# Comparing Two Information Flow Security Properties

Riccardo Focardi
Dipartimento di Scienze dell'Informazione
Università di Bologna
via mura A. Zamboni 7, I-40127, Bologna, Italy
focardi@cs.unibo.it

## Abstract

*In this paper we compare two information flow security properties: the lazy security (L-Sec) [11] and the Bisimulation Non-deducibility on Compositions (BNDC) [4]. To make this we define the Failure Non-deducibility on Compositions, a failure semantics version of the BNDC. The common specification language used for the comparison is the Security Process Algebra [4], an extension of CCS [8] which permits to describe systems where actions belong to two different levels of confidentiality. We prove that BNDC applied to a restricted class of systems, the low-deterministic and non-divergent ones, is equal to L-Sec. So these two properties, which are based on quite different underlying intuitions, become the same if we add some conditions to BNDC.*

## 1 Introduction

In this paper we compare two information flow security properties: the *lazy security* (*L-Sec*) [11] and the *Bisimulation Non-deducibility on Compositions* (*BNDC*) [4]. Intuitively, the first one requires that the obscuring of high level actions by interleaving does not introduce any nondeterminism in the system; the second one implies that high level users cannot modify what a low level user can see of the system.

To make this we introduce the *Failure Non-deducibility on Compositions* (*FNDC*), a failure semantics version of *BNDC*. The specification language used to compare the properties is the *Security Process Algebra* (SPA), an extension of CCS [8]. This language permits to describe systems where actions belong to two different levels of confidentiality, and it has been introduced in [4] in order to compare and classify a number of information flow security properties.

The main result in this work is that *BNDC* is equal to *L-Sec*, when applied to a particular class of systems: the *low-deterministic* and *non-divergent* ones. In the failure setting, a system is low-deterministic if after a certain low level trace $\gamma$, no low level action $l$ can be both accepted and refused. We have that every *L-Sec* system is low-deterministic. It is interesting to observe how these two properties, which are based on quite different underlying intuitions, become the same when dealing with processes which are low-deterministic and non-divergent.

The paper is organized as follows. In Section 2 we present SPA and semantic equivalences. In Section 3 we define *L-Sec* in the SPA setting showing that its action is restricted to low-deterministic processes. Section 4 describes the failure and bisimulation based Non Deducibility on Composition. Section 5 compares *L-Sec* and *BNDC* in the class of low-deterministic and non-divergent systems. Finally, Section 6 contains some concluding remarks on the automatic verification of the two compared security properties.

## 2 SPA and Semantic Equivalences

In the following, systems will be specified using the *Security Process Algebra*, an extension of Milner's CCS [8]. SPA has two additional operators, namely the hiding operator $E/L$ of CSP [7] and the (new) input restriction operator $E \setminus_I L$, which are useful in characterizing some security properties in an algebraic style. Moreover the set of visible actions is partitioned into high and low level actions in order to specify multilevel systems. [1]

SPA syntax is based on the following elements: a set $I = \{a, b, \ldots\}$ of *input* actions, a set $O = \{\bar{a}, \bar{b}, \ldots\}$ of *output* actions, a set $\mathcal{L} = I \cup O$ of *visible* actions, ranged over by $\alpha$, and the usual function $\bar{\cdot} : \mathcal{L} \rightarrow \mathcal{L}$ such that $a \in I \Longrightarrow \bar{a} \in O$ and $\bar{a} \in O \Longrightarrow \bar{\bar{a}} = a \in I$; two sets $Act_H$ and $Act_L$ of high and low level actions such that $\overline{Act_H} = Act_H$, $\overline{Act_L} = Act_L$, $Act_H \cup Act_L = \mathcal{L}$

[1] Actually, only two-level systems can be specified. Note that this is not a real limitation because it is always possible to deal with the multilevel case by grouping – in several ways – the various levels in two clusters.

and $Act_H \cap Act_L = \emptyset$ where $\bar{L} \stackrel{\text{def}}{=} \{\bar{a} : a \in L\}$; a set $Act = \mathcal{L} \cup \{\tau\}$ of actions ($\tau$ is the internal, invisible action), ranged over by $\mu$; a set $K$ of constants, ranged over by $Z$. The syntax of SPA *agents* is defined as follows:

$$E ::= \underline{0} \mid \mu.E \mid E + E \mid E|E \mid E \setminus L \mid E \setminus_I L \mid \\ \mid E/L \mid E[f] \mid Z$$

where $L \subseteq \mathcal{L}$ and $f : Act \to Act$ is such that $f(\bar{\alpha}) = \overline{f(\alpha)}$, $f(\tau) = \tau$. Moreover, for every constant $Z$ there must be the corresponding definition: $Z \stackrel{\text{def}}{=} E$. The meaning of $\underline{0}$, $\mu.E$, $E+E$, $E|E$, $E \setminus L$, $E[f]$ and $Z \stackrel{\text{def}}{=} E$ is as for CCS [8]. Intuitively, $\underline{0}$ is the empty process, which cannot do any action; $\mu.E$ can do an action $\mu$ and then behaves like $E$; $E_1 + E_2$ can alternatively choose [2] to behave like $E_1$ or $E_2$; $E_1|E_2$ is the parallel composition of $E_1$ and $E_2$, where the executions of the two systems are interleaved, possibly synchronized on complementary input/output actions, producing an internal $\tau$. $E \setminus L$ can execute all the actions $E$ is able to do, provided that they do not belong to $L \cup \bar{L}$, while $E \setminus_I L$ requires that the actions of $E$ do not belong to $L \cap I$; $E/L$ turns all the actions in $L$ into internal $\tau$'s; if $E$ can execute action $\mu$, then $E[f]$ performs $f(\mu)$; finally, $Z$ does what $E$ does, when $Z \stackrel{\text{def}}{=} E$.

Let $\mathcal{E}$ be the set of SPA agents, ranged over by $E$, $F$. Let $\mathcal{L}(E)$ denote the *sort* of $E$, i.e., the set of the (possibly executable) actions occurring syntactically in $E$. The sets of high level agents and low level ones are defined as $\mathcal{E}_H \stackrel{\text{def}}{=} \{E \in \mathcal{E} \mid \mathcal{L}(E) \subseteq Act_H \cup \{\tau\}\}$ and $\mathcal{E}_L \stackrel{\text{def}}{=} \{E \in \mathcal{E} \mid \mathcal{L}(E) \subseteq Act_L \cup \{\tau\}\}$, respectively. The operational semantics of SPA is given (as usual) associating to each agent a particular state of the labelled transition system $(\mathcal{E}, Act, \to)$ where $\to \subseteq \mathcal{E} \times Act \times \mathcal{E}$ and, intuitively, $E \stackrel{\mu}{\to} E'$ means that agent $E$ can execute $\mu$ moving to $E'$ (see [4] for more details).

In the following the expression $E \stackrel{\mu}{\Longrightarrow} E'$ is a shorthand for $E(\stackrel{\tau}{\to})^* E_1 \stackrel{\mu}{\to} E_2(\stackrel{\tau}{\to})^* E'$, where $(\stackrel{\tau}{\to})^*$ denotes a (possibly empty) sequence of $\tau$ labelled transitions. Moreover $E \stackrel{\mu}{\not\Longrightarrow}$ means that $\nexists E'$ such that $E \stackrel{\mu}{\Longrightarrow} E'$ and $E \stackrel{K}{\not\Longrightarrow}$ with $K \subseteq \mathcal{L}$ stands for $\forall \mu \in K, E \stackrel{\mu}{\not\Longrightarrow}$. We also extend the '$\Longrightarrow$' notation to sequences of actions; $E \stackrel{\gamma}{\Longrightarrow} E'$ with $\gamma \in \mathcal{L}^+, \gamma = \gamma_1\gamma_2 \ldots \gamma_n$ means that $\exists E_1, E_2, \ldots, E_{n-1}$ such that $E \stackrel{\gamma_1}{\Longrightarrow} E_1 \stackrel{\gamma_2}{\Longrightarrow} \ldots \stackrel{\gamma_{n-1}}{\Longrightarrow} E_{n-1} \stackrel{\gamma_n}{\Longrightarrow} E'$. For the empty sequence $\langle\rangle$ we have that $E \stackrel{\langle\rangle}{\Longrightarrow} E'$ stands for $E(\stackrel{\tau}{\to})^* E'$.

We recall here the definition of traces and failure equivalence [1]

**Definition 2.1** *A trace of a process is a sequence of actions that it can execute. The set of traces is defined as follows:* $traces(E) \stackrel{\text{def}}{=} \{\gamma \in \mathcal{L}^* \mid \exists E' : E \stackrel{\gamma}{\Longrightarrow} E'\}$. ■

**Definition 2.2** *If $\gamma \in traces(E)$ and if, after executing $\gamma$, $E$ can refuse all the actions in set $X \subseteq \mathcal{L}$, then we say that the pair $(\gamma, X)$ is a failure of the process $E$. Formally we have that:*

$$failures(E) \stackrel{\text{def}}{=} \{(\gamma, X) \subseteq \mathcal{L}^* \times \mathbb{P}(\mathcal{L}) \mid \exists E' \text{ such that } E \stackrel{\gamma}{\Longrightarrow} E' \text{ and } E' \stackrel{X}{\not\Longrightarrow}\}$$

*When $failures(E) = failures(F)$ we write $E \approx_F F$ (failure equivalence).* ■

We identify a process $E$ with its failure set. So if $(\gamma, X) \in failures(E)$ we write $(\gamma, X) \in E$. Note that $\gamma \in traces(E)$ if and only if $(\gamma, \emptyset) \in E$. So $E \approx_F F$ implies $traces(E) = traces(F)$.

We also recall the definition of weak bisimulation [8]. In the following the expression $E \stackrel{\hat{\mu}}{\Longrightarrow} E'$ stands for $E \stackrel{\mu}{\Longrightarrow} E'$ if $\mu \in \mathcal{L}$, and for $E (\stackrel{\tau}{\to})^* E'$ if $\mu = \tau$ (note that $(\stackrel{\tau}{\to})^*$ means "zero or more $\tau$ labelled transitions" while $\stackrel{\tau}{\Longrightarrow}$ requires at least one $\tau$ labelled transition).

**Definition 2.3** *A relation $R \subseteq \mathcal{E} \times \mathcal{E}$ is a* weak bisimulation *if $(E, F) \in R$ implies, for all $\mu \in Act$,*

- *whenever $E \stackrel{\mu}{\longrightarrow} E'$ then there exists $F' \in \mathcal{E}$ such that $F \stackrel{\hat{\mu}}{\Longrightarrow} F'$ and $(E', F') \in R$;*
- *conversely, whenever $F \stackrel{\mu}{\longrightarrow} F'$ then there exists $E' \in \mathcal{E}$ such that $E \stackrel{\hat{\mu}}{\Longrightarrow} E'$ and $(E', F') \in R$.*

*Two SPA agents $E, F \in \mathcal{E}$ are observational equivalent, notation $E \approx_B F$, if there exists a weak bisimulation containing the pair $(E, F)$. Note that $\approx_B$ is an equivalence relation.* ■

It is easy to prove that $E \approx_B F$ implies $E \approx_F F$.

## 3 Lazy Security

In this Section we report the *lazy security* property [11] and we show that it can only deal with low-deterministic processes, i.e., processes which have a deterministic behaviour with respect to low level actions. Here we do not consider the *eager security* property (introduced in [11] to deal with output actions) since it supposes that high level actions happen instantaneously while in SPA, which has synchronous communications, both input and output actions can be delayed by users. We start with a formal definition of determinism.

**Definition 3.1** *$E$ is deterministic ($E \in Det$) if and only if whenever $\gamma a \in traces(E)$ then $(\gamma, \{a\}) \notin E$.* ■

[2] For notational convenience, we use sometimes the $\sum$ operator to represent a general n-ary (or even infinitary) sum operator.

So a process is deterministic if after every trace $\gamma$ it cannot both accept and refuse a certain action $a$. We give another characterization for determinism. A system $E$ is deterministic if and only if whenever it can move to two different processes $E'$ and $E''$ executing a certain trace $\gamma$, such processes are failure equivalent.

**Proposition 3.2** *$E \in Det$ if and only if for all $\gamma \in traces(E)$ we have that $E \stackrel{\gamma}{\Longrightarrow} E'$, $E \stackrel{\gamma}{\Longrightarrow} E''$ implies $E' \approx_F E''$.*

PROOF. ($\Rightarrow$) Let $E \in Det$, $E \stackrel{\gamma}{\Longrightarrow} E'$, $E \stackrel{\gamma}{\Longrightarrow} E''$ and $(\delta, K) \in E'$. We want to prove that $(\delta, K) \in E''$. Since $E \stackrel{\gamma}{\Longrightarrow} E'$, we have that $(\gamma\delta, K) \in E$. By $E \in Det$ we obtain that $\forall a \in K, \gamma\delta a$ is not a trace for $E$. We also have that $\delta$ is a trace for $E''$; in fact, if $E''$ can execute only a prefix of $\delta$, i.e. $E'' \stackrel{\alpha}{\Longrightarrow} E'''$ with $\delta = \alpha b \beta$, we have that $E$ can execute trace $\gamma\alpha b$ (through $E'$) and can refuse $b$ after $\gamma\alpha$ (through $E''$) contradicting the determinism hypothesis. Now, since $\forall a \in K, \gamma\delta a \notin traces(E)$, we also have that $\forall a \in K, \delta a \notin traces(E'')$ and so $(\delta, K) \in E''$.
($\Leftarrow$) Trivial. ■

**Corollary 3.3** *If $E \stackrel{\gamma}{\Longrightarrow} E'$ and $E \in Det$ then $E' \in Det$.*

PROOF. We have to prove that $E' \stackrel{\delta}{\Longrightarrow} E''$ and $E' \stackrel{\delta}{\Longrightarrow} E'''$ implies $E'' \approx_F E'''$. Consider $E \stackrel{\gamma\delta}{\Longrightarrow} E''$ and $E \stackrel{\gamma\delta}{\Longrightarrow} E'''$ then by $E \in Det$ we have that $E'' \approx_F E'''$. ■

In the following we will also use the $E|||F$ expression (interleaving without communication) as a shorthand for $(E[A/\mathcal{L}(E)] \mid F[B/\mathcal{L}(F)])[\mathcal{L}(E)/A, \mathcal{L}(F)/B]$ where $A, B \subseteq \mathcal{L}, A \cap B = \emptyset$. Moreover, $A/\mathcal{L}(E)$ is a bijective function which maps all the actions executable by $E$ (the actions in $\mathcal{L}(E)$) into actions in $A$. Finally, $\mathcal{L}(E)/A$ is the inverse of $A/\mathcal{L}(E)$ (the same holds for $B/\mathcal{L}(F)$ and $\mathcal{L}(F)/B$). This expression means that the actions in $E$ and $F$ are first relabelled using the two disjoint sets $A$ and $B$, then interleaved (no communication is possible) and finally renamed to their original labels.

We will also say that a process is *divergent* if it can execute an infinite sequence of internal actions $\tau$. As an example consider the agent $A \stackrel{\text{def}}{=} \tau.A + b.\underline{0}$ which can execute an arbitrary number of $\tau$ actions. We define *Nondiv* as the set of all the non-divergent processes.

We can now present the *lazy security* property [11]. This property implies that the obscuring of high level actions by interleaving does not introduce any non-determinism. The obscuring of high level actions of process $E$ by interleaving is obtained considering process $E|||RUN_H$ where $RUN_H \stackrel{\text{def}}{=} \sum_{h \in Act_H} h.RUN_H$. In such a process an outside observer is not able to tell if a certain high level action cames from $E$ or from $RUN_H$.

*L-Sec* also requires that $E|||RUN_H$ is non-divergent. [3] This is equivalent to requiring that $E$ is non-divergent, because $RUN_H$ is non-divergent and the $|||$ operator does not allow synchronizations (which could generate new $\tau$ actions).

**Definition 3.4** *$E \in$ L-Sec $\Leftrightarrow E|||RUN_H \in Det \cap Nondiv$.* ■

In the following we want to show that *L-Sec* can only analyze systems which are *low-deterministic*, i.e., where after any low level trace $\gamma$ no low level action $l$ can be both accepted and refused. The low-determinism requirement is not strictly necessary to avoid information flows from high to low level. So, in some cases, *L-Sec* is too strong. As an example consider the following non-deterministic system without high level actions: $E \stackrel{\text{def}}{=} l.l'.\underline{0} + l.l''.\underline{0}$. It is obviously secure but it is not low-deterministic and so it is not *L-Sec*. Formally we have that:

**Definition 3.5** *$E$ is low-deterministic ($E \in$ Lowdet) if and only if $E \setminus Act_H \in Det$.* ■

The following holds:

**Theorem 3.6** *L-Sec $\subseteq$ Lowdet.*

PROOF. Let $E \in$ *L-Sec*. Consider a trace $\gamma a$ of $E \setminus Act_H$ and suppose that $(\gamma, \{a\}) \in E \setminus Act_H$. So there exists $E'$ such that $E \setminus Act_H \stackrel{\gamma}{\Longrightarrow} E' \setminus Act_H$ and such that $E' \setminus Act_H \stackrel{a}{\not\Longrightarrow}$. Since $RUN_H$ cannot execute the low level action $a$ then we have that $E'|||RUN_H \stackrel{a}{\not\Longrightarrow}$ and so $(\gamma, \{a\}) \in E|||RUN_H$ because $E|||RUN_H \stackrel{\gamma}{\Longrightarrow} E'|||RUN_H$. Since $\gamma a$ is a trace for $E \setminus Act_H$ then it is also a trace for $E|||RUN_H$ and we obtain that $E|||RUN_H$ is not deterministic, contradicting the hypothesis. So $(\gamma, \{a\}) \notin E \setminus Act_H$ and $E \in$ *Lowdet*. ■

## 4 Bisimulation and Failure Non Deducibility on Compositions

In [4] we proposed a notion of information flow security: *Bisimulation Non Deducibility on Compositions*. A system $E$ is *BNDC* if for every high level process $\Pi$ a low level user cannot distinguish between processes $E$ and $(E|\Pi) \setminus Act_H$. 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 $\Pi$ to $E$.

**Definition 4.1** *$E \in$ BNDC if and only if $\forall \Pi \in \mathcal{E}_H$ we have $E/Act_H \approx_B (E \mid \Pi) \setminus Act_H$.* ■

[3] Note that in [11] the non-divergence requirement is inside the deterministic one. This is because the authors use the failure-divergence semantics [2]. In this work we use the failure equivalence which does not deal with divergences. So, in order to obtain exactly the *L-Sec* property, we require the non-divergence condition explicitly.

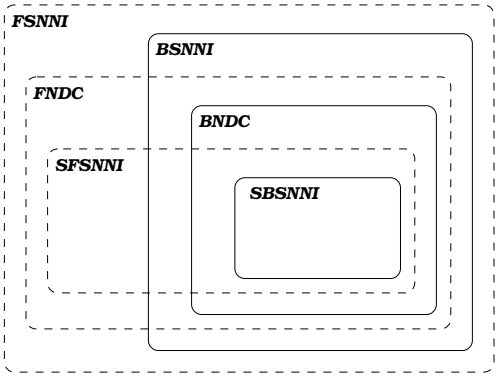


**Figure 1. Failure based and bisimulation based properties.**

A static characterization of *BNDC* – which does not involve composition with every processes $\Pi$ – is not immediate. As a matter of fact, this problem is still open. In [6] we proposed the *SBSNNI* property which is static, compositional (i.e., if two systems are *SBSNNI* their composition is *SBSNNI*) and strictly stronger than *BNDC*. We first define the *Bisimulation Strong Non-deterministic Non Interference* (*BSNNI*).

**Definition 4.2** $E \in BSNNI \Leftrightarrow E/Act_H \approx_B E \setminus Act_H$. ■

Now we can define the *Strong BSNNI*.

**Definition 4.3** *A system $E \in SBSNNI$ if and only if for all $E'$ such that $\exists \gamma : E \stackrel{\gamma}{\Longrightarrow} E'$ we have $E' \in BSNNI$.* ■

The following holds [4].

**Theorem 4.4** *SBSNNI $\subset$ BNDC $\subset$ BSNNI.* ■

Now we define the failure based security properties by simply substituting $\approx_B$ with $\approx_F$ in all the bisimulation based properties previously defined.

**Definition 4.5** *(Failure based properties)*

(*i*) $E \in FNDC \Leftrightarrow E/Act_H \approx_F (E \mid \Pi) \setminus Act_H$, *for all* $\Pi \in \mathcal{E}_H$;

(*ii*) $E \in FSNNI \Leftrightarrow E/Act_H \approx_F E \setminus Act_H$;

(*iii*) $E \in SFSNNI \Leftrightarrow \forall E'$ *such that* $\exists \gamma : E \stackrel{\gamma}{\Longrightarrow} E'$ *we have* $E' \in FSNNI$. ■

Since bisimulation equivalence is stronger than failure equivalence, it can be proved that each of these new property is weaker then its corresponding bisimulation based one. E.g. *BNDC* $\subset$ *FNDC*. Moreover we prove that the results of Theorem 4.4 can be extended also to these new properties.

**Theorem 4.6** *SFSNNI $\subset$ FNDC $\subset$ FSNNI.*

PROOF. (*SFSNNI $\subset$ FNDC*) Let $E$ be a *SFSNNI* process. We have to prove that $(E|\Pi) \setminus Act_H \approx_F E/Act_H$ for every high level process $\Pi$.
We first prove that $(\gamma, K) \in (E|\Pi)\setminus Act_H$ implies $(\gamma, K) \in E/Act_H$. Consider $(\gamma, K) \in (E|\Pi) \setminus Act_H$, then $\exists E', \Pi'$ such that $(E|\Pi) \setminus Act_H \stackrel{\gamma}{\Longrightarrow} (E'|\Pi') \setminus Act_H \stackrel{K}{\not\Rightarrow}$. Hence $E' \setminus Act_H \stackrel{K}{\not\Rightarrow}$ because $traces(E' \setminus Act_H) \subseteq traces((E'|\Pi') \setminus Act_H)$. Now, since $E \in SFSNNI$ then $E' \setminus Act_H \approx_F E'/Act_H$; hence $E'/Act_H \stackrel{K}{\not\Rightarrow}$. Note that $E/Act_H \stackrel{\gamma}{\Longrightarrow} E'/Act_H$, hence $(\gamma, K) \in E/Act_H$.
We now prove that $(\gamma, K) \in E/Act_H$ implies $(\gamma, K) \in (E|\Pi) \setminus Act_H$. Consider $(\gamma, K) \in E/Act_H$. By hypothesis we have that $(\gamma, K) \in E \setminus Act_H$ and so $\exists E'$ such that $E \setminus Act_H \stackrel{\gamma}{\Longrightarrow} E' \setminus Act_H \stackrel{K}{\not\Rightarrow}$. Since $E \in SFSNNI$ then $E'/Act_H \stackrel{K}{\not\Rightarrow}$. Hence we also have that $(E'|\Pi) \setminus Act_H \stackrel{K}{\not\Rightarrow}$ because $traces((E'|\Pi) \setminus Act_H) \subseteq traces(E'/Act_H)$. Since we have that $E \setminus Act_H \stackrel{\gamma}{\Longrightarrow} E' \setminus Act_H$ then $(E|\Pi) \setminus Act_H \stackrel{\gamma}{\Longrightarrow} (E'|\Pi) \setminus Act_H$ and so $(\gamma, K) \in (E|\Pi) \setminus Act_H$.
The inclusion is strict because agent $E \stackrel{\text{def}}{=} l.h.l.\underline{0} + l.\underline{0} + l.l.\underline{0}$ is *FNDC* but not *SFSNNI*.
(*FNDC $\subset$ FSNNI*) It is sufficient to consider $\Pi = \underline{0}$. We have that $(E|\underline{0}) \setminus Act_H \approx_F E \setminus Act_H$ and so, since $(E|\underline{0}) \setminus Act_H \approx_F E/Act_H$ we have $E/Act_H \approx_F E \setminus Act_H$.
The inclusion is strict because agent $E \stackrel{\text{def}}{=} l.h.l.h'.l.\underline{0} + l.\underline{0} + l.l.l.\underline{0}$ is *FSNNI* but not *FNDC*. ■

Figure 1 summarizes the inclusions between the presented security properties. It can be drawn using the previous inclusion results and the following remarks: *BNDC* $\not\subseteq$ *SFSNNI*, in fact agent $l.h.l.\underline{0} + l.\underline{0} + l.l.\underline{0}$ is *BNDC* but not *SFSNNI*; we also have that *BSNNI* $\not\subseteq$ *FNDC* because of agent $h.l.h'.l.\underline{0} + l.l.\underline{0}$; finally *SFSNNI* $\not\subseteq$ *BSNNI* because of agent $h.l.(l'.\underline{0} + l''.\underline{0}) + l.l'.\underline{0} + l.l''.\underline{0}$.

The next theorem shows that under the low-determinism assumption the properties *SFSNNI* and *FNDC* collapse into the same one. We need the following Lemma.

**Lemma 4.7** *If* $E, \tilde{E} \in Det$, $E \stackrel{\gamma}{\Longrightarrow} E'$, $\tilde{E} \stackrel{\gamma}{\Longrightarrow} \tilde{E}'$ *and* $E \approx_F \tilde{E}$ *then* $E' \approx_F \tilde{E}'$.

PROOF. We prove that if $(\delta, K) \in E'$ then $(\delta, K) \in \tilde{E}'$. Let $(\delta, K) \in E'$. Then $(\gamma\delta, K) \in E$ and by $E \approx_F \tilde{E}$ we obtain that $(\gamma\delta, K) \in \tilde{E}$. So $\exists \tilde{E}'', \tilde{E}'''$ such that $\tilde{E} \stackrel{\gamma}{\Longrightarrow} \tilde{E}'' \stackrel{\delta}{\Longrightarrow} \tilde{E}''' \stackrel{K}{\not\Rightarrow}$, hence $(\delta, K) \in \tilde{E}''$. Since $\tilde{E} \in Det$ then by Proposition 3.2 and hypothesis we have that $\tilde{E}'' \approx_F \tilde{E}'$ and so $(\delta, K) \in \tilde{E}'$. We can prove in the same way that if $(\delta, K) \in \tilde{E}'$ then $(\delta, K) \in E'$. So $E' \approx_F \tilde{E}'$ ■

**Theorem 4.8** *FNDC* $\cap$ *Lowdet* $\subseteq$ *SFSNNI.*

PROOF. Since *FNDC* $\subset$ *FSNNI* and $E \in$ *FNDC*, we have that $E \setminus Act_H \approx_F E/Act_H$. By $E \in$ *Lowdet* we obtain $E/Act_H \in Det$. Now consider $E \stackrel{\gamma}{\Longrightarrow} E'$. We have to prove that $E'/Act_H \approx_F E' \setminus Act_H$. Let $\Pi'$ be the high level process which executes exactly the complement of the high level projection of $\gamma$, i.e. the complement of the subsequence of $\gamma$ composed by all the high level actions in $\gamma$. If $\gamma'$ is the low level projection of $\gamma$ we have that $(E|\Pi') \setminus Act_H \stackrel{\gamma'}{\Longrightarrow} (E'|\underline{0}) \setminus Act_H \approx_F E' \setminus Act_H$. Since $E \stackrel{\gamma}{\Longrightarrow} E'$ then $E/Act_H \stackrel{\gamma'}{\Longrightarrow} E'/Act_H$. By hypothesis we have that $(E|\Pi') \setminus Act_H \approx_F E/Act_H$. Since $E/Act_H \in Det$ then, by Lemma 4.7, we have that $E'/Act_H \approx_F (E'|\underline{0}) \setminus Act_H \approx_F E' \setminus Act_H$. ■

**Corollary 4.9** *FNDC* $\cap$ *Lowdet* $=$ *SFSNNI* $\cap$ *Lowdet.*

PROOF. Trivial by Theorems 4.8 and 4.6. ■

## 5 Comparison

In this section we show that under the low-determinism and the non-divergence assumption the *BNDC* property is equal to *L-Sec*. We start proving this result for *FNDC*.

**Theorem 5.1** *L-Sec* $\subseteq$ *SFSNNI.*

PROOF. Let $E \in$ *L-Sec*. Then we have to prove that if $E \stackrel{\gamma}{\Longrightarrow} E'$ then $E' \setminus Act_H \approx_F E'/Act_H$. We first prove that if $(\delta, K) \in E'/Act_H$ then $(\delta, K) \in E' \setminus Act_H$. Consider $(\delta, K) \in E'/Act_H$. Then we have that $\exists E''$ such that $E'/Act_H \stackrel{\delta}{\Longrightarrow} E''/Act_H \stackrel{K}{\not\Rightarrow}$.

Now we want to prove that $\delta$ is a trace also for $E' \setminus Act_H$. Let $\delta = \delta_1\delta_2 \ldots \delta_n$ and consider the execution $E'/Act_H \stackrel{\delta_1}{\Longrightarrow} E'_1/Act_H \stackrel{\delta_2}{\Longrightarrow} \ldots \stackrel{\delta_n}{\Longrightarrow} E''/Act_H$. Suppose that $\delta_i$ is the first action in $\delta$ that $E' \setminus Act_H$ is not able to execute. In other words we have that

$$E' \setminus Act_H \stackrel{\delta_1}{\Longrightarrow} E'_1 \setminus Act_H \stackrel{\delta_2}{\Longrightarrow} \ldots \stackrel{\delta_{i-1}}{\Longrightarrow} E'_{i-1} \setminus Act_H \stackrel{\delta_i}{\not\Rightarrow}$$

This means that in order to execute $\delta_i$, process $E'_{i-1}/Act_H$ executes some hidden high level actions $h_1 \ldots h_k$. So $E'_{i-1} \stackrel{h_1 \ldots h_k \delta_i}{\Longrightarrow} E'_i$. If we execute such high level actions with $RUN_H$ we obtain that $E|||RUN_H \stackrel{\gamma\delta_1 \ldots \delta_{i-1} h_1 \ldots h_k}{\Longrightarrow} E'_{i-1}|||RUN_H \stackrel{\delta_i}{\not\Rightarrow}$. Since $E'_{i-1} \setminus Act_H$ and $\delta_i \in Act_L$ then we obtain that $(\gamma\delta_1 \ldots \delta_{i-1} h_1 \ldots h_k, \{\delta_i\}) \in E|||RUN_H$. Moreover, if we execute actions $h_1 \ldots h_k$ with $E'_{i-1}$ we have that $E|||RUN_H \stackrel{\gamma\delta_1 \ldots \delta_{i-1} h_1 \ldots h_k \delta_i}{\Longrightarrow} E'_i|||RUN_H$ and so $\gamma\delta_1 \ldots \delta_{i-1} h_1 \ldots h_k \delta_i$ is a trace for $E|||RUN_H$. This means that $E|||RUN_H \notin Det$ hence $E \notin$ *L-Sec*. We obtain a contradiction, so no $\delta_i$ can be refused by $E' \setminus Act_H$ and $\delta$ is a trace for such process. So we have that $E' \setminus Act_H \stackrel{\delta}{\Longrightarrow} E''' \setminus Act_H$.

Now we want to prove that $(\delta, K) \in E' \setminus Act_H$. Let $E' \setminus Act_H \stackrel{\delta}{\Longrightarrow} E''' \setminus Act_H$ and suppose that $E''' \setminus Act_H$ can execute a certain action $a \in K \cap Act_L$ (the actions in $K \cap Act_H$ cannot be executed by such process) then $\gamma\delta a$ is a trace for $E|||RUN_H$. Now consider the sequence $\delta'$ obtained by adding to $\delta$ all the high level action executed by $E'$ in order to reach $E''$ in the transition $E'/Act_H \stackrel{\delta}{\Longrightarrow} E''/Act_H$; i.e. $E' \stackrel{\delta'}{\Longrightarrow} E''$. Then we will have that $E'|||RUN_H \stackrel{\delta'}{\Longrightarrow} E''|||RUN_H$ and since $E''/Act_H \stackrel{K}{\not\Rightarrow}$ then $E''|||RUN_H \stackrel{a}{\not\Rightarrow}$ and so $(\gamma\delta', \{a\}) \in E|||RUN_H$. Now if $\gamma\delta a$ is a trace for $E|||RUN_H$ then also $\gamma\delta' a$ is, and so, again, we obtain that $E|||RUN_H \notin Det$ and $E \notin$ *L-Sec*. Hence $E''' \setminus Act_H \stackrel{a}{\not\Rightarrow}$ for every $a \in K$ and so $(\delta, K) \in E' \setminus Act_H$.

Now we prove that if $(\delta, K) \in E' \setminus Act_H$ then $(\delta, K) \in E'/Act_H$. Suppose $(\delta, K) \in E' \setminus Act_H$. Then we have that $\exists E''$ such that $E' \setminus Act_H \stackrel{\delta}{\Longrightarrow} E'' \setminus Act_H \stackrel{K}{\not\Rightarrow}$. Hence also $E'/Act_H \stackrel{\delta}{\Longrightarrow} E''/Act_H$. Suppose that $E''/Act_H$ can execute a certain $a \in K \cap Act_L$ then consider $\delta'$ obtained by adding to $\delta$ all the high level actions executed by $E'$ before $a$ in the transition $E'/Act_H \stackrel{\delta}{\Longrightarrow} E''/Act_H \stackrel{a}{\Longrightarrow} E'''/Act_H$, i.e., such that $\delta' a$ is a trace for $E'$. We have that $\gamma\delta' a$ is a trace for $E|||RUN_H$. Now, $(\delta, \{a\}) \in E' \setminus Act_H$ with $a \in Act_L$ and so $(\delta, \{a\}) \in E'|||RUN_H$ which implies that $(\delta', \{a\}) \in E'|||RUN_H$ and finally $(\gamma\delta', \{a\}) \in E|||RUN_H$. This contradict the fact that $E \in$ *L-Sec* and so $E''/Act_H \stackrel{a}{\not\Rightarrow}, \forall a \in K$. Hence $(\delta, K) \in E'/Act_H$. ■

**Theorem 5.2** *SFSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* $\subseteq$ *L-Sec.*

PROOF. Let $E \in$ *SFSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* and $\gamma a$ be a trace for process $E|||RUN_H$. We want to prove that $(\gamma, \{a\}) \notin E|||RUN_H$. It trivially holds if $a \in Act_H$ because in such a case it can always be executed by $RUN_H$. So let $a \in Act_L$. Suppose $E|||RUN_H \stackrel{\gamma}{\Longrightarrow} E'|||RUN_H \stackrel{a}{\not\Rightarrow}$ and consider the sequence $\gamma'$ obtained removing all the high level actions from $\gamma$. Then $E/Act_H \stackrel{\gamma'}{\Longrightarrow} E'/Act_H$

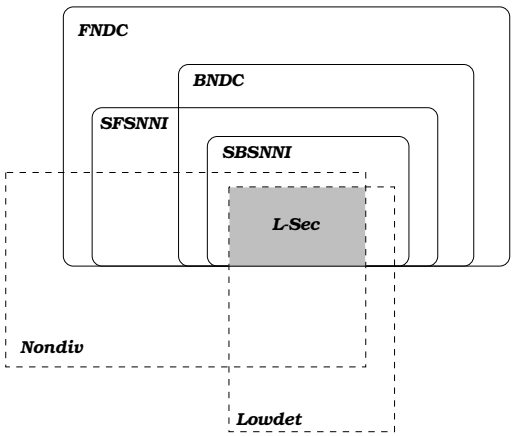


**Figure 2. Relations between properties.**

and by hypothesis $E'/Act_H \approx_F E' \setminus Act_H$. Since $E'|||RUN_H \stackrel{a}{\not\Rightarrow}$ then $E' \setminus Act_H \stackrel{a}{\not\Rightarrow}$ and so $E'/Act_H \stackrel{a}{\not\Rightarrow}$ and $(\gamma', \{a\}) \in E/Act_H$. Since $E \in SFSNNI$ we obtain that $(\gamma', \{a\}) \in E \setminus Act_H$. Now $\gamma a$ is a trace for $E|||RUN_H$ and so $\gamma' a$ must be a trace for $E/Act_H$ this means that $\gamma' a$ is also a trace for $E \setminus Act_H$. Since $E \in Lowdet$ then $E \setminus Act_H$ is deterministic. However we found that $\gamma' a$ is a trace for $E \setminus Act_H$ and $(\gamma', \{a\}) \in E \setminus Act_H$ obtaining a contradiction. So $E'|||RUN_H$ cannot refuse $a$ and $(\gamma, \{a\}) \notin E|||RUN_H$. Hence $E|||RUN_H \in Det$ and since $E \in Nondiv$ we also have that $E|||RUN_H \in Nondiv$ ■

**Corollary 5.3** *SFSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* = *L-Sec.*

PROOF. By Theorems 3.6 and 5.1 and by Definition 3.4 we find that L-Sec $\subseteq$ *SFSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv.* Finally by Theorem 5.2 we obtain the thesis. ■

Note that by Corollary 4.9 we also have that *FNDC* $\cap$ *Lowdet* $\cap$ *Nondiv* = *L-Sec*. Now we show that this result also hold for *SBSNNI* and *BNDC.* We first prove that for deterministic processes $\approx_F$ becomes equal to $\approx_B$.

**Proposition 5.4** $E \in Det, E \approx_F F \Longrightarrow E \approx_B F$.

PROOF. If $E \in Det$ and $E \approx_F F$ we also have that $F \in Det$. Now it is sufficient to consider the relation $R \subseteq \mathcal{E} \times \mathcal{E}$ defined as follows: $(E', E'') \in R$ if and only if $\exists \gamma : E \stackrel{\gamma}{\Longrightarrow} E'$, $E \stackrel{\gamma}{\Longrightarrow} E''$. It is easy to show that $R$ is a weak bisimulation. ■

Finally, the following holds.

**Theorem 5.5** *BNDC* $\cap$ *Lowdet* $\cap$ *Nondiv* = *SBSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* = *L-Sec.*

PROOF. (*SBSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* = *L-Sec*). We have that *SBSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* $\subseteq$ *SFSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* because *SBSNNI* $\subset$ *SFSNNI*. So by Theorem 5.2 *SBSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* $\subseteq$ *L-Sec*.

Now we prove that *L-Sec* $\subseteq$ *SBSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv.* If $E \in$ *L-Sec* then by Corollary 5.3 we have that $E \in$ *SFSNNI*$\cap$*Lowdet*$\cap$*Nondiv.* So $\forall E'$ such that $\exists \gamma : E \stackrel{\gamma}{\Longrightarrow} E'$ we have $E' \setminus Act_H \approx_F E'/Act_H$ with $E \setminus Act_H \in Det$. In particular we also have that $E \setminus Act_H \approx_F E/Act_H$ and since $E \setminus Act_H \in Det$, we obtain that $E/Act_H \in Det$. Note that $E/Act_H \stackrel{\gamma'}{\Longrightarrow} E'/Act_H$ where $\gamma'$ is the sequence obtained removing all the high level actions from $\gamma$. Hence, by Corollary 3.3, $E'/Act_H \in Det$. Finally, by Proposition 5.4 we obtain that $E' \setminus Act_H \approx_B E'/Act_H$.
(*BNDC* $\cap$ *Lowdet* $\cap$ *Nondiv* = *SBSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv*)
Trivial by *SBSNNI* $\subset$ *BNDC* $\subset$ *FNDC* and since *SBSNNI* $\cap$ *Lowdet* $\cap$ *Nondiv* = *L-Sec* = *FNDC* $\cap$ *Lowdet* $\cap$ *Nondiv.* ■

Figure 2 summarizes the relations between various properties and conditions.

Consider the following agent: $E \stackrel{\text{def}}{=} l.l'.\underline{0} + l.l''.\underline{0} + h.(l.l'.\underline{0} + l.l''.\underline{0})$. It is *SBSNNI* but not *L-Sec* because it is not *Lowdet*. In [10] systems like this are considered not secure because they have a not secure refinement. As an example for $E$ we have the refinement $E' \stackrel{\text{def}}{=} l.l'.\underline{0} + h.l.l''.\underline{0}$ which is clearly not secure.

## 6 Conclusion

We have shown that *BNDC* and *SBSNNI* are equal to *L-Sec* when dealing with low-deterministic and non-divergent processes. In [6, 5] we introduced the Security Checker (SC), a tool based on Concurrency Workbench [3], which is able to automatically check the *SBSNNI* property over finite state agents. This implies that for low-deterministic, non-divergent and finite-state processes it is possible to use the SC in order to verify the *L-Sec* property. Moreover, SC offers an automatic compositional checking (see [5] for more details) which reduces the exponential state explosion

due to parallel composition operator by exploiting the compositionality of security properties. A security property is compositional if it is closed with respect to $|$ and $\setminus$ operators. The basic idea of the compositional verification is the following: if we have to check if agent $(E|F) \setminus L$ is secure we simply check the security of $E$ and $F$. If it is satisfied then we conclude that $(E|F) \setminus L$ is secure, otherwise we check the security of the whole agent. Note that this strategy can be used to check *SBSNNI*, since in [6] it has been proved that *SBSNNI* is compositional.

In [11] it is shown how to use the FDR tool [9] to check the *L-Sec* property. Note that it would be interesting to compare the performance of FDR and SC for the verification of such a property.

We also want to point out that *SBSNNI* $\cap$ *Lowdet* can extend in a *fair* manner the *L-Sec* property to divergent processes. *L-Sec* assumes that processes cannot diverge. The semantics used by authors to define *L-Sec* is the failure-divergence one [2]. Failure-divergence semantics gives a *catastrophic* interpretation of divergences, since in the presence of divergences a process may show any behaviour. For example, consider agents $A$ and $C$ defined as follows: $A \stackrel{\text{def}}{=} a.B$ with $B \stackrel{\text{def}}{=} \tau.B + b.\underline{0}$ and $C \stackrel{\text{def}}{=} a.D$ with $D \stackrel{\text{def}}{=} \tau.D + d.\underline{0}$. They are failure-divergence equivalent, but they are not trace equivalent, in fact $A$ can only execute $a$ and $ab$ while $C$ can only execute $a$ and $ad$. Technically, this is obtained by inserting a completely non deterministic behaviour every time we have a divergence. On the other hand, weak bisimulation gives a fair interpretation of divergences. As an example the agents $A$ and $C$ are not weak bisimulation equivalent. Moreover consider agent $A' \stackrel{\text{def}}{=} a.b.\underline{0}$. We have that $A$ and $A'$ are weak bisimulation equivalent but they are not failure-divergence equivalent because of the divergence in agent $A$. The basic idea is that the $\tau$-loop in $B$ is executed an arbitrary but finite number of times. So in $A$ action $b$ will eventually be enabled, and this makes $A$ equivalent to $A'$ and not equivalent to $C$. This is useful, for example, if we want to model a fair communication media, where a $\tau$-loop represents the unbounded but finite losses of messages. So the property *SBSNNI* $\cap$ *Lowdet* can be seen as an extension of *L-Sec* which gives a fair interpretation of divergences.

### Acknowledgements

We would like to thank the anonymous referees for helpful comments and suggestions.

## References

[1] S. D. Brookes, C. A. R. Hoare, and A. W. Roscoe. "A Theory of Communicating Sequential Processes". *Journal of the Association for Computing Machinery*, 31(3):560–599, July 1984.

[2] S. D. Brookes and A. W. Roscoe. "An Improved Failures Model for Communicating Processes". In *Proceedings of the Pittsburgh seminar on concurrency*, pages 281–305. Springer-Verlag, LNCS 197, 1985.

[3] R. Cleaveland, J. Parrow, and B. Steffen. "The Concurrency Workbench: a Semantics Based Tool for the Verification of Concurrent Systems". *ACM Transactions on Programming Languages and Systems*, Vol. 15 No. 1:36–72, Jan. 1993.

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

[5] R. Focardi and R. Gorrieri. "Automatic Compositional Verification of Some Security Properties". In *Proceedings of Second International Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*, pages 167–186, Passau (Germany), March 1996. Springer-Verlag, LNCS 1055.

[6] R. Focardi, R. Gorrieri, and V. Panini. "The Security Checker: a Semantics-based Tool for the Verification of Security Properties". In *Proceedings Eight IEEE Computer Security Foundation Workshop, (CSFW'95) (Li Gong Ed.)*, pages 60–69, Kenmare (Ireland), June 1995. IEEE Press.

[7] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

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

[9] A. W. Roscoe. "Model Checking CSP". In A. W. Roscoe (ed) *A Classical Mind*. Prentice Hall, 1994.

[10] A. W. Roscoe. "CSP and Determinism in Security Modelling". In *Proceedings, 1995 IEEE Symposium on Security and Privacy*, pages 114–127. IEEE Computer Society Press, 1995.

[11] A. W. Roscoe, J. C. P. Woodcock, and L. Wulf. "Noninterference through Determinism". In *Proceeding of European Symposium on Research in Computer Security 1994 (ESORICS'94)*, pages 33–53. Springer-Verlag LNCS 875, 1994.