Test based security *

Damas P. GRUSKA

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

Abstract. Different techniques for expressing an amount of information on secrete data which can be obtained by a process testing are presented. They are based on (probalistic) process algebra and (probabilistic) testing. Besides tested noninterference they express certainty about 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 the size of the test.

Keywords: probabilistic process algebras, information flow, security

1 Introduction

We propose formalisms for analysis of systems of various nature specified by process algebras. They allow 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 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, security property called Bisimulation Strong Nondeterministic Non-Interference requires that it cannot be distinguished (by bisimulation) between forbidding and hiding of private actions. In our approach we exploit this idea but we somehow weaken it by requiring more realistic assumption that forbidding and hiding of private actions cannot be distinguished by a (finite state) test, i.e. we exploit a kind of testing equivalence. In [Gru11] we have studied information flows by defining two sets - the set of private actions which execution is guaranteed by a given observation of public actions and the set of actions which execution is excluded by a given observation. Here we apply this idea and we define sets of gained and excluded actions by a given test instead of an observation.

Traditional security properties could be criticized for being either too restrictive or too benevolent.

Even in the case testing scenario, a standard access control process to be insecure since there is always some (even very small) information flow for an attacker which tries to learn a password - at least (s)he can learn what is not

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

the correct password. On the other side, it can happen that the set of excluded or gained (or both) sets (of possible passwords) are empty but a membership of some possible password to some of them is very likely since not certain. There are several ways how 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 algebra. Another possibility is to exploit probabilistic theory as it was used for process algebras in [Gru09]. Resulting techniques lead to quantifications of how many bits of private information can leak or how probable is that an intruder can learn some secrete property on processes. Here we exploit probabilistic process algebras (either for a test, tested process, or both) to express probabilities of passing the test or probabilities of membership of elements in sets of gained or excluded actions. Moreover, we relate obtained information to the size of the test. Presented testing approach is strictly stronger then that of [Gru11], which is based on traces. 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 can jeopardize this security. Hence the resulting level of security gives us relevant information on real (practical) system security.

The paper is organized as follows. In Section 2 we describe our testing scenario. In Sections 4 and 5 we define test based noninterference and probabilistic noninterference, respectively.

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. Closed terms which are guarded (each occurrence of X is within some subexpression $u.A$ 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 processes by given labeled transition systems. The set of terms represents a set of states, labels are actions from Act (see [Mil89]).

The transition relation $\rightarrow$ is a subset of CCS $\times$ Act $\times$ 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 $\bigoplus_{i \in I} q_i.P_i, q_i$ 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). Hence 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.

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

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

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 process 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 (except 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 $\Rightarrow_h$. 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 all actions are replaced by its complementary action if it exists and left as they are otherwise. For example, if $s = a.\bar{b}.\tau.c$ we have $\bar{s} = \bar{a}.b.\tau.\bar{c}$.

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 $Tr_{wM}(P) = Tr_{wM}(Q)$. Again we will write $\approx_w$ instead of $\approx_{w\emptyset}$.

Definition 2. Let (CCS, $Act, \rightarrow$) be a labelled 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 strong 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 next section. 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 a resulting process again pass the resulting test. By test T we will consider process which does not contain τ action 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 $(T, R) \in \Re$ 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 define equivalence between processes if they cannot be distinguished by test T.

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

$$P \approx_{T,M} Q \text{ iff } T \preceq_M P \leftrightarrow T \preceq_M Q.$$

We will need also stronger variant of testing, which require that every response to the test 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 $(T, R) \in \Re$ then for every $a \in A$ if $T \xrightarrow{a} T'$ then 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 $\preceq_{sM}$.

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*

$$P \approx_{s,T,M} Q \text{ iff } T \preceq_{sM} P \leftrightarrow T \preceq_{sM} Q.$$

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

To define non-interference for process algebra setting 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 $\bar{h} \in H, \bar{l} \in L$. To denote sequences of public actions, i.e sequences consisting of actions from L and sequences of private actions from H, we will use notation $\tilde{l}, \tilde{l'}, \ldots$ for sequences from $L^\star$ and $\tilde{h}, \tilde{h'}, \ldots$ for sequences from $H^\star$, respectively. The set of actions could be divided to more than into two subsets, what would correspond into 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 hiding operator $P/M, M \subseteq A$, for which it holds 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 $a \in M \cup \bar{M}$. Formal definition of BSNNI follows.

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

Inspired by BSNNI property we define our security tests. Process passes a test if tester cannot detect communication via actions from M of tested process (see Fig 1., here we assume that $Succ(Q) \subseteq M$).

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

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

Fig. 1. Testing scenario

Lemma 1. *Let $P \in BSNNI$. Then we have $P \in ST_{T,H}$ for every test T.*

Proof. Sketch. Let $P \in BSNNI$ then $P \setminus H$ and P/H cannot be distinguished by weak bisimulation and hence cannot be distinguished by test equivalence with respect to any test and the set of high level actions.

Lemma 2. *Let $P \notin BSNNI$. Then there exists finite state test T such that $P \notin ST_{T,H}$.*

Proof. Sketch. If $P \notin BSNNI$. This means that $P \setminus H \not\approx P/H$, i.e. if one tries to establish corresponding bisimulation than this has to fail. The required test would mimic that.

In similar way we can prove the inverse of the following lemma.

Lemma 3. *There does not exist finite state test T such that $P \in ST_{T,H}$ would imply $P \in BSNNI$.*

4.1 Gained and excluded private actions

First we define a set of private actions which occurrence can be learned by a test T.

Definition 9. *We say that test T can gain information on occurrence of high level actions of CCS process P (we will denote their set as $g(P,T)$) if*

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

According to Definition 9 set $g(P,T)$ represents private actions which performance could be deduced by test T. By stronger test we can learn more, as it is stated by the following proposition.

Proposition 1. *Let we have two tests T, T' such that $T' \prec T$. Then we have $g(P,T') \subseteq g(P,T)$ for every process P.*

Proof. Main idea. Let $T' \prec T$ then every action performed by T can be simulated by T'. Hence every action which occurrence could be detected by T can be detected also by T'.

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

Definition 10. *Let $P \in CCS$ and $\tilde{l} \in Tr_{wH}(P)$. Then the occurrence of the set of private action which can be gained about P by public observing $\tilde{l}$ is defined as follows:*

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

As it is clear from the following lemma and example, here presented notion is strictly stronger.

Lemma 4. *Let $\bar{\tilde{l}} \in Tr_{wH}(P)$. Then it holds $g(P, \tilde{l}.Nil) = g(P, \bar{\tilde{l}})$.*

Proof. Sketch. In case of simple test $T, T = \tilde{l}.Nil$ we can obtain exactly the same information as by observing $\bar{\tilde{l}}$.

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

Proof. Let $T' = \bar{\tilde{l}}.Nil$. Then we have from Propositions 1 and Lemma 4 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 $\tilde{l}$.

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 gain by test T some information about the correct one - 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.

Proposition 2. *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. Main idea. 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 way how these sets are defined. 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 (how much information on P can be obtained by test T) depends on "size" of T. For example, if T could try all possible passwords then P would be not secure with respect to such test. On the other hand, if it can try only small fragment of passwords, then P is usually considered to be reasonable secure. Formally we define size of process as follows.

Definition 13. *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. *Level of secrecy of P with respect test T for gained (excluded) actions (denoted by LSg and LSe, respectively), can be expressed as $LSg = (|g(P,T)|.|H|)/|T|$ ($LSe = (|H|.|ge(P,T)|/|T|$).*

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

5 Probabilistic noninterference

Above mentioned properties are qualitative ones. In Definition 14 we propose some quantification as a numerical expression how much information can be

obtained with respect to size of H and T, but it is based on qualitative basis. By 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 $P \stackrel{x_1}{\to} P_1 \stackrel{x_2}{\to} P_2 \stackrel{x_3}{\to} \ldots \stackrel{x_n}{\to} P_n$, 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 pathes 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 techniques how to define 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 how a computation path of P could be derived, i.e. it records which process was chosen in case of several nondeterministic possibilities. If there is only one possible successor transitions are indexed by 1 (i.e. corresponding $i_l = 1$) If transition $P \stackrel{x}{\to} P'$ is indexed by k (i.e. corresponding $i_l = k$) then transition $P + Q \stackrel{x}{\to} P'$ is indexed by $k.1$ and transition $Q + P \stackrel{x}{\to} P'$ is indexed by $k.2$. If transition $P_i \stackrel{x}{\to} P'$ is indexed by k then transition $\bigoplus_{i \in I} q_i.P_i \stackrel{x}{\to} P'$ is indexed by $k.i$, and if transitions $P \stackrel{x}{\to} P'$ and $Q \stackrel{x}{\to} Q'$ 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 $\stackrel{a}{\to}$ and $\stackrel{s}{\Rightarrow}_M$.

Definition 15. *Let $P \in pCCS$. We define $Pr(P, a, O) = Pr(C)$, where $C = \{c | label(c) = a, P \stackrel{a}{\to} P', P' \in O\}$ 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 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 strong test ϵ probabilistic equivalence.

Definition 17. *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*

$$P \approx_{s,T,M,\epsilon} Q \text{ iff } T \preceq_{sM\epsilon} P \leftrightarrow T \preceq_{sM\epsilon} Q.$$

Here we have simple observation with a straightforward proof.

Lemma 5. $\approx_{s,T,M,1}=\approx_{s,T,M}$.

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

Definition 18. *We say that process P is probabilistically deterministic for every $P' \in Succ(P)$ it holds that there is at most one transition $P' \xrightarrow{qa}$ for every a.*

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

Definition 19. *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 probabilistic test size, which takes into account also probabilistic distribution in case of choice operator. Basically, we multiply sum of sizes of process P_i by entropy of $\oplus_{i\in I} q_i$ in case of process $\oplus_{i\in I} q_i.P_i$.

Definition 20. *Size of process probabilistic process P (denoted $|P|$) is defined 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, level of secrecy in case of 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. The probabilistic test T would be then

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

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 would like to check.

6 Conclusions

We have presented several security concepts based on an information flow observed by testing 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.

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 information, which could help intruder very 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 a way how to measure size of a test and how to relate it to the size of obtained private information on processes. We consider such quantification as one of the most important advantages of 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.

[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