E \xrightarrow{h} E'$. By the unwinding condition on E , there exists E'' such that $E \xrightarrow{\hat{\tau}} E''$, $E'' \approx_l E'$. Hence $F_1|F_2 \dots |F_n|!E \xrightarrow{\hat{\tau}} F_1|F_2 \dots |F_n|E''|!E$ and, by Lemma 3, $F_1|F_2 \dots |F_n|E'|!E \approx_l F_1|F_2 \dots |F_n|E''|!E$. □