Information Flow Testing *

Damas P. GRUSKA

Institute of Informatics, Comenius University,
Mlynska dolina, 842 48 Bratislava, Slovakia,
gruska@fmph.uniba.sk.

Abstract. Process testing as a way to obtain information on confidential data is investigated. Our working formalism is based on an appropriate (probabilistic) process algebra and (probabilistic) testing. We define testing noninterference as well as sets of private actions which execution is guaranteed by a given test and sets of actions which execution could be excluded by a given test. Moreover, we relate obtained information to a size of the test.

Keywords: probabilistic process algebras, information flow, security

1 Introduction

We propose a formalism for a security analysis of systems specified by process algebras. It allows us to formalize security properties based on an absence of information flow between private and public system's activities. The presented approach combines several ideas emerged from the security theory. We exploit an idea (of an absence) of information flow between public and private system's behaviour (see [GM82]). This concept has been exploited many times in various formalisms. For example, the security property called Bisimulation Strong Nondeterministic Non-Interference requires that it cannot be distinguished (by means of bisimulation) between forbidding and hiding of private actions. In our approach we exploit this idea but we weaken it by requiring that forbidding and hiding of the private actions cannot be distinguished by a given test, i.e. we exploit a kind of testing equivalence (see also [NH84,SL95]). In [Gru11] we have studied information flows by means of two sets - the set of private actions which execution is guaranteed by a given observation (trace) of public actions and the set of actions which execution is excluded by a given observation. Here we apply this idea and we define the sets of gained and excluded actions by a given test.

Qualitative security properties are often criticized for being either too restrictive or too benevolent. For example, a standard access control process should be considered insecure even if there always exists some (even very small) information flow which could help an attacker who tries to learn a password. By every attempt an attacker can learn, at least, what is not the correct one. On the other side, it can happen that the sets of excluded and/or gained (possible) passwords are empty but some passwords "almost" belong to some of these sets.

* Work supported by the grant VEGA 1/1333/12.

There are several ways to overcome these disadvantages. An amount of leaked information could be expressed by means of the Shannon's information theory as it was done, for example, in [CHM07,CMS09] for simple imperative languages and in [Gru08] for process algebras. Another possibility is to exploit the probability theory as it was done for process algebras in [Gru09]. In this way we can obtain quantification of information flow either as a number of bits of private information which could leak or as a probability that an intruder can learn some secrete property. Here we exploit the probabilistic process algebra (to describe either the test, tested process, or both) to express probabilities for private actions being in the set of gained or excluded actions.

Moreover, we relate an amount of obtained information to the size of the test. The presented testing approach is strictly stronger then that of [Gru11], which is based on simple process's observations. On the other side, bisimulation based approach, which is stronger, is often too strong and does not correspond to real(istic) possible intruders. Moreover, testing allow us, besides other advantages, to express security of a system with respect to size of the test which could jeopardize its security. Hence the resulting level of security gives us relevant information on real (practical) system security.

The paper is organized as follows. Our working formalism, i.e. the probabilistic process algebra, is introduced in Section 2. In Section 3 we describe our testing scenario. In Sections 4 we define test based noninterference and in Section 5 the sets of gained and excluded actions by a given test. Section 6 is devoted to probabilistic tested noninterference.

2 Probabilistic Process Algebra

In this section we define the Probabilistic Process Algebra, pCCS for short, which is based on Milner's CCS (see [Mil89]). First, we assume a set of atomic action symbols A not containing symbol τ and such that for every $a \in A$ there exists $\overline{a} \in A$ and $\overline{\overline{a}} = a$. We define $Act = A \cup \{\tau\}$. We assume that $a, b, \ldots$ range over A and $u, v, \ldots$ range over Act. Assume the signature $\Sigma = \bigcup_{n\in\{0,1,2\}} \Sigma_n$, where

$$\begin{aligned}
\Sigma_0 &= \{Nil\} \\
\Sigma_1 &= \{x. \mid x \in Act\} \cup \{[S] \mid S \text{ is a relabeling function}\} \\
&\quad \cup \{\backslash M \mid M \subseteq A\} \\
\Sigma_2 &= \{|, +\}
\end{aligned}$$

with the agreement to write unary action operators in prefix form, the unary operators $[S], \backslash M$ in postfix form, and the rest of operators in infix form. Relabeling functions, $S : Act \to Act$ are such that $\overline{S(a)} = S(\bar{a})$ for $a \in A$ and $S(\tau) = \tau$.

The set of CCS terms over the signature Σ is defined by the following BNF notation:

$$P \ ::= \ X \ \mid \ op(P_1, P_2, \ldots P_n) \ \mid \ \mu X P$$

where $X \in Var$, Var is a set of process variables, $P, P_1, \ldots P_n$ are CCS terms, $\mu X-$ is the binding construct, $op \in \Sigma$.

We will use an usual definition of opened and closed terms where μX is the only binding operator. Guarded closed terms (i.e. terms for which each occurrence of X is within some subexpression of the form $y.A$) are called CCS processes. Note that Nil will be often omitted from processes descriptions and hence, for example, instead of $a.b.Nil$ we will write just $a.b$. Structural operational semantics for CCS processes is given by labeled transition systems. The set of CCS processes (denoted as CCS) represents a set of states, labels are actions from Act (see [Mil89]). The transition relation $\rightarrow$ is a subset of $\mathrm{CCS} \times Act \times \mathrm{CCS}$. We write $P \xrightarrow{x} P'$ instead of $(P, x, P') \in \rightarrow$ and $P \not\xrightarrow{x}$ if there is no P' such that $P \xrightarrow{x} P'$. The meaning of the expression $P \xrightarrow{x} P'$ is that the term P can evolve to P' by performing action x, by $P \xrightarrow{x}$ we will denote that there exists a term P' such that $P \xrightarrow{x} P'$.

Now we add probabilities to CCS calculus. We will follow alternating model (the approach presented in [HJ90]) which is neither reactive nor generative nor stratified (see [LN04]) but instead of that it will be based on separation of probabilistic and nondeterministic transitions and states. Probabilistic transitions are not associated with actions but they are labeled with probabilities. In so called probabilistic states a next transition is chosen according to probabilistic distribution. For example, process $a.(0.3.b.Nil \oplus 0.7.(a.Nil + b.Nil))$ can perform action a and after that it reaches the probabilistic state and from this state it can reach with probability 0.3 the state where only action b can be performed or with probability 0.7 it can reach the state where it can perform either a or b .

Formally, to add probabilities to CCS calculus we introduce a new operator $\bigoplus_{i \in I} q_i.P_i$ with q_i being real numbers in $(0, 1]$ such that $\sum_{i \in I} q_i = 1$. Processes which can perform as the first action probabilistic transition will be called probabilistic processes or states (to stress that P is non-probabilistic process we will sometimes write P_N if necessary). We require that all P_i processes in $\bigoplus_{i \in I} q_i.P_i$ and in $P_1 + P_2$ are non-probabilistic ones. By pCCS we will denote the set of all probabilistic and non-probabilistic processes and all definitions and notations for CCS processes are extended for pCCS ones. We need new transition rules for pCCS processes (here p, q and q_i for $i \in I$ are real numbers from $(0, 1]$).

$$\frac{}{P_N \xrightarrow{1} P_N}\ A1 \qquad \frac{}{\bigoplus_{i \in I} q_i.P_i \xrightarrow{q_i} P_i}\ A2$$

$$\frac{P \xrightarrow{q} P', Q \xrightarrow{r} Q'}{P \mid Q \xrightarrow{q.r} P' \mid Q'}\ Pa$$

For probabilistic choice we have the rule $A2$ and for a probabilistic transition of two processes running in parallel we have the rule Pa. The technical rule $A1$ enables parallel run of probabilistic and non-probabilistic processes by allowing to non-probabilistic processes to perform $\xrightarrow{1}$ transition and hence the rule Pa could be applied.

Introducing probabilities to process algebras usually causes several technical complications. For example, an application of the restriction operator to probabilistic processes may lead to unwanted deadlock states or to a situation when a sum of probabilities of all outgoing transitions is less than 1. A normalization is usually applied to overcome similar situations. We do not need to resolve such situations on the level of pCCS calculus since we will use only relative probabilities of sets of computations. To compute these probabilities normalization will be also exploited but only as the very last step.

To express what an observer can see from system behaviour we will define modified transitions $\stackrel{x}{\Rightarrow}_M$ which hide actions from M (as well as τ and probabilities). Formally, we will write $P \stackrel{x}{\Rightarrow}_M P'$ for $M \subseteq A$ iff $P \stackrel{s_1}{\rightarrow}\stackrel{x}{\rightarrow}\stackrel{s_2}{\rightarrow} P'$ for $s_1, s_2 \in (M \cup \{\tau\} \cup (0,1])^\star$ and $P \stackrel{s}{\Rightarrow}_M$ instead of $P \stackrel{x_1}{\Rightarrow}_M \stackrel{x_2}{\Rightarrow}_M \ldots \stackrel{x_n}{\Rightarrow}_M$. Instead of $\Rightarrow_\emptyset$ we will write $\Rightarrow$ and instead of $\Rightarrow_{\{h\}}$ we will write $\Rightarrow_h$. We will write $P \stackrel{x}{\Rightarrow}_M$ if there exists P' such that $P \stackrel{x}{\Rightarrow}_M P'$. By ϵ we will denote the empty sequence of actions and by $s \sqsubseteq s'$, $s, s' \in (Act \cup (0,1])^\star$ we will denote that s is a prefix of s'. By $Sort(P)$ we will denote the set of actions from A which can be performed by P i.e. $Sort(P) = \{x | P \stackrel{s.x}{\longrightarrow}$ for some $s \in (Act \cup (0,1])^\star$ and $x \in A\}$. By $\widehat{x}$ we mean x for $x \neq \tau$ and ϵ otherwise.

Let $s \in (Act \cup (0,1])^\star$. By $s|_B$ we will denote the sequence obtained from s by removing all actions not belonging to B and we will write $x \in s$ if the sequence s contains x as its element. We will write $\bar{s}$ for sequence obtained from s in such a way that every action in s is replaced by its complementary action if such action exists and otherwise it is left unchanged. For example, if $s = a.\bar{b}.0, 2.\tau.c$ we have $\bar{s} = \bar{a}.b.0, 2.\tau.\bar{c}$.

We conclude this section with definitions of trace equivalence, weak simulation and weak bisimulation.

Definition 1. *The set of weak traces of process P with respect to the set $M, M \subseteq A$ is defined as $Tr_{wM}(P) = \{s \in A^\star | \exists P'.P \stackrel{s}{\Rightarrow}_M P'\}$. Instead of $Tr_{w\emptyset}(P)$ we will write $Tr_w(P)$.*

Two processes P and Q are weakly trace equivalent with respect to M ($P \approx_{wM} Q$) iff $Tr_{wM}(P) = Tr_{wM}(Q)$. We will write $\approx_w$ instead of $\approx_{w\emptyset}$.

Definition 2. Let (CCS, $Act, \rightarrow$) be a labeled transition system (LTS). A relation $\Re \subseteq$ CCS $\times$ CCS is called a *weak bisimulation* if it is symmetric and it satisfies the following condition: if $(P, Q) \in \Re$ and $P \stackrel{x}{\rightarrow} P', x \in Act$, then there exists a process Q' such that $Q \stackrel{\widehat{x}}{\Rightarrow} Q'$ and $(P', Q') \in \Re$. Two processes P, Q are *weakly bisimilar*, abbreviated $P \approx Q$, if there exists a weak bisimulation relating P and Q. If it is not required that relation $\Re$ is symmetric we call it simulation and we say that process P simulates process Q, abbreviated $P \prec Q$, if there exists a simulation relating P and Q.

3 Testing

In this section we define basic testing scenario which will be applied in the subsequent sections. First we start with M-simulations. We say that process P passes test T if whenever the test makes an action then process can perform the complementary action (optionally together with τ action and actions from M) in such a way that some resulting process again passes the resulting test. By test T we will consider process which does not contain τ actions and such that $Sort(T) \cap M = \emptyset$.

Definition 3. *Let $M \subset A$. We say that process P passes test T with respect to M iff there exists relation $\Re$ called M -simulation such that whenever $(T, P) \in \Re$ then for every $a \in A$ if $T \xrightarrow{a} T'$ then there exists process P' such that $P \stackrel{\bar{a}}{\Rightarrow}_M P'$ and $(T', P') \in \Re$. We will denote the union of all M -simulations as $\preceq_M$. If $M = \emptyset$ we will write $\preceq$ instead of $\preceq_\emptyset$.*

Now we will present some useful properties of testing. The first one claims that whenever a process passes a stronger test (in a sense of simulation) then it has to pass also a weaker test.

Lemma 1. *Let $T_1 \prec T_2$. Then $T_2 \preceq_M P$ implies $T_1 \preceq_M P$.*

Proof. Let $T_2 \preceq_M P$ but $T_1 \not\preceq_M P$. Since $T_1 \prec T_2$ we know that every "computation" (execution) of T_1 could be simulated by T_2. This is true also for a computation due to which process P does not pass test T_1 and hence we have a contradiction with $T_2 \preceq_M P$.

The following lemma is mostly a consequence of the previous one. It states some simple compositional results.

Lemma 2. *Let $T_1 + T_2 \preceq_M P$ or $T_1|T_2 \preceq_M P$. Then $T_1 \preceq_M P$. Let $T \preceq_M P$. Then $T \setminus M' \preceq_M P$ for every $M', M' \subseteq A$. Let $T \preceq_M P$. Then $x.T \preceq_M x.P$.*

Proof. Except the last implication, all properties are direct consequences of the previous lemma since $T_1 \prec T_1 + T_2$, $T_1 \prec T_1|T_2$, $T \setminus M' \prec T$. The last implication follows directly from Definition 3.

A kind of an inverse lemma to Lemma 1 holds: if a weaker process passes a test then also a stronger one has to pass the test.

Lemma 3. *Let $P_1 \prec P_2$. Then $T \preceq_M P_1$ implies $T \preceq_M P_2$.*

Proof. Again let $T \not\preceq_M P_2$. But since every computation of P_1 could be simulated by P_2 and P_1 passes the test we have have a contradiction.

Now we present some compositional properties for tested processes.

Lemma 4. *Let $T \preceq P$. Then $T \preceq_M P + Q$, $T \preceq_M P|Q$ and $x.T \preceq_M x.P$ for $x \notin M$.*

Proof. The proof follows directly from the previous lemma and the fact that $P \prec P + Q, P|Q$.

Now we will define the test equivalence with respect to given test T. Two process will be test equivalent if they cannot be distinguished by test T. In fact we should prove that this relation is really equivalence relation but this follows from properties of logical equivalence ($\leftrightarrow$).

Definition 4. *We define the test equivalence with respect to test T and set $M, M \subset A$ (we will denote it as $\approx_{T,M}$) as $P \approx_{T,M} Q$ iff $T \preceq_M P \leftrightarrow T \preceq_M Q$.*

We will need also stronger variant of testing, which requires that every "test" successor of a tested process has to pass "testing".

Definition 5. *Let $M \subset A$. We say that process P strongly passes test T with respect to M iff there exists relation $\Re$ called M -strong simulation such that whenever $(T, P) \in \Re$ then for every $a \in A$ if $T \xrightarrow{a} T'$ then $P \stackrel{\bar{a}}{\Rightarrow}_M$ and for every P' such that $P \stackrel{\bar{a}}{\Rightarrow}_M P'$ we have $(T', P') \in \Re$.*

We will denote the union of all M -strong simulations as $\preceq_{sM}$.

Clearly, strong testing is really stronger than testing (see also Example 1). The following two lemmas, which are counterparts of Lemma 1 and 2 hold also for the strong testing and also their proofs are similar. Note that the other two lemmas (3 and 4) which hold for the testing do not hold for the strong testing. But a weaker variant of Lemma 3 still holds (Lemma 7).

Lemma 5. *Let $T_1 \prec T_2$. Then $T_2 \preceq_{sM} P$ implies $T_1 \preceq_{sM} P$.*

Lemma 6. *Let $T_1 + T_2 \preceq_{sM} P$ or $T_1|T_2 \preceq_{sM} P$. Then $T_1 \preceq_{sM} P$.*

Lemma 7. *Let $T \preceq P$ and $T \preceq Q$. Then $T \preceq_{sM} P + Q$.*

Proof. The proof follows from the fact that every successor of $P+Q$ is a successor either of P or Q. Note that without the assumption $T \preceq Q$ (as it is in Lemma 7) this lemma does not hold.

Now we are ready to define the strong test equivalence.

Definition 6. *We define strong test equivalence with respect to test T and set $M, M \subset A$ (we will denote it as $\approx_{s,T,M}$) as $P \approx_{s,T,M} Q$ iff $T \preceq_{sM} P \leftrightarrow T \preceq_{sM} Q$.*

Example 1. Let $T = a.(b.Nil + c.Nil)$ and $M = \{d, d'\}$. Then we have $P \approx_{T,M} P'$ and $P \not\approx_{s,T,M} P'$ for $P = \tau.d.\bar{a}.(d'.b.a.b.Nil + \tau.\bar{c}.Nil) + a.d.b.d.c.d.Nil$, $P' = \tau.d.\bar{a}.(d'.b.a.b.Nil + \tau.\bar{c}.Nil)$. Note that in general it holds $\approx_{s,T,M} \subset \approx_{T,M}$.

4 Non-interference by testing

To define non-interference for process algebras we suppose that all actions are divided into two groups, namely public (low level) actions L and private (high level) actions H i.e. $A = L \cup H, L \cap H = \emptyset$. Moreover, we suppose that $H \neq \emptyset$ and $L \neq \emptyset$ and that for every $h \in H, l \in L$ we have $\overline{h} \in H, \overline{l} \in L$. To denote sequences of public actions, i.e. the sequences consisting of actions from L and sequences of private actions from H, we will use notation $\tilde{l}, \tilde{l'}, \ldots$ for the sequences from $L^\star$ and $\tilde{h}, \tilde{h'}, \ldots$ for the sequences from $H^\star$, respectively. Note that the set of actions A could be divided into more than two subsets, what would correspond to more levels of classification. All the following concepts could be naturally extended for such setting.

First we formally define an absence-of-information-flow property - Bisimulation Strong Nondeterministic Non-Interference (BSNNI, for short, see [FGM00]). Process P has BSNNI property (we will write $P \in BSNNI$) if $P \setminus H$ behaves like P for which all high level actions are hidden for an observer. To express this hiding we introduce the hiding operator $P/M, M \subseteq A$, for which it holds that if $P \xrightarrow{a} P'$ then $P/M \xrightarrow{a} P'/M$ whenever $a \notin M \cup \bar{M}$ and $P/M \xrightarrow{\tau} P'/M$ whenever $a \in M \cup \bar{M}$. The formal definition of BSNNI follows.

Definition 7. *Let $P \in CCS$. Then $P \in BSNNI$ iff $P \setminus H \approx P/H$.*

Now we define a new security property called a security test which is inspired by property BSNNI. A process passes a test if a tester cannot detect communication via actions from M of tested process (see Fig 1., here we assume that $Sort(Q) \subseteq M$).

Definition 8. *Let T be a test and $M \subset A$. We say that CCS process P passes security test T with respect to M (we will write $P \in ST_{T,M}$) if $P \approx_{T,M} P/M$.*

Fig. 1. Testing scenario

Now we present some useful properties of security property $ST_{T,M}$.

Lemma 8. *Let $M_1 \subseteq M_2$. Then $ST_{T,M_2} \subseteq ST_{T,M_1}$.*

Proof. Let $P \in ST_{T,M_2}$. We have to show that $P \approx_{T,M_1} P \setminus M_1$ but it is enough to show that if $T \preceq_{M_1} P$ then $T \preceq_{M_1} P \setminus M_1$. So let $T \preceq_{M_1} P$. From the assumption we know that if $T \preceq_{M_2} P$ then $T \preceq_{M_2} P \setminus M_2$. But process $P \setminus M_2$ cannot perform actions from M_2 hence in this case relation $\Rightarrow_{M_2}$ coincides with $\Rightarrow$ but for processes the of the form $Q \setminus M_1$ in general it holds that if $Q \setminus M_1 \stackrel{\bar{a}}{\Rightarrow}$ then also $Q \setminus M_1 \stackrel{\bar{a}}{\Rightarrow}_{M_1}$ with the same result. Hence $T \preceq_{M_1} P \setminus M_1$.

Lemma 9. *Let $P \in BSNNI$. Then we have $P \in ST_{T,H}$ for every test T.*

Proof. Let $P \in BSNNI$ and $P \notin ST_{T,H}$ for some test T. Then we have $P \not\approx_{T,M} P/M$ i.e. test T can distinguish between P and P/M (with respect to M) what is a contradiction with our assumption that $P \approx_{T,M} P/M$.

Lemma 10. *Let $P \notin BSNNI$. Then there exists finite state test T such that $P \notin ST_{T,H}$.*

Proof. Let $P \notin BSNNI$ i.e. there does not exist a weak bisimulation relation such that it would contain $(P, P/M)$. In the other words whenever we try to construct such the relation we fail and we obtain pair (P, Q) of successors of P and P/M, respectively, such that there is an action which could be performed only by one of these processes. In this way we can construct the required test.

On the other side no finite state test is strong enough that it could cause $ST_{T,H}$ being as strong as $BSNNI$.

Lemma 11. *There does not exist finite state test T such that whenever $P \in ST_{T,H}$ then $P \in BSNNI$.*

Proof. It holds $BSNNI \subseteq ST_{T,H}$ (Lemma 9). Suppose that there exists finite state test T such that $ST_{T,H} = BSNNI$. Let T has k states. Let us consider process P which can perform public action l or private action h in any state except that already $k+1$-times action l was performed. In such the case only l action can be performed. Clearly $P \notin BSNNI$ but test T cannot distinguish between P and P/H.

Lemma 12. *Let $T_1 \prec T_2$. Then $ST_{T_2,M} \subseteq ST_{T_1,M}$.*

Proof. Let $P \in ST_{T_2,M}$ and suppose that $P \notin ST_{T_1,M}$ i.e. test T_1 can distinguish between P and P/M, but since $T_1 \prec T_2$ also T_2 should be able to distinguish between them what is in contradiction with our assumption (see Lemma 1).

Lemma 13. $ST_{T_1|T_2,H} \subseteq ST_{T_1+T_2,H} \subseteq ST_{T_1,H} \cap ST_{T_1,H}$.

Proof. According to the previous lemma it is enough to show that $T_1, T_2 \prec T_1 + T_2 \prec T_1|T_2$ what follows directly from inference rules for $+, |$ and the definition of simulation $\prec$.

The following compositional properties hold for $ST_{T,M}$ (for the proof see Lemma 4).

Lemma 14. *Let $P, Q \in ST_{T,M}$. Then $P + Q \in ST_{T,M}$ and $x.P \in ST_{x.T,M}$ for $x \notin M$.*

5 Testing gained and excluded private actions

In this section we will study processes which do not have property $ST_{T,M}$, i.e. test T could detect occurrence of some actions belonging to M. We will define which information on private actions could be obtained by T. First we define a set of private actions which occurrence can be learned by test T.

Definition 9. *We define the set of gained private actions of P by test T (denoted as $g(P,T)$) as*

$$g(P,T) = \bigcup_{x \in H} \{x | P \in ST_{T,H\setminus\{x\}} \wedge P \notin ST_{T,H}\}.$$

According to Definition 9 , set $g(P,T)$ represents private actions which occurrence could be revealed by test T. Now we will formulate some properties of this concept. First, we show that by a stronger test we can obtain more information as it is stated by the following proposition.

Proposition 1. *For every two tests T, T' such that $T' \prec T$ it holds $g(P,T') \subseteq g(P,T)$ for every process P.*

Proof. Let $x \in g(P,T')$ i.e. $P \in ST_{T,H\setminus\{x\}} \wedge P \notin ST_{T,H}$. Suppose that $x \notin g(P,T')$ i.e. $P \notin ST_{T',H\setminus\{x\}} \vee P \in ST_{T',H}$. From Lemma 8 we know that if $P \in ST_{T',H}$ then $P \in ST_{T',H\setminus\{x\}}$ and on the other side from Lemma 12 we know that $P \notin ST_{T,H}$ implies $P \notin ST_{T',H}$ what is in contradiction with the assumption that $x \notin g(P,T')$.

In [Gru11] we have defined a set of gained actions by observing a sequence of public actions $\tilde{l}$. We repeat the definition.

Definition 10. *Let $P \in CCS$ and $\tilde{l} \in Tr_{wH}(P)$. We define the set of gained private action of P by observing $\tilde{l}$ as*

$$g(P,\tilde{l}) = \{h | h \in H, P \stackrel{\tilde{l}}{\not\Rightarrow}_{H\setminus\{h\}}\}.$$

As it is clear from the following lemma and example, the test based notion of the gained private actions is strictly stronger then observation based one.

Lemma 15. *Let $\bar{\tilde{l}} \in Tr_{wH}(P)$. Then it holds $g(P,\tilde{l}.Nil) = g(P,\bar{\tilde{l}})$.*

Proof. Let $x \in g(P,\tilde{l}.Nil)$ i.e. $P \in ST_{\tilde{l}.Nil,H\setminus\{x\}} \wedge P \notin ST_{\tilde{l}.Nil,H}$. From this we know that $P \stackrel{\tilde{l}}{\Rightarrow}_H$ and $P \stackrel{\tilde{l}}{\not\Rightarrow}_{H\setminus\{x\}}$ i.e. that $x \in g(P,\bar{\tilde{l}})$.

Now let $x \in g(P,\bar{\tilde{l}})$ i.e. $P \stackrel{\tilde{l}}{\not\Rightarrow}_{H\setminus\{x\}}$. From this we know that $P \notin ST_{\tilde{l}.Nil,H}$ and from the assumption we know that $P \in ST_{\tilde{l}.Nil,H\setminus\{x\}}$ and hence $x \in g(P,\tilde{l}.Nil)$.

Corollary 1. Let $\tilde{l} \in Tr_{wH}(T)$, then it holds $g(P,\bar{\tilde{l}}) \subseteq g(P,T)$.

Proof. Let $T' = \bar{\tilde{l}}.Nil$, then we have from Propositions 1 and Lemma 15 that $g(P, \bar{\tilde{l}}) \subseteq g(P, T)$.

Example 2. Let $P = l_1.h.l_2.Nil + l_1.l_2.Nil$ and $P' = l_1.h.h'.l_2.Nil + l_1.h.l_2.Nil$. Let $\tilde{l} = l_1.l_2$ then we have $g(P, \tilde{l}) = \emptyset, g(P', \tilde{l}) = \{h\}$. Let $P'' = l.l_1.Nil + l.(h.l_1.Nil + l_2.Nil)$ and $T = \bar{l}.(\bar{l}_1.Nil + \bar{l}_2.Nil)$ then we have $g(P'', T) = \{h\}$ but $g(P, \tilde{l}) = \emptyset$ for every $\tilde{l}$.

Now we present some compositional results (for composition of tests as well as tested processes).

Proposition 2. *For every processes T_1, T_2 and P it holds $g(P, T_1) \cup g(P, T_2) = g(P, T_1 + T_2)$ and $g(P, T_1) \cup g(P, T_2) \subseteq g(P, T_1|T_2)$.*

Proof. Cleraly, $T_1, T_2 \prec T_1 + T_2, T_1|T_2$. By Proposition 1 we have $g(P, T_1) \cup g(P, T_2) \subseteq g(P, T_1 + T_2)$ and $g(P, T_1) \cup g(P, T_2) \subseteq g(P, T_1|T_2)$ Suppose that there exists $x, x \in g(P, T_1 + T_2)$ but $x \notin g(P, T_1) \cup g(P, T_2)$. This means neither test T_1 nor T_2 can reveal x what contradicts the assumption that x could be gained by nondeterministic choice of these two tests.

Proposition 3. *For every processes P_1, P_2 and T it holds $g(P_1, T) \cup g(P_2, T) = g(P_1 + P_2, T)$ and $g(P_1, T) \cup g(P_2, T) \supseteq g(P_1|P_2, T)$.*

Proof. By Lemma 4 and by the similar arguments which were used in the proof of the previous proposition.

By process's testing one can obtain information not only about actions which had to be performed but also about actions which could be excluded (they could not be performed). We start with a motivation example taken from [Gru11].

Example 3 (Access control process). Let Psw be a set of all possible passwords. Let us consider a simple access control process defined as follows (the set of high level action H_{Psw} consists of actions $h_w, w \in Psw$ and actions $\bar{l}_{\text{login}}$, $\bar{l}_{\text{access denied}}, l_w, w \in Psw$ are low level actions).

$$P = l_v.h_v.\bar{l}_{\text{login}}.Nil + \sum_{u \in Psw, u \neq v} l_u.h_v.\bar{l}_{\text{access denied}}.Nil$$

This process could represent, for example, an access to safe-deposit where no name of a bank client is required just a private key (or pin code - i.e. some password, in general). An attacker tries to guess the correct password. (S)he enters u what is modeled by performing low level action l_u ((s)he can see/observe what (s)he tries - a public action l_u could be "observed".) The guessed password (u) is compared with the correct one (v, represented by high level action h_v, which is unknown to the attacker). The attacker behaves as the test

$$T = \sum_{w \in V} \bar{l}_w.(l_{\text{login}}.Nil + l_{\text{access denied}}.Nil)$$

where $V \subseteq Psw$. If $|V|$ is much smaller then $|Psw|$ it is very unlikely that V contains a correct password, but even if V really does not contain the correct password, we can obtain some information about the correct one by test T since the correct one has to be from the reduced set $Psw \setminus V$. Note that we still have $g(P,T) = \emptyset$ and hence to describe the knowledge obtained by test T we need a new concept.

To obtain negative information on action performance we need to exploit strong testing (see Definition 5).

Definition 11. *Let T be a test and $M \subset A$. We say that CCS process P passes strong security test with respect to M (we will write $P \in sST_{T,M}$) if*

$$P \approx_{s,T,M} P/M.$$

Now we can define a set of actions which occurrence can be excluded by test T.

Definition 12. *Let T be a test and P process. We define a set of high level actions (denoted as $e(P,T)$) which occurrence could be excluded by test T as*

$$e(P,T) = \bigcap_{x \in H} \{x | P \in ST_{s,T,H\setminus\{x\}} \wedge P \notin ST_{s,T,H}\}.$$

Note that for sets of excluded action one can formulate similar properties as for gained ones. Concepts of the gained and excluded sets of private actions are complementary as it is stated in the following proposition. Roughly speaking, only systems for which both the sets - gained and excluded private actions are empty could be considered fully secure. Now we present some compositional results as well as relationship between sets of gained and excluded high level actions.

Proposition 4. *For every processes T_1, T_2 and P it holds $e(P,T_1) \cup e(P,T_2) = e(P, T_1 + T_2)$ and $e(P,T_1) \cup e(P,T_2) \subseteq e(P, T_1|T_2)$.*

Proof. The proof is based on the same arguments as the proof of Proposition 2.

Proposition 5. *For every processes P_1, P_2 and T it holds $e(P_1,T) \cup e(P_2,T) = e(P_1 + P_2, T)$.*

Proof. From Lemma 7 we know that if $T \preceq P_1$ and $T \preceq P_2$ then $T \preceq_{sM} P_1 + P_2$. The rest of the proof follows from the fact that every successor of $P_1 + P_2$ is a successor of P_1 or P_2.

Proposition 6. *For every process P and every test T it holds $g(P,T) \cap e(P,T) = \emptyset$ and $\emptyset \subseteq g(P,T) \cup e(P,T) \subseteq H$.*

Proof. Suppose that there exists h, $h \in g(P,T) \cap e(P,T)$ for some P and T. This high level action is excluded and gained by the test what is contradiction with the definition of these sets since something what is refused by testing cannot be accepted by strong testing (note that this does not hold vice versa). On the other side, we could find process P and test T such that no action is gained or excluded and that every private actions is either gained or excluded, respectively.

Let us return to Example 2. Security of process P (amount of information on P which can be obtained by test T) depends on the "size" of test T. For example, if T could try all possible passwords then P would not be secure with respect to such test. On the other hand, if it can try only a small fragment of passwords, then P is usually considered to be reasonable secure. Formally, we define the size of process as follows.

Definition 13. *The size of process P (denoted $|P|$) is defined as follows:*

$$\begin{aligned} |Nil| &= 0 \\ |x.P| &= 1 + |P| \\ |P + Q| &= |P| + |Q| \\ |P|Q| &= |P| + |Q| \\ |P \setminus M| &= 1 + |P| \\ |P[S]| &= 1 + |P| \\ |\mu X P| &= 1 + |P| \end{aligned}$$

Now we are ready to defines quantifications of levels of security.

Definition 14. *We define the level of security of P with respect to test T for gained (excluded) actions (denoted by LSg and LSe, respectively) as $LSg = ()|g(P,T)|.|H|)/|T|$ ($LSe = (|H|.|ge(P,T)|/|T|$).*

Note that thanks to Proposition 6 we have that $0 \leq |g(P,T)| + |e(P,T)| \leq |H|$.

Example 4. Let us return to Example 2. We have $LSe = |Psw|$, i.e. the level of security of P with respect to test T for excluded actions is given by the size of the set of all passwords.

The levels of security (Definition 14) strongly depend on the size of test T. We cannot replace this number with the size of "minimal" test which is bisimilar to T for two reasons. First, to find the minimal test in the equational class is undecidable in general (due to the universal power of CCS calculus). Moreover, the presented levels of security are based on a concrete process description which represents some realistic and practical approximation of the "theoretical" values.

6 Probabilistic noninterference

The security properties which were defined in the previous sections are qualitative ones. In Definition 14 we propose some quantification as a numerical expression of amount of information which can be obtained with respect to the size of H and T, but it is based on qualitative basis. By the test we can obtain information or not. In this section we extend this approach by directly employing probabilistic tests and testing.

First we need some preparatory work. Let P be a pCCS process and let $P \stackrel{x_1}{\rightarrow} P_1 \stackrel{x_2}{\rightarrow} P_2 \stackrel{x_3}{\rightarrow} \ldots \stackrel{x_n}{\rightarrow} P_n$, where $x_i \in Act \cup (0,1]$ for every $i, 1 \leq i \leq n$. The sequence $P.x_1.P_1.x_2 \ldots x_n.P_n$ will be called a finite computational path of P (path, for short), its label is a subsequence of $x_1 \ldots . x_n$ consisting of those elements which belong to Act i.e. $label(P.x_1.P_1.x_2 \ldots x_n.P_n) = x_1 \ldots . x_n|_{Act}$ and its probability is defined as a multiplication of all probabilities contained in it, i.e. $Prob(P.x_1.P_1.x_2 \ldots x_n.P_n) = 1 \times q_1 \times \ldots \times q_k$ where $x_1 \ldots . x_n|_{(0,1]} = q_1 \ldots g_k$. The multiset of finite paths of P will be denoted by $Path(P)$. For example, the path $(0.5.a.Nil \oplus 0.5.a.Nil).0.5.(a.Nil).a.(Nil)$ is contained in $Path(0.5.a.Nil \oplus 0.5.a.Nil)$ two times. There exist a few possible definitions of this multiset. For example, in [SL95] a technique of schedulers are used to resolve the nondeterminism and in [GSS95] all transitions are indexed and hence pathes can be distinguished by different indexes. In the former case, every scheduler defines (schedules) a particular computation path and hence two different schedulers determine different pathes, in the later case, the index records which transition was chosen in the case of several possibilities. The set of indexes for process P consists of sequences $i_1 \ldots i_k$ where $i_j \in \{0, \ldots, n\} \cup \{0, \ldots, n\} \times \{0, \ldots, n\}$ where n is the maximal cardinality of I for subterms of P of the form $\bigoplus_{i \in I} q_i.P_i$. An index records a way in which a computation path of P could be derived, i.e. it records which process was chosen in the case of several nondeterministic possibilities. If there is only one possible successor then transitions are indexed by 1 (i.e. corresponding $i_l = 1$). If transition $P \stackrel{x}{\rightarrow} P'$ is indexed by k (i.e. corresponding $i_l = k$) then transition $P + Q \stackrel{x}{\rightarrow} P'$ is indexed by $k.1$ and transition $Q + P \stackrel{x}{\rightarrow} P'$ is indexed by $k.2$. If transition $P_i \stackrel{x}{\rightarrow} P'$ is indexed by k then transition $\bigoplus_{i \in I} q_i.P_i \stackrel{x}{\rightarrow} P'$ is indexed by $k.i$, and if transitions $P \stackrel{x}{\rightarrow} P'$ and $Q \stackrel{x}{\rightarrow} Q'$ are indexed by k and l, respectively, then transitions of $P|Q$ have indexes from $\{(k,0),(0,l),(k,l)\}$ depending on which transition rule for parallel composition was applied. Every index defines at most one path and the set of all indexes defines the multisets of pathes $Path(P)$. Let $C, C \subseteq Path(P)$ be a finite multiset. We define $Pr(C) = \sum_{c \in C} Prob(c)$ if $C \neq \emptyset$ and $Pr(\emptyset) = 0$. Now we are ready to associate probabilities to relations $\stackrel{a}{\rightarrow}$ and $\stackrel{s}{\Rightarrow}_M$.

Definition 15. *Let $P \in pCCS$. We define $Pr(P, a, O) = Pr(C)$, where $C = \{c | label(c) = a, P \stackrel{a}{\rightarrow} P', P' \in O\}$ and $Pr_M(P, a, O) = Pr(C)$ where $C = \{c | label(c) = s, s|_L = a, s \in (\{a\} \cup M)*, P \stackrel{s}{\Rightarrow}_M P', P' \in O\}$.*

Probabilistic testing with a given accuracy ϵ is similar to non-probabilistic testing but it requires that probabilities with which the test and process perform required actions do not differ more then by ϵ.

Definition 16. *Let $M \subset A$. We say that process process P passes test T with respect to M with accuracy ϵ iff there exists relation $\Re$ called M -strong ϵ-simulation such that whenever $(T,R) \in \Re$ $Pr(T,a,O) \times Pr_M(P,a,O') \neq 0$, $|Pr(T,a,O) - Pr_M(P,a,O')| \leq \epsilon$ and $(O,O') \in pCCS \times pCCS/\Re$.*
We will denote the union of all M -strong simulations with accuracy ϵ as $\preceq_{sM\epsilon}$.

Finally we can define the strong test ϵ probabilistic equivalence.

Definition 17. *Let $M \subset A$. We say that process process P passes test T with respect to M with accuracy ϵ iff there exists M -strong ϵ-simulation such that whenever $(T,R) \in \Re$ $Pr(T,a,O) \times Pr_M(P,a,O') \neq 0$, $|Pr(T,a,O) - Pr_M(P,a,O')| \leq \epsilon$ and $(O,O') \in pCCS \times pCCS/\Re$.*
We will denote the union of all M -strong simulations with accuracy ϵ as $\preceq_{sM\epsilon}$.

Finally we can define the strong test ϵ probabilistic equivalence.

Definition 18. *We define strong test ϵ probabilistic equivalence with respect to test T for pCCS processes and set $M, M \subset A$ (we will denote it as $\approx_{s,T,M,\epsilon}$) as $P \approx_{s,T,M,\epsilon} Q$ iff $T \preceq_{sM\epsilon} P \leftrightarrow T \preceq_{sM\epsilon} Q$.*

Here we have simple observation with a straightforward proof.

Lemma 16. $\approx_{s,T,M,1} = \approx_{s,T,M}$.

As regards tests, we assume that they cannot perform the same action with two different probabilistic steps. Formally:

Definition 19. *We say that process P is probabilistically deterministic if for every $P' \in Succ(P)$ and every a there exists at most one transition $P' \xrightarrow{qa}$.*

Finally we are ready to define the probabilistic test with a given accuracy.

Definition 20. *Let T be a probabilistically deterministic test and $M \subset A$. We say that CCS process P passes security test with respect to M and accuracy ϵ if $P \approx_{T,M,\epsilon} P/M$. In this case we write $P \in ST_{T,M,\epsilon}$.*

We could extend Definitions 12 and 9 to probabilistic setting as well as Definition 14 of quantifications of levels of probabilistic security.

To do so we need more accurate measure for the probabilistic test size, which takes into account also a probabilistic distribution in the case of the choice operator. Basically, we multiply the sum of the sizes of processes P_i by entropy of $\oplus_{i \in I} q_i$ in the case of process $\oplus_{i \in I} q_i.P_i$.

Definition 21. *We define the size of probabilistic process P (denoted $|P|$) as for non-probabilistic one (see Definition 13) with*

$$|\oplus_{i \in I} q_i.P_i| = sum_{i \in I}|P_i| \times sum_{i \in I} q_i . \log_2 \frac{1}{q_i}.$$

Now, the level of security in the case of the probabilistic test can be defined in the same way as it was done for non-probabilistic one (see Definition 14).

Example 5. Let us return to Example 2. Corresponding robabilistic test T would be as follows

$$T = \oplus_{i \in V} q_i.\bar{l}_w.(l_{\text{login}}.Nil + l_{\text{access denied}}.Nil)$$

The maximal size of T is reached if all probabilities q_i are equal, i.e. there is no preliminary belief expressed by the test.

Note that the presented formalism cover all combinations of testings: none of processes is probabilistic, both processes are probabilistic, only test is probabilistic and only tested process is probabilistic. Each testing has its meaning depending on security property we want to check.

7 Conclusions

We have presented several security concepts based on an information flow obtained by testing of processes by means of public actions. They express process security, sets of private actions which have to be performed (the gained sets) or the set of private actions which could be excluded by a given test. The notion of excluded actions can be used for a reduction of a space of possible private actions and if the reduction is significant then it really threatens systems security. We have presented also some compostional properties which are useful in the case of bottom-up system design.

Moreover, our formalism includes also probabilistic tests of probabilistic processes. In this way we can capture also cases when the membership to the sets of gained and excluded actions is not certain but only very likely what is the information, which could help intruder significantly. Note that without "quantification" of the set membership, we can consider a system to be secure despite the fact that a successful attack could be possible. We have also defined the size of the test and how to relate it to the size of obtained private information on processes. We consider the possibility of such quantification as one of the most important advantages of the presented testing approach.

References

[CHM07] Clark D., S. Hunt and P. Malacaria: A Static Analysis for Quantifying the Information Flow in a Simple Imperative Programming Language. The Journal of Computer Security, 15(3). 2007.

[CMS09] Clarkson, M.R., A.C. Myers, F.B. Schneider: Quantifying Information Flow with Beliefs. Journal of Computer Security, to appear, 2009.

[NH84] De Nicola R. and M. C. B. Hennessy: Testing Equivalences for Processes, Theoretical Computer Science, 34, 1984

[FGM00] Focardi, R., R. Gorrieri, and F. Martinelli: Information flow analysis in a discrete-time process algebra. Proc. 13^{th} Computer Security Foundation Workshop, IEEE Computer Society Press, 2000.

[GSS95] Glabbeek R. J. van, S. A. Smolka and B. Steffen: Reactive, Generative and Stratified Models of Probabilistic Processes Inf. Comput. 121(1): 59-80, 1995

[GM82] Goguen J.A. and J. Meseguer: Security Policies and Security Models. Proc. of IEEE Symposium on Security and Privacy, 1982.

[Gru11] Gruska D.P.: Gained and Excluded Private Actions by Process Observations. To apear in Fundamenta Informaticae, 2011.

[Gru09] Gruska D.P.: Quantifying Security for Timed Process Algebras, Fundamenta Informaticae, vol. 93, Numbers 1-3, 2009.

[Gru08] Gruska D.P.: Probabilistic Information Flow Security. Fundamenta Informaticae, vol. 85, Numbers 1-4, 2008.

[Gru07] Gruska D.P.: Observation Based System Security. Fundamenta Informaticae, vol. 79, Numbers 3-4, 2007.

[HJ90] Hansson, H. a B. Jonsson: A Calculus for Communicating Systems with Time and Probabilities. In Proceedings of 11th IEEE Real - Time Systems Symposium, Orlando, 1990.

[LN04] López N. and Núñez: An Overview of Probabilistic Process Algebras and their Equivalences. In Validation of Stochastic Systems, LNCS 2925, Springer-Verlag, Berlin, 2004

[Mil89] Milner, R.: *Communication and concurrency.* Prentice-Hall International, New York,1989.

[SL95] Segala R. and N. Lynch: Probabilistic Simulations for Probabilistic Processes. Nord. J. Comput. 2(2): 250-273, 1995

[YL92] Yi W. and K.G. Larsen: Testing Probabilistic and Nondeterministic Processes. Proceeding Proceedings of the IFIP TC6 - WG6.1, 1992.