A General Framework for Nondeterministic, Probabilistic, and Stochastic Noninterference

Alessandro Aldini and Marco Bernardo

University of Urbino "Carlo Bo" – Italy
Information Science and Technology Institute

Abstract. We introduce a notion of stochastic noninterference aimed at extending the classical approach to information flow analysis with fine-grain information describing the temporal behavior of systems. In particular, we refer to a process algebraic setting that joins durational activities expressing time passing through exponentially distributed random variables, zero duration activities allowing for prioritized/probabilistic choices, and untimed activities with unspecified duration. In this setting unifying time, priority, probability, and nondeterminism, we highlight the expressive power of stochastic noninterference with respect to the existing definitions of nondeterministic and probabilistic noninterference. From this comparison, we obtain that stochastic noninterference turns out to be very strict and limiting in real-world applications and, therefore, requires the use of relaxation techniques. Among them we advocate performance evaluation as a means for achieving a reasonable balance between security requirements and quality.

1 Fine-grain Models and Noninterference

Information flow analysis is a basic approach to the verification of security properties of systems which, in general, require to control who has access to what and when. Among the several conditions that describe the characteristics of unauthorized information flows, called covert channels, one of the most interesting, for its intuitive and wide-used idea, is the noninterference requirement [11]. Very briefly, in a multi-level secure system, the user at the high security level should not be able to affect what the user at the low security level can observe. Independently of the specific formalization of this notion, the underlying approach is based on the idea that checking noninterference is actually checking the indistinguishability of the different low-level views of the system that are obtained by changing the high-level behavior, see e.g. [22, 9, 17] and the references therein.

Along the 90s generalized notions of noninterference were designed to analyze deterministic and nondeterministic systems. However, it was immediately clear that moving to a quantitative framework including fine-grain information, such as probability distributions associated with event execution, augments the distinguishing power of the observer [14, 18, 19, 3]. More recently, the awareness that perfect noninterference is difficult to achieve has urged to estimate the information flow by employing this quantitative information and a relaxed notion of

noninterference, see e.g. [4] and the references therein. A quantitative notion of noninterference can be based not only on probabilistic information, but also on temporal information. In particular, in this paper we refer to the representation of time passing that uses non-negative random variables, which is particularly appropriate when the time taken by an event fluctuates according to some probability distribution. Among the many probability distributions that can be used to model event timing, we concentrate on exponential distributions, which are equipped with a widely developed theory.

As known in the probabilistic setting, the more information is added to the system, the higher the number of potential vulnerabilities revealed through fine-grain notions of noninterference. Hence, a noninterference property taking into account the timing of events, described through exponentially distributed random variables, would be sensitive to stochastically timed covert channels. In the literature, some proposals concerned with discrete-time extensions of noninterference have been proposed, e.g. in the setting of real-time process algebra [8] and probabilistic timed automata [12, 7], while other time-based aspects of noninterference have been investigated in the general setting of Shannon information theory, see e.g. [13]. However, to the best of our knowledge, no formal definition of noninterference has been proposed in the case of stochastic time.

This paper faces the problem of defining noninterference in the setting of a stochastic process algebra encompassing nondeterminism, probability, priority, and stochastic time. The semantics of this paradigm is defined in terms of a fine-grain notion of bisimulation that extends the classical bisimulation relation of Milner [15]. In this general framework, it is possible to formalize not only stochastic noninterference properties, but also to evaluate the expressive power that is gained when adding fine-grain information in an incremental way. The result we obtain is that when moving from nondeterministic noninterference to fine-grain definitions of noninterference, security constraints become severe and hard to accomplish in practice. In the most restrictive scenario where temporal behavior and stochastic noninterference come into play, we will see that these constraints may be impractical, so that in real-world applications we need relaxation techniques aiming at trading the amount of (unavoidable) information leakage with the complexity of the adopted securing countermeasures.

We will discuss the expressive power of the different notions of noninterference through a client-server example with two access clearance levels, high and low. The clients compete for the resource on the basis of their access clearance (assume that high-level clients take precedence over low-level clients). The service center is composed of two buffers of capacity 1 – one for each security level – and one shared server providing a service to the clients of both levels. This toy example is more than enough to highlight the severeness of the constraints to which a completely secure system should be subjected.

The paper is organized as follows. In Sect. 2 we describe the stochastic process algebra framework, on the base of which in Sect. 3 we introduce the stochastic version of noninterference together with a comparison with the nondeterministic/probabilistic setting. Some conclusions are drawn in Sect. 4.

2 Stochastic Process Algebra Framework

The general framework for the analysis of fine-grain noninterference we use is based on a Markovian process calculus extended with prioritized/weighted immediate actions, in which multiway communication is based on a mixture of the generative and reactive models of [10]. All these features, which are equipped with a widely developed theory, provide the expressiveness needed to model fine-grain details of real systems.

In the following, we explain the syntax and semantics of this calculus, how to derive nondeterministic and probabilistic sub-calculi, and the bisimulation semantics.

2.1 Markovian Process Calculus

The basic elements of the calculus are durational actions. They are represented as pairs of the form $<a, \tilde{\lambda}>$, where a is the action name and $\tilde{\lambda}$ is the action rate. There are three kinds of actions: exponentially timed, immediate, and passive.

Exponentially timed actions are of the form $<a, \lambda>$ with $\lambda \in \mathbb{R}_{>0}$. The duration of each such action is exponentially distributed with parameter equal to the action rate (hence its average duration is the inverse of its rate). Whenever several exponentially timed actions are enabled, the race policy is adopted, hence the action that is executed is the fastest one. As a consequence, the execution probability of an exponentially timed action is proportional to its rate.

Immediate actions are of the form $<a, \infty_{l,w}>$, where $l \in \mathbb{N}_{>0}$ is the priority level and $w \in \mathbb{R}_{>0}$ is the weight. Each immediate action has duration zero and takes precedence over exponentially timed actions, which are assumed to have priority level 0. Whenever several immediate actions are enabled, the generative preselection policy is adopted. This means that the lower priority immediate actions are discarded, whereas each of the highest priority immediate actions is given an execution probability equal to the action weight divided by the sum of the weights of all the highest priority immediate actions.

Passive actions are of the form $<a, *_w^{l'}>$, where $l' \in \mathbb{N}$ is the priority constraint and $w \in \mathbb{R}_{>0}$ is the weight. Each passive action with priority constraint l' can synchronize only with another passive action having the same name and the same priority constraint, or with a non-passive action having the same name and priority level l such that $l = l'$. Whenever several passive actions are enabled, the reactive preselection policy is adopted. This means that, within every set of enabled passive actions having the same name, each such action is given an execution probability equal to the action weight divided by the sum of the weights of all the actions in the set. Instead, the choice among passive actions having different names is nondeterministic. Likewise, the choice between a passive action and a non-passive action is nondeterministic.

Definition 1. *Let Act = Name × Rate be a set of actions, with Name being a set of action names containing a distinguished symbol τ for the invisible action and Rate* $= \mathbb{R}_{>0} \cup \{\infty_{l,w} \mid l \in \mathbb{N}_{>0} \wedge w \in \mathbb{R}_{>0}\} \cup \{*_w^{l'} \mid l' \in \mathbb{N} \wedge w \in \mathbb{R}_{>0}\}$ *being*

a set of action rates (ranged over by $\tilde{\lambda}$). The set $\mathcal{L}$ of process terms is generated by the following syntax:

$$P ::= \underline{0} \mid <a, \tilde{\lambda}>.P \mid P + P \mid A \mid P/L \mid P \parallel_S P$$

where $L, S \subseteq Name - \{\tau\}$ and A is a process constant defined through the (possibly recursive) equation $A \stackrel{\Delta}{=} P$.

The semantics for the set $\mathcal{P}$ of closed and guarded process terms of $\mathcal{L}$ is defined in the usual operational style. The behavior of each term P is given by a labeled multitransition system $[\![P]\!]$, whose states correspond to process terms and whose transitions – each of which has a multiplicity equal to the number of proofs of its derivation – are labeled with actions.

The null term $\underline{0}$ cannot execute any action, hence the corresponding semantics is given by a state with no transitions. The action prefix term $<a, \tilde{\lambda}>.P$ can execute an action with name a and rate $\tilde{\lambda}$ and then behaves as P:

$$<a, \lambda>.P \stackrel{a,\lambda}{\longrightarrow} P \qquad <a, \infty_{l,w}>.P \stackrel{a,\infty_{l,w}}{\longrightarrow} P \qquad <a, *^{l'}_{w}>.P \stackrel{a,*^{l'}_{w}}{\longrightarrow} P$$

The alternative composition $P_1 + P_2$ behaves as either P_1 or P_2 depending on whether P_1 or P_2 executes an action first:

$$\frac{P_1 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P'}{P_1 + P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P'} \qquad \frac{P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P'}{P_1 + P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P'}$$

The process constant A behaves as the right-hand side process term in its defining equation:

$$\frac{P \stackrel{a,\tilde{\lambda}}{\longrightarrow} P' \qquad A \stackrel{\Delta}{=} P}{A \stackrel{a,\tilde{\lambda}}{\longrightarrow} P'}$$

The hiding term P/L behaves as P with the difference that the name of every action executed by P that belongs to L is turned into τ:

$$\frac{P \stackrel{a,\tilde{\lambda}}{\longrightarrow} P' \qquad a \in L}{P/L \stackrel{\tau,\tilde{\lambda}}{\longrightarrow} P'/L} \qquad \frac{P \stackrel{a,\tilde{\lambda}}{\longrightarrow} P' \qquad a \notin L}{P/L \stackrel{a,\tilde{\lambda}}{\longrightarrow} P'/L}$$

The parallel composition $P_1 \parallel_S P_2$ behaves as P_1 in parallel with P_2 as long as actions are executed whose name does not belong to S:

$$\frac{P_1 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_1' \qquad a \notin S}{P_1 \parallel_S P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_1' \parallel_S P_2} \qquad \frac{P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_2' \qquad a \notin S}{P_1 \parallel_S P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_1 \parallel_S P_2'}$$

Generative-reactive synchronizations are forced between any non-passive action executed by one term and any passive action executed by the other term that

have the same name belonging to S and the same priority level/constraint:

$$\frac{P_1 \xrightarrow{a,\lambda} P_1' \quad P_2 \xrightarrow{a,*^0_w} P_2' \quad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\lambda \cdot \frac{w}{weight(P_2,a,0)}} P_1' \parallel_S P_2'} \qquad \frac{P_1 \xrightarrow{a,*^0_w} P_1' \quad P_2 \xrightarrow{a,\lambda} P_2' \quad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\lambda \cdot \frac{w}{weight(P_1,a,0)}} P_1' \parallel_S P_2'}$$

$$\frac{P_1 \xrightarrow{a,\infty_{l,v}} P_1' \quad P_2 \xrightarrow{a,*^l_w} P_2' \quad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\infty_{l,v} \cdot \frac{w}{weight(P_2,a,l)}} P_1' \parallel_S P_2'} \qquad \frac{P_1 \xrightarrow{a,*^l_w} P_1' \quad P_2 \xrightarrow{a,\infty_{l,v}} P_2' \quad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\infty_{l,v} \cdot \frac{w}{weight(P_1,a,l)}} P_1' \parallel_S P_2'}$$

where $weight(P,a,l) = \sum \{\!|\, w \mid \exists P' \in \mathcal{P}.\, P \xrightarrow{a,*^l_w} P' \,|\!\}$. Following [6], the adopted synchronization discipline mixes generative and reactive probabilistic models. Among all the enabled timed actions whose names belong to a synchronization set, the proposal of an action name is generated after a selection based on the rates of those actions. Then the enabled passive actions that have the same name as the proposed one react by means of a selection based on their weights. Finally, the timed action winning the generative selection and the passive action winning the reactive selection synchronize with each other.

Reactive-reactive synchronizations are forced between any two passive actions of the two terms that have the same name belonging to S and the same priority constraint:

$$\frac{P_1 \xrightarrow{a,*^l_{w_1}} P_1' \quad P_2 \xrightarrow{a,*^l_{w_2}} P_2' \quad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,*^l_{\frac{w_1}{weight(P_1,a,l)} \cdot \frac{w_2}{weight(P_2,a,l)} \cdot (weight(P_1,a,l) + weight(P_2,a,l))}} P_1' \parallel_S P_2'}$$

We will use the abbreviation $P \backslash S$ to stand for $P \parallel_S \underline{0}$, which intuitively describes the behavior of a restriction operator. Moreover, if $P \xrightarrow{a_1,\tilde{\lambda}_1} \ldots \xrightarrow{a_n,\tilde{\lambda}_n} P'$, with $n \in \mathbb{N}$, then we say that P' is a derivative of P and we denote with $Der(P)$ the set of derivatives of P. We denote with $\mathcal{P}_{\mathrm{pc}}$ the set of performance closed process terms of $\mathcal{P}$, i.e. those terms with no passive transitions. The stochastic process underlying $P \in \mathcal{P}_{\mathrm{pc}}$ is a continuous-time Markov chain (CTMC) possibly extended with immediate transitions. States having outgoing immediate transitions are called vanishing as the sojourn time in these states is zero. In order to retrieve a pure CTMC stochastically equivalent to an extended CTMC, all the vanishing states can be adequately eliminated.

Example 1. Let us exemplify the use of the operators through the running example modeling the client-server system. The arrival process for clients of level $k \in \{L, H\}$, where L stands for low access level and H stands for high access level, can be modeled as:

$$A_L \stackrel{\Delta}{=} <a_L, \lambda_L>.A_L \quad \text{and} \quad A_H \stackrel{\Delta}{=} <a_H, \lambda_H>.A_H$$

where λ_k is the parameter of the Poisson arrival process for clients of access level

k. The buffer for clients of access level k – which we assume to have capacity 1 – can be modeled as a completely passive entity as follows:

$$B_k \stackrel{\Delta}{=} <a_k, *^0_w>.<d_k, *^{f(k)}_w>.B_k$$

where, for instance, we assume $f(L) = 1$ and $f(H) = 2$ to denote that high-level requests pre-empt low-level ones. The shared server can be modeled as follows:

$$S \stackrel{\Delta}{=} \sum_{k \in \{L,H\}} <d_k, \infty_{f(k),w}>.<s_k, \mu_k>.S$$

where μ_k is the service rate for clients of access level k. Hence, the overall client-server system can be modeled as follows:

$$CS \stackrel{\Delta}{=} (A_L \parallel_\emptyset A_H) \parallel_{\{a_L,a_H\}} (B_L \parallel_\emptyset B_H) \parallel_{\{d_L,d_H\}} S$$

■

2.2 Nondeterministic and Probabilistic Sub-calculi

From the calculus above it is straightforward to derive two sub-calculi: a nondeterministic one and a probabilistic one. The former is obtained by removing all the fine-grain information related to time, probabilities, and priorities. Syntactically, durational actions of the form $<a, \lambda>$ and $<a, \infty_{l,w}>$ are replaced by the output action a, while passive actions of the form $<a, *^l_w>$ become the input action a_*. Semantically, the result is a classical calculus with CSP-like communication, whose underlying model is a labeled transition system. We denote with $nd(P)$ the nondeterministic version of process P obtained in this way.

The latter sub-calculus is obtained by removing all the fine-grain information related to time and priorities. Syntactically, the exponentially timed action $<a, \lambda>$ is mapped to the immediate action $<a, \infty_{0,\lambda}>$. In this way, the action rate is interpreted as the action weight that governs the execution probability of the action. The immediate action $<a, \infty_{l,w}>$ is simply mapped to $<a, \infty_{0,w}>$, thus losing the information about its priority. Similarly, the passive action $<a, *^l_w>$ is turned into the action $<a, *^0_w>$. Semantically, the result is a calculus where the execution probabilities of the actions are calculated on the basis of the associated weights. The underlying model for this calculus is a mixture of the generative and reactive probabilistic transition systems. A variant of this calculus, where the probabilities are associated with the operators rather than attached to the actions as weights, has been introduced in [6] and used in [3] to define probabilistic noninterference. We denote with $pr(P)$ the probabilistic version of process P obtained in this way.

2.3 Bisimulation Semantics

We now recall from [5] an extension of Markovian bisimilarity, whose basic idea is to compare the exit rates of the process terms by taking into account the various kinds of actions (exponentially timed, immediate, and passive). This is accomplished by parameterizing the notion of exit rate with respect to a number in $\mathbb{Z}$ representing the priority level of the action, which is 0 if the action is exponentially timed, l if the action rate is $\infty_{l,w}$, $-l' - 1$ if the action rate is $*^{l'}_w$.

Definition 2. *Let $P \in \mathcal{P}$, $a \in Name$, $l \in \mathbb{Z}$, and $C \subseteq \mathcal{P}$. The exit rate of P when executing actions with name a and priority level l that lead to C is defined through the following non-negative real function:*

$$rate(P,a,l,C) = \begin{cases} \sum\{\!|\, \lambda \mid \exists P' \in C.\, P \xrightarrow{a,\lambda} P' \,|\!\} & \text{if } l = 0 \\ \sum\{\!|\, w \mid \exists P' \in C.\, P \xrightarrow{a,\infty_{l,w}} P' \,|\!\} & \text{if } l > 0 \\ \sum\{\!|\, w \mid \exists P' \in C.\, P \xrightarrow{a,*_w^{-l-1}} P' \,|\!\} & \text{if } l < 0 \end{cases}$$

where each sum is taken to be zero whenever its multiset is empty.

Extended Markovian bisimilarity compares the process term exit rates for all possible action names and priority levels, except for those actions that will always be pre-empted by higher priority actions of the form $<\tau,\infty_{l,w}>$. We denote by $pri^{\tau}_{\infty}(P)$ the priority level of the highest priority internal immediate action enabled by P, and we set $pri^{\tau}_{\infty}(P) = 0$ if P does not enable any internal immediate action. Moreover, given $l \in \mathbb{Z}$, we use $no\text{-}pre(l,P)$ to denote that no action of level l can be pre-empted in P. Formally, this is the case whenever $l \geq pri^{\tau}_{\infty}(P)$ or $-l-1 \geq pri^{\tau}_{\infty}(P)$.

Definition 3. *An equivalence relation $\mathcal{B} \subseteq \mathcal{P} \times \mathcal{P}$ is an extended Markovian bisimulation iff, whenever $(P_1,P_2) \in \mathcal{B}$, then for all action names $a \in Name$, equivalence classes $C \in \mathcal{P}/\mathcal{B}$, and priority levels $l \in \mathbb{Z}$ such that $no\text{-}pre(l,P_1)$ and $no\text{-}pre(l,P_2)$:*

$$rate(P_1,a,l,C) = rate(P_2,a,l,C)$$

Extended Markovian bisimilarity, denoted by $\sim_{\mathrm{EMB}}$, is the union of all the extended Markovian bisimulations.

We relax the definition of exit rate by means of a notion of reachability that involves the internal actions with zero duration $<\tau,\infty_{l,w}>$, which are unobservable. The idea is that, if a given class of process terms is not reached directly after executing an action with a certain name and level, then we have to explore the possibility of reaching that class indirectly via a finite-length unobservable path starting from the term reached after executing the considered action.

Definition 4. *Let $P \in \mathcal{P}$ and $l \in \mathbb{N}_{>0}$. We say that P is l-unobservable iff $pri^{\tau}_{\infty}(P) = l$ and P does not enable any immediate non-τ-action with priority level $l' \geq l$, nor any passive action with priority constraint $l' \geq l$.*

Definition 5. *Let $n \in \mathbb{N}_{>0}$ and $P_1, P_2, \ldots, P_{n+1} \in \mathcal{P}$. A path π of length n:*

$$P_1 \xrightarrow{\tau,\infty_{l_1,w_1}} P_2 \xrightarrow{\tau,\infty_{l_2,w_2}} \ldots \xrightarrow{\tau,\infty_{l_n,w_n}} P_{n+1}$$

is unobservable iff for all $i = 1,\ldots,n$ process term P_i is l_i-unobservable. In that case, the probability of executing path π is given by:

$$prob(\pi) = \prod_{i=1}^{n} \frac{w_i}{rate(P_i,\tau,l_i,\mathcal{P})}$$

Definition 6. *Let $P \in \mathcal{P}$, $a \in Name$, $l \in \mathbb{Z}$, and $C \subseteq \mathcal{P}$. The weak exit rate at which P executes actions of name a and level l that lead to C is defined through the following non-negative real function:*

$$rate_{\rm w}(P,a,l,C) = \sum_{P' \in C_{\rm w}} rate(P,a,l,\{P'\}) \cdot prob_{\rm w}(P',C)$$

where $C_{\rm w}$ *is the weak backward closure of* C*:*

$$C_{\rm w} = C \cup \{Q \in \mathcal{P} - C \mid Q \text{ can reach } C \text{ via unobservable paths}\}$$

and $prob_{\rm w}$ *is a* $\mathbb{R}_{]0,1]}$*-valued function representing the sum of the probabilities of all the unobservable paths from a term in* $C_{\rm w}$ *to* C*:*

$$prob_{\rm w}(P',C) = \begin{cases} 1 & \text{if } P' \in C \\ \sum\{\!|\, prob(\pi) \mid \pi \text{ unobs. path from } P' \text{ to } C \,|\!\} & \text{if } P' \in C_{\rm w} - C \end{cases}$$

When comparing process term weak exit rates, besides taking pre-emption into account, we also have to skip the comparison for classes that contain certain unobservable terms. More precisely, we distinguish among observable, initially unobservable, and fully unobservable terms. An observable term is a term that enables a non-τ-action that cannot be pre-empted by any enabled immediate τ-action. An initially unobservable term is a term in which all the enabled non-τ-actions are pre-empted by some enabled immediate τ-action, but at least one of the paths starting at this term with one of the higher priority enabled immediate τ-actions reaches an observable term. A fully unobservable term is a term in which all the enabled non-τ-actions are pre-empted by some enabled immediate τ-action, and all the paths starting at this term with one of the higher priority enabled immediate τ-actions are unobservable (note that $\underline{0}$ is fully unobservable).

The weak exit rate comparison with respect to observable and fully unobservable classes must obviously be performed. In order to maximize the abstraction power in the presence of quantitative information attached to immediate τ-actions, the comparison should be conducted with respect to the whole set $\mathcal{P}_{\rm fu}$ of fully unobservable process terms of $\mathcal{P}$. By contrast, the comparison with respect to initially unobservable classes should be skipped as each of them can reach an observable class.

Definition 7. *An equivalence relation* $\mathcal{B} \subseteq \mathcal{P} \times \mathcal{P}$ *is a weak extended Markovian bisimulation iff, whenever* $(P_1,P_2) \in \mathcal{B}$*, then for all action names* $a \in Name$ *and levels* $l \in \mathbb{Z}$ *such that no-pre*(l,P_1) *and no-pre*(l,P_2)*:*

$$rate_{\rm w}(P_1,a,l,C) = rate_{\rm w}(P_2,a,l,C) \quad \text{for all observable } C \in \mathcal{P}/\mathcal{B}$$
$$rate_{\rm w}(P_1,a,l,\mathcal{P}_{\rm fu}) = rate_{\rm w}(P_2,a,l,\mathcal{P}_{\rm fu})$$

Weak extended Markovian bisimilarity, denoted by $\approx_{\rm EMB}$*, is the union of all the weak extended Markovian bisimulations.*

As examples of weakly extended Markovian bisimilar process terms we mention:

$$P_1 \equiv <a,\lambda>.<\tau,\infty_{l,w}>.<b,\mu>.\underline{0}$$
$$P_2 \equiv <a,\lambda>.<b,\mu>.\underline{0}$$

and also:

$$P_3 \equiv <a,\lambda>.(<\tau,\infty_{l,w_1}>.<b,\mu>.\underline{0} + <\tau,\infty_{l,w_2}>.<c,\gamma>.\underline{0})$$
$$P_4 \equiv <a,\lambda \cdot \tfrac{w_1}{w_1+w_2}>.<b,\mu>.\underline{0} + <a,\lambda \cdot \tfrac{w_2}{w_1+w_2}>.<c,\gamma>.\underline{0}$$

which is related to vanishing state elimination. We also point out that $\approx_{\rm EMB}$ can abstract not only from intermediate immediate τ-actions, but also from intermediate unobservable self-loops, consistently with the fact that the probability to escape from them is 1. Moreover, $\approx_{\rm EMB}$ cannot abstract from initial immediate

τ-actions, otherwise compositionality with respect to the alternative composition operator would be broken. The careful classification of states on the basis of their functional and performance observability is a key ingredient thanks to which congruence and axiomatization can be achieved for $\approx_{\rm EMB}$. In particular, compositionality with respect to parallel composition is preserved by restricting to a well-prioritized subset of non-divergent process terms of $\mathcal{P}$ [5].

3 Noninterference Properties

In a typical two-level access system we distinguish between the high security access level and the low security access level. Information flowing from low level to high level is authorized, while the inverse flow is not. The noninterference analysis aims at revealing direct and indirect information flows, called covert channels, in the unauthorized path. In a process algebraic setting this classification of security levels is realized by assuming that the set $Name$ of action names includes a set $Name_L \subseteq Name$ of low-level names and a set $Name_H \subseteq Name$ of high-level names, such that $Name_L \cap Name_H = \emptyset$. These sets describe the interactions between the system and the low-level and high-level parts of the environment, respectively. We can also have that $Name_L \cup Name_H = Name$. If this is not the case, we assume that the remaining action names can be disregarded as they are not related to interactions with the users at different access clearances. For instance, they could represent synchronizing activities performed by different system components that cooperate to handle the service requests. Hence, they will be hidden by turning them into unobservable actions.

In the following, we describe two noninterference properties relying on this classification and on the bisimulation semantics, named strong noninterference property and strong local noninterference property.

3.1 Bisimulation-based Strong Noninterference

The requirement at the base of the lack of any interference from high level to low level can be easily expressed by a property called Strong Nondeterministic Noninterference, which informally says that a system is secure if its observable low-level behavior is the same in the presence and in the absence of high-level interactions. The stochastic version of this property is called Bisimulation-based Strong Stochastic Noninterference ($BSSNI$) and is defined as follows, where $P/Name_H$ expresses the system view in the presence of high-level actions, which are hidden because they cannot be explicitly seen by a low-level observer, while $P\backslash Name_H$ represents the system view whenever the high-level actions are absent, i.e. they are prevented from execution by means of the restriction operator.

Definition 8. ***(BSSNI)*** *$P \in \mathcal{P}_{\rm pc}$ is secure iff $P/Name_H \approx_{\rm EMB} P\backslash Name_H$.*

For brevity, we assume that P is performance closed. For the formal treatment of open systems with respect to high-level inputs, the interested reader is referred to [3]. The nondeterministic version of $BSSNI$, termed $BSNNI$ [9], is obtained

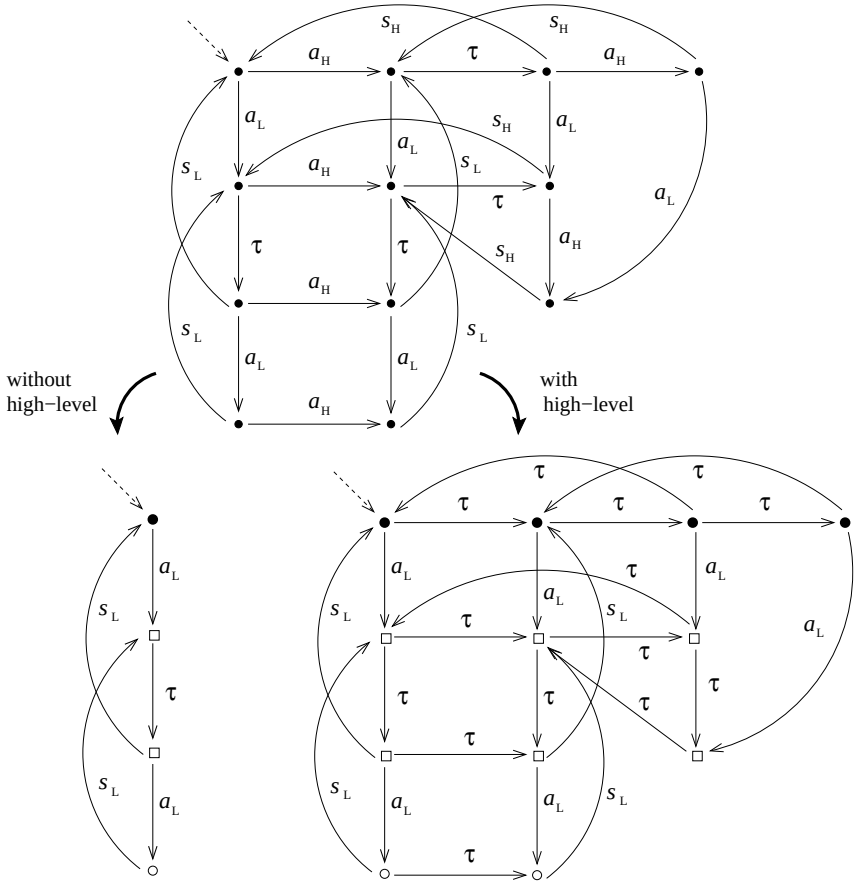

Fig. 1. Nondeterministic version of the client-server system and its low-level views.

in this framework by replacing $\approx_{\mathrm{EMB}}$ with the Milner's weak bisimilarity [15], denoted by $\approx_{\mathrm{B}}$.

Definition 9. ***(BSNNI)*** *$nd(P)$ is secure iff $nd(P)/Name_H \approx_{\mathrm{B}} nd(P)\backslash Name_H$.*

Similarly, the probabilistic version of *BSSNI*, termed *BSPNI* [3], is defined in the same framework by replacing $\approx_{\mathrm{EMB}}$ with the weak probabilistic bisimilarity of [6], denoted by $\approx_{\mathrm{PB}}$.

Definition 10. ***(BSPNI)*** *$pr(P)$ is secure iff $pr(P)/Name_H \approx_{\mathrm{PB}} pr(P)\backslash Name_H$.*

Example 2. We now discuss an incremental analysis of the client-server system introduced in Ex. 1. This is done by considering the three fine-grain notions of noninterference in order to show which kind of information flow can be revealed and what strategies should be implemented to circumvent the related

covert channels. We start by classifying the activities performed by the client-server system into low level and high level. The low-level client is represented by process A_L, thus the action name modeling the arrival process, a_L, and the action name modeling the delivered service, s_L, are the unique action names in $Name_L$. Similarly, given that A_H represents the high-level client, we assume that $Name_H = \{a_H, s_H\}$. The communications between the buffers and the server represent internal activities among components of the service center. Thus, they should not be observed by any client. Hence, the noninterference check is applied to system $CS/\{d_L, d_H\}$ even if, for brevity, we will continue to refer to CS.

First, consider the property *BSNNI* and the system $nd(CS)$. The underlying semantic model is represented by the labeled transition system $[\![nd(CS)]\!]$ in the upper part of Fig. 1, which also includes in its lower part the two labeled transition systems to compare according to the noninterference property. The weak bisimulation relating these two sub-systems is illustrated by graphically representing in the same way the states that belong to the same class. Hence, it is easy to see that the noninterference check is satisfied and the functional behavior of the system is secure. Intuitively, the availability to serve the low-level requests is never compromised independently of the behavior of the high-level client.

Second, consider the property *BSPNI* and the system $pr(CS)$. The semantics of this system is obtained from $[\![\mathrm{nd}(CS)]\!]$ by enriching every edge in Fig. 1 with a weight. The intuition is that with this extension we add fine-grain information which allow nondeterministic choices to be solved according to probability distributions modeling the quantitative behavior of system activities. In spite of this enriched, detailed description, the noninterference check based on *BSPNI* is satisfied and the system turns out to be secure. As a sketch of the proof of this result, consider the semantics of the sub-system $pr(CS)\backslash Name_H$.

From the unique state of class $\bullet$ it is possible to reach, through the action a_L, class $\square$ with probability 1. Then, after an internal move within the same class, the unique probabilistic choice of this sub-system governs the choice among the action s_L leading to class $\bullet$ and the action a_L leading to class $\circ$. Finally, from class $\circ$ the unique enabled transition, which is labeled with s_L, leads to class $\square$ with probability 1. Now, consider the semantics of the sub-system $pr(CS)/Name_H$. From states of class $\bullet$ it is possible to pass, through the weak transition $\tau^* a_L$, to class $\square$ with probability 1. With the same probability, we then reach, by following unobservable paths, one of two states of class $\square$ enabling the same probabilistic choice between s_L and a_L that characterizes the former sub-system. Finally, the two states of class $\circ$ lead with probability 1 to states of class $\square$ through the weak transition $\tau^* s_L$. Summing up, adding probabilistic information to the system does not increase the discriminating power of the low-level observer, because any high-level activity is not able to alter the unique low-level probabilistic choice.

Third, consider the property *BSSNI* and the system CS. The intuition is that with this extension the clients observe the passage of time and the priority-based behavior of the server. The introduction of this fine-grain information causes an interfering information flow from high level to low level, which is revealed by the

violation of the *BSSNI* condition. In the following, we show the nature of these covert channels and how the diagnostic information provided by $\approx_{\mathrm{EMB}}$ has been exploited to avoid them.

On the one hand, the high-level process A_H is immediately revealed by the low-level client. Indeed, $A_H/Name_H$ describes a working process that, according to the race policy, competes with the other durational processes, while $A_H \backslash Name_H$ does not (first interference). On the other hand, from the viewpoint of the low-level client, the time spent by the server to deliver the high-level services describes an observable busy waiting phase (second interference). From a performance-oriented perspective, these interferences affect the low-level throughput of the client-server system, in terms of number of s_L actions executed per unit of time, which is higher in the absence of high-level interactions. The removal of the two interferences requires strong control mechanisms that, as expected, aim at degrading the performance of the system in order to make the behavior of the high-level client transparent to the low-level client.

As far as the first interference is concerned, it is necessary to confine the behavior of the high-level client in order to hide its impact on the timing of the system activities. This can be done by employing a sort of high-level box which, somehow, limits and controls the activities performed by the high-level client. Formally, we replace process A_H with the following one:

$$H\_box \stackrel{\Delta}{=} <\tau, \infty_{l,w}>.\\ (<a_H, \infty_{h+1,w}>.<a_H, \lambda_H>.H\_box + <\tau, \infty_{h,w}>.<\tau, \lambda_H>.H\_box)$$

The action $<\tau, \infty_{l,w}>$ represents the activation of the box and is technically needed because it allows $\approx_{\mathrm{EMB}}$ to abstract from the subsequent intermediate immediate τ-actions. The branch guarded by the action $<\tau, \infty_{h,w}>$ is enabled if and only if the high-level client is absent. Its role is to simulate the presence of such a client in a way that makes the high-level behavior invisible to the low-level client. Intuitively, process H_box is a black box acting as an interface between the system and the high-level client, who can interact with such a box by pushing a button whenever he decides to communicate with the system.

As far as the second interference is concerned, making the server transparent to the low-level client is achieved by following an approach borrowed by round-robin scheduling strategies. In practice, the intuition is similar to that underlying the definition of process H_box. The service activities are divided into temporal slots, each one dedicated to a class of clients in a round-robin fashion. Independently of the presence of a pending request from a client of the currently managed class, the temporal slot is spent. In this way a low-level client cannot deduce whether the high-level slot has been actively exploited by the high-level client. Formally, we replace process S with the following one:

$$S_L \stackrel{\Delta}{=} <d_L, \infty_{f(L),w}>.<s_L, \mu_L>.S_H + <\tau, \mu_L>.S_H$$
$$S_H \stackrel{\Delta}{=} <d_H, \infty_{f(H),w}>.<s_H, \mu_H>.S_L + <\tau, \mu_H>.S_L$$

This is not enough to hide completely the interference of the high-level client. Indeed, whenever such a client is blocked because the related buffer is not available to accept further requests, then process H_box does not compete for the resource time. This observable behavior would reveal to the low-level client that

the high-level buffer is full. The covert channel can be avoided by changing the high-level buffer as follows:

$$B_H \stackrel{\Delta}{=} <a_H, *_w^0>.B'_H \qquad B'_H \stackrel{\Delta}{=} <d_H, *_w^{f(H)}>.B_H + <\tau, \lambda_H>.B'_H$$

■

3.2 Bisimulation-based Strong Local Noninterference

The introduction of fine-grain information about time makes the satisfaction of noninterference properties a very hard task, which can be accomplished through invasive strategies aiming at controlling the temporal behavior of the system. This claim is strengthened by the fact that we employed one of the least restrictive noninterference properties in the literature. As an example, we ignored the interference caused by a high-level user that blocks the high-level output modeling the service delivery, because this problem can be easily avoided by making such a communication asynchronous [1]. This and more subtle problems can be revealed by security properties that have been defined in the literature with the aim of capturing more information flows with respect to the noninterference property surveyed above. For instance, the strongest property of [9] is the Strong Bisimulation Nondeducibility on Compositions, which corresponds to the Strong Local Noninterference property that has been separately defined in [16]. In our framework, we consider such a property under the name Bisimulation-based Strong Stochastic/Probabilistic/Nondeterministic Local Noninterference, respectively, depending on the nature of the fine-grain information we take into consideration. The underlying intuition states that the absence of any interference is ensured when the low-level user cannot distinguish which, if any, high-level event has occurred at some point in the past.

Definition 11. ***(BSSLNI)*** *P is secure if and only if $\forall P' \in Der(P)$ and $\forall P''$ such that $P' \xrightarrow{a,\tilde{\lambda}} P''$, with $a \in Name_H$, then $P'\backslash Name_H \approx_{\mathrm{EMB}} P''\backslash Name_H$.*

In essence, this property states that the low-level view of the system is not affected by the execution of a high-level action. The nondeterministic version, *BSNLNI*, and the probabilistic version, *BSPLNI*, can be derived as before. However, the introduction of the temporal aspect changes the nature of the relations among these properties. The reason stems from the fine-grain information associated with the high-level actions. The intuition is that the strong local noninterference property analyzes the low-level view of the system before and after the execution of a high-level action without taking into account neither the time spent by its execution (if it is durational) nor its priority (if it is immediate). In practice, in the stochastic setting the original notions of strong noninterference and strong local noninterference cannot be compared.

Theorem 1. ***(Inclusion relations)***
(i) $BSNLNI \subset BSNNI$
(ii) $BSPLNI \subset BSPNI$
(iii) $BSSLNI \not\subset BSSNI$

Proof. (i) *and* (ii) *are standard results shown in [9] and [3], respectively. To show* (iii)*, consider the process term* $P_7 \equiv <h,\lambda>.<l,\mu>.\underline{0} + <l,\mu>.\underline{0}$*, which satisfies BSSLNI, because its low-level view is* $<l,\mu>.\underline{0}$ *before and after the execution of the high-level action. However,* P_7 *is not BSSNI secure, because* $<l,\mu>.\underline{0}$ *and* $P_8 \equiv <\tau,\lambda>.<l,\mu>.\underline{0} + <l,\mu>.\underline{0}$ *are not weakly extended Markovian bisimilar. The motivation is given by the race policy between the two exponentially timed actions. Similarly, consider the process term* $P_9 \equiv <h,\infty_{2,w}>.P_{10} + P_{10}$*, where* $P_{10} \equiv <l,\infty_{1,w}>.\underline{0} + <l',\infty_{2,w}>.\underline{0}$*.* P_9 *is clearly BSSLNI secure, but not BSSNI secure: in* $P_9/Name_H$ *the low-level action with name* l *is initially pre-empted, thus altering the probability of executing the two low-level actions.*

Example 3. As far as the client-server system is concerned, $nd(CS)$ and $pr(CS)$ satisfy *BSNLNI* and *BSPLNI*, respectively, provided that the delivery of the high-level service is made asynchronous. Such a relaxation is not enough to make process *CS* a *BSSLNI*-secure system. For this purpose, all the exponentially timed actions executed by process *H_box* must be made invisible in order to keep them out of the control of the high-level client. Then, process *H_box* should be further complicated in such a way that the (priority, probabilistic, and temporal) low-level view of the system must be the same before and after the execution of action $<a_H,\infty_{h,w}>$.

The stochastic noninterference check pinpoints the covert channels that affect the quantitative behavior of the system and, therefore, the delivered quality of service. The low-level throughput of the original client-server system offers different results depending on the presence of high-level interactions. This different quantitative behavior that is exhibited by the two sub-systems is formally captured by the $\approx_{\mathrm{EMB}}$-based check as a diagnostic information revealing the interference. Estimating the metric above in the presence and in the absence of the high-level client is a way to measure the covert channel bandwidth. The same performance metric reveals the quality of service degradation that is paid by making the system completely secure through the high-level black box and the revised versions of the service center components. Hence, a balanced trade-off between security and performance strictly depends on the value of this metric. From the analysis standpoint, since the semantic model underlying the system *CS* is a CTMC, such a metric can be easily evaluated through reward-based steady-state numerical analysis in order to obtain an estimation of the information leakage on the long run [20]. ■

In conclusion, strong notions of noninterference are hard to accomplish when modeling real-world systems that turn out to be much more complex than the client-server example. Adding fine-grain information such as time opens new scenarios where covert channels cannot be completely eliminated without severely limiting the system behavior. On the basis of the discussed strategies that are needed to pass the *BSSNI*-based check, it is easy to observe that the minimization of the covert channel bandwidth corresponds to a reduction of the quality of service. Hence, an approach towards the mitigation of the strict, unrealistic constraints imposed by the noninterference properties should be based on

tolerance thresholds, which are expressed in terms of negligible difference with respect to a family of performance metrics related to quality of service. A similar approach is used in [2], where the impact of the high-level strategies (and of the related countermeasures adopted by the system) is estimated from the performance standpoint.

4 Conclusions

The definition and analysis of stochastic noninterference is not only yet another extension of classical nondeterministic noninterference. In fact, it allowed us to highlight some interesting relations between the noninterference approach to information flow analysis and the quality of the service delivered by real-world systems. In a framework where time is important, the objective is not to guarantee the complete absence of any kind of interference, which is a task that is either impossible to achieve or impractical because of the strong functional limitations that should be imposed. Instead, the attention should pass to the analysis of the performance measures of interest that are affected by the high-level interferences. The goal is to estimate whether such interferences are negligible (e.g. because they are revealed if and only if the system execution is observed for a long time) and, therefore, cause a tolerable amount of information leakage in terms of sensitive data that are revealed on the long run.

From a practical standpoint, we intend to use the notion of stochastic noninterference to support a methodology for the analysis of a balanced trade-off between the security degree of the system and the impact of different securing strategies on the system quality of service. The intuition is that the stronger the noninterference validation is, the more complicated and invasive the related securing strategy should be. From a theoretical standpoint, it would be useful to investigate properties stronger than *BSSNI* which do not imply the explicit analysis of every possible high-level user, as instead required, e.g., by the Noninducibility on Compositions property.

Acknowledgement
The authors thank the anonymous referees for their valuable comments. This work has been funded by MIUR-PRIN project *PaCo – Performability-Aware Computing: Logics, Models, and Languages.*

References

1. Aldini, A.: Classification of security properties in a Linda-like process algebra. J. of Science of Computer Programming **63** (2006) 16–38.
2. Aldini, A., Bernardo, M.: A formal approach to the integrated analysis of security and QoS. J. of Reliability Engineering & System Safety **92** (2007) 1503–1520.
3. Aldini, A., Bravetti, M., Gorrieri, R.: A process-algebraic approach for the analysis of probabilistic noninterference. J. of Computer Security **12** (2004) 191–245.

4. Aldini, A., Bravetti, M., Di Pierro, A., Gorrieri, R., Hankin, C., Wiklicky, H.: Two formal approaches for approximating noninterference properties. Foundations of Security Analysis and Design II, LNCS **2946** (2004) 1–43.
5. Bernardo, M., Aldini, A.: Weak Markovian bisimilarity: abstracting from prioritized/weighted internal immediate actions. 10th Italian Conf. on Theoretical Computer Science, World Scientific (2007) 39–56.
6. Bravetti, M., Aldini, A.: Discrete time generative-reactive probabilistic processes with different advancing speeds. Theoretical Comp. Science **290** (2003) 355–406.
7. Di Pierro, A., Wiklicky, H.: Quantifying timing leaks and cost optimisation. 10th Conf. on Information and Comm. Security, LNCS **5308** (2008) 81–96.
8. Focardi, R., Martinelli, F., Gorrieri, R.: Information flow analysis in a discrete-time process algebra. IEEE Computer Security Foundations Workshop (2000) 170–184.
9. Focardi, R., Gorrieri, R.: A classification of security properties. J. of Computer Security **3** (1995) 5–33.
10. van Glabbeek, R.J., Smolka, S.A., Steffen, B.: Reactive, generative and stratified models of probabilistic processes. Information and Comput. **121** (1995) 59–80.
11. Goguen, J.A., Meseguer, J.: Security policy and security models. IEEE Symp. on Security and Privacy (1982) 11–20.
12. Lanotte, R., Maggiolo-Schettini, A., Troina, A.: A classification of time and/or probability dependent security properties. 3rd Int. Workshop on Quantitative Aspects of Programming Languages, ENTCS **153** (2005) 177–193.
13. Mantel, H., Sudbrock, H.: Comparing countermeasures against interrupt-related covert channels in an information-theoretic framework. IEEE Computer Security Foundations Symposium (2007) 326–340.
14. McLean, J.: Security models and information flow. IEEE Symp. on Research in Security and Privacy (1990) 180–187.
15. Milner, R.: Communication and Concurrency. Prentice Hall (1989).
16. Roscoe, A.W., Reed, G.M., Forster, R.: The successes and failures of behavioural models. Millenial Perspectives in Computer Science (2000).
17. Ryan, P.Y.A., Schneider, S.A.: Process algebra and non-interference. J. of Computer Security, **9** (2001) 75–103.
18. Sabelfeld, A., Sands, D.: Probabilistic noninterference for multi-threaded programs. IEEE Computer Security Foundations Workshop (2000) 200–214.
19. Smith, G.: Probabilistic noninterference through weak probabilistic bisimulation. IEEE Computer Security Foundations Workshop (2003) 3–13.
20. Stewart, W.J.: Introduction to the numerical solution of Markov chains. Princeton University Press (1994).
21. Volpano, D., Smith, G.: Probabilistic noninterference in a concurrent language. IEEE Computer Security Foundations Workshop (1998) 34–43.
22. Wittbold, J.T., Johnson, D.M.: Information flow in nondeterministic systems. IEEE Symp. on Research in Security and Privacy (1990) 144–161.