Opacity Testing *

Damas P. Gruska

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

Abstract. Opacity testing is formalized and studied. We specify opacity testers as well as tested systems by (timed) process algebras. We model various testers according to how sophisticated observations of tested system they can make and which kind of conclusions they can obtain. We use this technique to define several realistic security properties. The properties are studied and compared with other security concepts.

Keywords: opacity, process algebras, information flow, security

1 Introduction

Several formulations of system security can be found in the literature. Many of them are based on non-interference (see [GM82]) which assumes an absence of any information flow between private and public systems activities. More precisely, systems are considered to be secure if from observations of their public activities no information about private activities can be deduced. This approach has found many reformulations for different formalisms, computational models and nature or "quality" of observations.

One of the most general notion is opacity (see [BKR04,BKMR06]) and many security properties can be viewed as its special cases (see, for example, [Gru07]). A predicate is opaque if for any trace of a system for which it holds there exists another trace for which it does not hold and both traces are indistinguishable for an observer. Opacity is widely studied also in process algebras framework. Here, as well as later in this paper, we mention those ones which are close to the the presented work. For example, in [Gru07,Gru12] opacity for very simple observations is studied for timed process algebra. In [Gru09] a quantification of opacity by means of the information theory is studied. In [Gru10,Gru12a] we defined security properties which could be described by specific relations on contexts. In general, opacity is an undecidable property even for very simple observation functions or predicates. On the other side, opacity is based on traces and hence inadequate for any finer "attacker" who is capable not only observe traces but also interact with systems.

The aim of this paper is twofold. On the one side, we weaken opacity by modeling both predicate and observations by processes (particularly, finite state

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

processes) and hence we obtain (polynomial time) decidable properties. On the other side, we strength opacity by defining simulation opacity which is not restricted to trace observations and which is stronger than opacity. While opacity of predicate is defined for a given process (and an observation function), simulation opacity requires (roygly speaking) that it is opaque also for every successor of the process. Moreover, our formalism of timed process algebra, allows us to express various types of timed attacks.

The paper is organized as follows. In Section 2 we describe the timed process algebra TPA which will be used as a basic formalism. In Section 3 we present opacity and in the next section simulation opacity is defined and studied.

2 Timed Process Algebra

In this section we define Timed Process Algebra, TPA for short. TPA is based on Milner's CCS but the special time action t which expresses elapsing of (discrete) time is added. The presented language is a slight simplification of Timed Security Process Algebra introduced in [FGM00]. We omit an explicit idling operator ι used in tSPA and instead of this we allow implicit idling of processes. Hence processes can perform either "enforced idling" by performing t actions which are explicitly expressed in their descriptions or "voluntary idling". But in the both cases internal communications have priority to action t in the case of the parallel operator. Moreover we do not divide actions into private and public ones as it is in tSPA. TPA differs also from the tCryptoSPA (see [GM04]). TPA does not use value passing and strictly preserves *time determinacy* in case of choice operator $+$ what is not the case of tCryptoSPA.

To define the language TPA, we first assume a set of atomic action symbols A not containing symbols τ and t, and such that for every $a \in A$ there exists $\overline{a} \in A$ and $\overline{\overline{a}} = a$. We define $Act = A \cup \{\tau\}, Actt = Act \cup \{t\}$. We assume that $a, b, \ldots$ range over A, $u, v, \ldots$ range over Act, and $x, y \ldots$ range over $Actt$. Assume the signature $\Sigma = \bigcup_{n\in\{0,1,2\}} \Sigma_n$, where

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

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, $S : Actt \to Actt$ are such that $\overline{S(a)} = S(\bar{a})$ for $a \in A, S(\tau) = \tau$ and $S(t) = t$.

The set of TPA 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 TPA terms, $\mu X-$ is the binding construct, $op \in \Sigma$.

The set of CCS terms consists of TPA terms without t action. We will use an usual definition of opened and closed terms where μX is the only binding operator. Closed terms which are t-guarded (each occurrence of X is within some subexpression $t.A$ i.e. between any two t actions only finitely many non timed actions can be performed) are called TPA processes. Note that Nil will be often omitted from processes descriptions and hence, for example, instead of $a.b.Nil$ we will write just $a.b$.

We give a structural operational semantics of terms by means of labeled transition systems. The set of terms represents a set of states, labels are actions from $Actt$. The transition relation $\rightarrow$ is a subset of TPA $\times$ $Actt$ $\times$ TPA. 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 $P \xrightarrow{x} P'$. We define the transition relation as the least relation satisfying the inference rules for CCS plus the following inference rules:

$$\frac{}{Nil \xrightarrow{t} Nil} \quad A1 \qquad \frac{}{u.P \xrightarrow{t} u.P} \quad A2$$

$$\frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q', P \mid Q \not\xrightarrow{\tau}}{P \mid Q \xrightarrow{t} P' \mid Q'} \; Pa \qquad \frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q'}{P + Q \xrightarrow{t} P' + Q'} \; S$$

Here we mention the rules that are new with respect to CCS. Axioms $A1, A2$ allow arbitrary idling. Concurrent processes can idle only if there is no possibility of an internal communication (Pa). A run of time is deterministic (S). In the definition of the labeled transition system we have used negative premises (see Pa). In general this may lead to problems, for example with consistency of the defined system. We avoid these dangers by making derivations of τ independent of derivations of t. For an explanation and details see [Gro90]. Regarding behavioral relations we will work with the timed version of weak trace equivalence. Note that here we will use also a concept of observations which contain complete information which includes also τ actions and not just actions from A and t action as it is in [FGM00]. For $s = x_1.x_2.\ldots.x_n, x_i \in Actt$ we write $P \xrightarrow{s}$ instead of $P \xrightarrow{x_1}\xrightarrow{x_2} \ldots \xrightarrow{x_n}$ and we say that s is a trace of P. By ϵ we will denote the empty sequence of actions, by $Succ(P)$ we will denote the set of all successors of P. If the set $Succ(P)$ is finite we say that P is finite state.

Let $s \in Actt^\star$. By $|s|$ we will denote the length of s i.e. a number of action contained in s. By $s|_B$ we will denote the sequence obtained from s by removing all actions not belonging to B. For example, $|s|_{\{t\}}|$ denote a number of occurrences of t in s, i.e. time length of s.

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 action). Formally, we will write $P \stackrel{x}{\Rightarrow}_M P'$ for $M \subseteq A$ iff $P \xrightarrow{s_1}\xrightarrow{x}\xrightarrow{s_2} P'$ for $s_1, s_2 \in (M \cup \{\tau\})^\star$

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 $P \stackrel{x}{\Rightarrow}_M P'$. We will write $P \stackrel{\widehat{x}}{\Rightarrow}_M P'$ instead of $P \stackrel{\epsilon}{\Rightarrow}_M P'$ if $x \in M$.

We conclude this section with definitions of variants of weak simulation and weak bisimulation.

Definition 1. Let (TOA, $Actt, \rightarrow$) be a labelled transition system (LTS). A relation $\Re \subseteq \text{CCS} \times \text{CCS}$ is called a *weak M-bisimulation* if it is symmetric and it satisfies the following condition: if $(P, Q) \in \Re$ and $P \stackrel{x}{\rightarrow} P', x \in Actt$ then there exists a process Q' such that $Q \stackrel{\widehat{x}}{\Rightarrow}_M Q'$ and $(P', Q') \in \Re$. Two processes P, Q are *M-bisimilar*, abbreviated $P \approx_M Q$, if there exists a strong bisimulation relating P and Q. If it is not required that relation $\Re$ is symmetric we call it M-simulation and we say that process P simulates process Q, abbreviated $P \prec_M Q$, if there exists a simulation relating P and Q.

We will write $\approx$ and $\prec$ instead of $\approx_M$ and $\prec_M$, respectively, if $M = \emptyset$.

3 Opacity

To formalize an information flow we do not divide actions into public and private ones at the system description level, as it is done for example in [GM04,BG04], but we use a more general concept of observation and opacity. This concept was exploited in [BKR04] and [BKMR06] in a framework of Petri Nets and transition systems, respectively.

First we define observation function on sequences from $Actt^\star$.

Definition 2 (Observation). *Let Θ be a set of elements called observables. Any function $\mathcal{O} : Actt^\star \rightarrow \Theta^\star$ is an observation function. It is called static /dynamic /orwellian / m-orwellian ($m \geq 1$) if the following conditions hold respectively (below we assume $w = x_1 \ldots x_n$):*

- *static if there is a mapping $\mathcal{O}' : Actt \rightarrow \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1) \ldots \mathcal{O}'(x_n)$,*
- *dynamic if there is a mapping $\mathcal{O}' : Actt^\star \rightarrow \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1).\mathcal{O}'(x_1.x_2) \ldots \mathcal{O}'(x_1 \ldots x_n)$,*
- *orwellian if there is a mapping $\mathcal{O}' : Actt \times Actt^\star \rightarrow \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1, w).\mathcal{O}'(x_2, w) \ldots \mathcal{O}'(x_n, w)$,*
- *m-orwellian if there is a mapping $\mathcal{O}' : Actt \times Actt^\star \rightarrow \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1, w_1).\mathcal{O}'(x_2, w_2) \ldots \mathcal{O}'(x_n, w_n)$ where $w_i = x_{max\{1,i-m+1\}}.x_{max\{1,i-m+1\}+1} \ldots x_{min\{n,i+m-1\}}$.*

In the case of the static observation function each action is observed independently from its context. In the case of the dynamic observation function an observation of an action depends on the previous ones, in the case of the orwellian and m-orwellian observation function an observation of an action depends on the all and on m previous actions in the sequence, respectively. The

static observation function is the special case of m-orwellian one for $m = 1$. Note that from the practical point of view the m-orwellian observation functions are the most interesting ones. An observation expresses what an observer - eavesdropper can see from a system behavior and we will alternatively use both the terms (observation - observer) with the same meaning.

Now suppose that we have some security property. This might be an execution of one or more classified actions, an execution of actions in a particular classified order which should be kept hidden, etc. Suppose that this property is expressed by predicate ϕ over process traces. We would like to know whether an observer can deduce the validity of the property ϕ just by observing sequences of actions from $Actt^\star$ performed by given process.

The observer cannot deduce the validity of ϕ if there are two traces $w, w' \in Actt^\star$ such that $\phi(w), \neg\phi(w')$ and the traces cannot be distinguished by the observer i.e. $\mathcal{O}(w) = \mathcal{O}(w')$. We formalize this concept by opacity.

Definition 3 (Opacity). *Given process P, a predicate ϕ over $Actt^\star$ is opaque w.r.t. the observation function $\mathcal{O}$ if for every sequence w, $w \in Tr(P)$ such that $\phi(w)$ holds and $\mathcal{O}(w) \neq \epsilon$, there exists a sequence $w', w' \in Tr(P)$ such that $\neg\phi(w')$ holds and $\mathcal{O}(w) = \mathcal{O}(w')$. The set of processes for which the predicate ϕ is opaque with respect to $\mathcal{O}$ will be denoted by $Op^\phi_\mathcal{O}$.*

The definition of opacity (see Definition 3) of predicate ϕ is asymmetric in the sense that if $\phi(w)$ does not hold than it is not required that there exists another trace for which it holds (in general $Op^\phi_\mathcal{O} \neq Op^{\neg\phi}_\mathcal{O}$). This means that opacity says something to an intruder which tries to detect only validity of ϕ (if it is opaque, than validity cannot be detected) but not its non-validity i.e. it says nothing about predicate $\neg\phi$. Hence we define strong variant of opacity.

Definition 4 (Strong Opacity). *Given process P, a predicate ϕ over $Actt^\star$ is strongly opaque w.r.t. the observation function $\mathcal{O}$ if for every sequence w, $w \in Tr(P)$ such that $\phi(w)$ holds and $\mathcal{O}(w) \neq \epsilon$, there exists a sequence $w', w' \in Tr(P)$ such that $\neg\phi(w')$ holds and $\mathcal{O}(w) = \mathcal{O}(w')$. Moreover, for for every sequence w, $w \in Tr(P)$ such that $\neg\phi(w)$ holds and $\mathcal{O}(w) \neq \epsilon$, there exists a sequence $w', w' \in Tr(P)$ such that $\phi(w')$ holds and $\mathcal{O}(w) = \mathcal{O}(w')$. The set of processes for which the predicate ϕ is opaque with respect to $\mathcal{O}$ will be denoted by $sOp^\phi_\mathcal{O}$.*

Lemma 1. *$sOp^\phi_\mathcal{O} \subseteq Op^\phi_\mathcal{O}$ for every ϕ and $\mathcal{O}$. Moreover, there exist ϕ and $\mathcal{O}$ such that $sOp^\phi_\mathcal{O} \subset Op^\phi_\mathcal{O}$.*

Proof. Main idea. Let $P \in sOp^\phi_\mathcal{O}$. Then for every trace of P for which ϕ holds there exists a trace indistinguishable by observation function $\mathcal{O}$ for which ϕ does not hold and hence $P \in Op^\phi_\mathcal{O}$. Let us consider process $P = h.l.Nil + l.Nil + l'.Nil$ and let ϕ holds for tracecs which contain action h and observation function $\mathcal{O}$ which hide h action. Then we have $P \in Op^\phi_\mathcal{O}$ but $P \notin sOp^\phi_\mathcal{O}$ and hence the inclusion is proper, i.e $sOp^\phi_\mathcal{O} \subset Op^\phi_\mathcal{O}$ for such ϕ and $\mathcal{O}$.

4 Simulation Opacity

We start with a motivation example. Let us consider process $P = l.h.l'.Nil + l.(h.l'.Nil + l'.Nil)$, an observation function which does not see action h and a predicate which holds for sequences containing h action. It is easy to check that this predicate is opaque in this setting. That means than an attacker which can observe traces of P cannot deduce whether action h has occurred or nor. On the other side for a "simulation attacker" i.e. the attacker which can not only observe traces but can interact with systems, the predicate is not "opaque" anymore. This is a natural consequence of simulation being more powerful then just a trace inclusion. Now we will extend the notion of opacity to reflect more powerful attackers than those ones which just observe traces or alternatively, predicate should be opaque not only for a given process P but also for every its successor.

Definition 5 (Simulation Opacity). *Given a set of processes $\Re$, predicate ϕ over $Actt^\star$ is simulation opaque for $\Re$ w.r.t. the observation function $\mathcal{O}$ if for every $P \in \Re$ if $P \xrightarrow{s} P'$ for such s that $\phi(s)$ holds and $o(s) \neq \epsilon$ then there exists s' such that $\neg\phi(s')$ holds, $\mathcal{O}(s) = \mathcal{O}(s')$ and $P \xrightarrow{s'}$ and moreover $P' \in \Re$. Predicate ϕ is simulation opaque for process P with respect to $\mathcal{O}$ (denoted $P \in SOp^{\phi}_{\mathcal{O}}$) if $P \in \Re$ for some simulation opaque $\Re$ with respect to ϕ and $\mathcal{O}$.*

Now let us return to process P and the predicate and the observation function from the beginning of this section. Now we can check that P is not simulation opaque in this setting. This is also the proof that an inclusion from the next proposition is proper.

Proposition 1. *$SOp^{\phi}_{\mathcal{O}} \subseteq Op^{\phi}_{\mathcal{O}}$ for every ϕ and $\mathcal{O}$. Moreover, there exist ϕ and $\mathcal{O}$ such that $SOp^{\phi}_{\mathcal{O}} \subset Op^{\phi}_{\mathcal{O}}$.*

Proof. The main idea. Let $P \in SOp^{\phi}_{\mathcal{O}}$. Then we have that for every trace s of P for which ϕ holds there exists another trace s' for which ϕ does not hold and both traces cannot be distinguished by $\mathcal{O}$. Hence $P \in Op^{\phi}_{\mathcal{O}}$. The example from of this section we see that the inclusion can be proper.

The simulation opacity is defined for arbitrary predicates and observation functions. Now we we will reformulate it for those ones which can be expressed by process algebras. Now we will model simulation opacity in a process algebra setting. Suppose that $Actt \cap \Theta = \{t\}$ and hence we extend the set of actions A by Θ. We combine a process which checks validity of ϕ with a process which computes observation function $\mathcal{O}$ into two process O_ϕ and $O_{\neg\phi}$. Now we define process O_ϕ.

Definition 6. *Process O_ϕ is called process definition of predicate ϕ and observation function $\mathcal{O}$ over sequences of actions if for every P it holds $(P|O_\phi) \setminus A \stackrel{o}{\Rightarrow} (P'|O_\phi) \setminus A$ iff $P \xrightarrow{s} P'$ such that $\phi(s)$ and $\mathcal{O}(s) = o$.*

Note that we expect that process O_ϕ makes some computation resulting on observable o and then it returns to the initial state (actually, to be more precise, we should write to the process bisimilar with it). Now we will define simulation opacity with respect to O_ϕ and $O_{\neg\phi}$ (see Fig. 1). Its definition is a reformulation of Definition 5 in process algebra setting.

Fig. 1. Testing scenario

Definition 7. *We say that process P is simulation opaque with respect to O_ϕ and $O_{\neg\phi}$ (denoted $P \in SO(O_\phi, O_{\neg\phi})$ iff $(P|O_\phi) \setminus A \prec (P|O_{\neg\phi}) \setminus A$.*

In fact, from the following proposition we see that both types of simulation opacity coincide for those predicated and observation functions which can be expressed by processes.

Proposition 2. *Let O_ϕ and $O_{\neg\phi}$ are process definitions of observation function $\mathcal{O}$ and predicates ϕ and $\neg\phi$, respectively. Then $SOp^\phi_\mathcal{O} = SO(O_\phi, O_{\neg\phi})$.*

Proof. The main idea. Process definition O_ϕ and $O_{\neg\phi}$ mimic both observations and predicates validity (see Definition 7). Moreover, the simulation $\prec$ reflects the fact that after each "step" the resulting process is again opaque and hence similation opaque.

Many trace based security properties can be viewed as special cases of opacity (see for example [Gru07]) but not those ones which are based on more powerful equivalences. Now we show how we can express by simulation opacity a stronger security property. We define an absence-of-information-flow property - Bisimulation Strong Nondeterministic Non-Interference (BSNNI, for short, see [FGM00]). Suppose that all actions are divided in two groups, namely public (low level) actions L and private. 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 8. *Let $P \in TPA$. We say that P has BSNNI property, and we write $P \in BSNNI$ iff $P \setminus H \approx P/H$.*

Example 1. Let $\phi(s)$ holds iff s contains actions from H and let $\theta = \{o_x | x \in L$, $\mathcal{O}(s) = o$ such that $o = o_{l_1} \dots o_{l_n}$ where $s|_L = l_1.l_2 \dots l_n$.

Then the following process

$$O_\phi = \mu X.(\sum_{x\in L} x.o_{\bar{x}}.X + \sum_{x\in H} x.\mu Y.(\sum_{x\in L} x.o_{\bar{x}}.Y + \sum_{x\in H} x.Y))$$

is the process definition of predicate ϕ and observation function $\mathcal{O}$.

Moreover process

$$O_{\neg\phi} = \mu X.(\sum_{x\in L} x.o_x.X)$$

is the process definition of predicate $\neg\phi$ and observation function $\mathcal{O}$.

Proposition 3. *$P \in BSNNI$ iff $P \in SO(O_\phi, O_{\neg\phi})$ for $O_\phi, O_{\neg\phi}$ defined in the previous example.*

Proof. Sketch. Process O_ϕ outputs o_x for every low level action which can be performed by P and switches to "accepting" state after the first high level action occurs. Similarly for $O_{\neg\phi}$. In definition of BSNNI the weak bisimulation is exploited but clearly, everything which can be performed by $P \setminus H$ can be performed by P/H and hence no more then simulation is needed.

Now we can return to the strong opacity. First we define its simulation version.

Definition 9 (Strong Simulation Opacity). *Given a set of processes $\Re$, predicate ϕ over $Actt^\star$ is strongly simulation opaque for $\Re$ w.r.t. the observation function $\mathcal{O}$ if for every $P \in \Re$ if $P \xrightarrow{s} P'$ for such s that $\phi(s)$ holds and $o(s) \neq \epsilon$ then there exists s' such that $\neg\phi(s')$ holds , $\mathcal{O}(s) = \mathcal{O}(s')$ and $P \xrightarrow{s'}$, and $P' \in \Re$ and, moreover, $P \xrightarrow{s} P''$ for such s that $\neg\phi(s)$ holds and $o(s) \neq \epsilon$ then there exists s' such that $\phi(s')$ holds , $\mathcal{O}(s) = \mathcal{O}(s')$ and $P \xrightarrow{s'}$, and $P'' \in \Re$. Predicate ϕ is strongly simulation opaque for process P with respect to $\mathcal{O}$ (denoted $P \in sSOp^\phi_\mathcal{O}$) if $P \in \Re$ for some strongly simulation opaque $\Re$ with respect to ϕ and $\mathcal{O}$.*

Similarly to the opacity, its stronger version is really different as it is stated by the following proposition.

Proposition 4. *$sSOp^\phi_\mathcal{O} \subseteq SOp^\phi_\mathcal{O}$ for every ϕ and $\mathcal{O}$. Moreover, there exist ϕ and $\mathcal{O}$ such that $sSOp^\phi_\mathcal{O} \subset SOp^\phi_\mathcal{O}$.*

Proof. The proof is just a variation of the proof of Proposition 1.

Definition 10. *We say that process P is strongly simulation opaque with respect to O_ϕ and $O_{\neg\phi}$ (denoted $P \in sSO(O_\phi, O_{\neg\phi})$ iff $(P|O_\phi) \setminus A \approx (P|O_{\neg\phi}) \setminus A$.*

Proposition 5. *Let O_ϕ and $O_{\neg\phi}$ are process definitions of observation function $\mathcal{O}$ and predicates ϕ and $\neg\phi$, respectively. Then $sSOp^\phi_\mathcal{O} = sSO(O_\phi, O_{\neg\phi})$.*

Proof. Again, the proof is similar as the proof of Proposition 2.

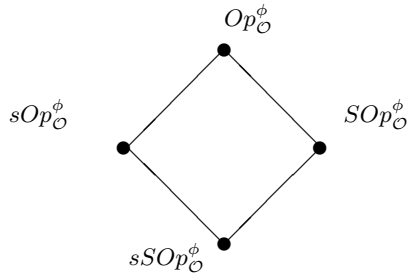

Fig. 2.

To complete a relationship between proposed opacity concepts we have the following proposition.

Proposition 6. *The relation between proposed opacities is depicted on Fig. 2.*

Proof. First we prove that $sSOp^{\phi}_{\mathcal{O}} \subset sOp^{\phi}_{\mathcal{O}}$. Let $P \in sSOp^{\phi}_{\mathcal{O}}$. Since for every sequence for which ϕ holds there exists observationally equal one, for which it does not hold and vice versa, we have $P \in sOp^{\phi}_{\mathcal{O}}$. Now let us consider process $P = l.l'.Nil + l.(h.l'.Nil + l'.Nil)$, an observation function which does not see action h and a predicate which holds for sequences containing h action. It easy to check that $P \in SOp^{\phi}_{\mathcal{O}}$ but $P \notin sOp^{\phi}_{\mathcal{O}}$. For process $P' = l.l'.Nil + l.h.l'.Nil$ we have $P' \notin SOp^{\phi}_{\mathcal{O}}$ but $P' \in sOp^{\phi}_{\mathcal{O}}$. The rest of the proof follows from Lemma 1, Propositions 1 and 4.

As it was mentioned, the opacity properties could be undecidable even for very simple observation functions or predicates (depending on their mutual combination). Here we can obtain its decidability by restrictions put on O_{ϕ} and $O_{\neg\phi}$, respectively. Note that existence of $O_{\neg\phi}$ for given O_{ϕ} is not guaranteed in general due to Turing power of TPA. As regards observation function, m-orwellian ones are the most interesting, since for their computations we do not need infinite memory and still the most of real attacks are based on them. As regards predicates, again those ones, which can be associated with finite automata are the most useful and frequent ones. If a combination of an observation function and predicates results in finite state process algebra the resulting properties are decidable. We elaborate this more precisely now.

We say that process E emulates an observational function $\mathcal{O}$ if it produces the corresponding output after receiving input traces. Formally, for every $w \in Actt^*$ it holds $\mathcal{O}(w) = o$ iff $(E|w.Nil) \setminus A \approx o$.

Lemma 2. *For every m-orwellian observation function there exists finite state process which emulates it.*

Proof. Sketch. Emulating process has to record the previous m inputs from emulated trace to produce an output. Emulation is straightforward. If $|A| = n$ then process which emulates given m-orwellian function has $O(m^n)$ states.

We call predicate ϕ finitely definable, if there exist finite state process T such that for every $w \in Actt^*$ $\phi(w)$ holds iff $(T|w.Nil) \setminus A \approx \surd.Nil$ where $\surd$ is a new symbol indicating the successful termination.

Proposition 7. *Let ϕ and $\neg\phi$ are finitely definable. Then opacity properties $SOp^{\phi}_{\mathcal{O}}$ and $sSOp^{\phi}_{\mathcal{O}}$ could be decided in time $O((n.m.k.|A|)^6)$ and $O((n.m.k.|A|)^3)$ for finite state processes and every m-orwellian observation function $\mathcal{O}$, where n, m, k are numbers of states of P, process emulating $\mathcal{O}$ and maximum of number of states of processes corresponding to $\phi, \neg\phi$, respectively.*

Proof. Sketch. We combine processes $\phi, \neg\phi$ and $\mathcal{O}$. First we need a special process which duplicates all action and one copy is send to process corresponding to the predicate and to proces for observation function. The size of this auxiliary process is $O(|A|)$. Hence the overall size of the process is $n.m.k.|A|$. The rest of the proof follows from complexity results for weak simulation and weak bisimulation (see [CPS90,KS83]).

If we have a process which does not belong to $SOp^{\phi}_{\mathcal{O}}$ for some ϕ and $\mathcal{O}$ then this means that the process could be jeopardize by an attacker which can react to process by means of $\mathcal{O}$ and is interested in validity of ϕ. But there are attacks which are not covered by our framework. For example, timing attacks, which have a particular position among attacks against systems security. They represent a powerful tool for "breaking" "unbreakable" systems, algorithms, protocols, etc. For example, by carefully measuring the amount of time required to perform private key operations, attackers may be able to find fixed Diffie-Hellman exponents, factor RSA keys, and break other cryptosystems (see [Ko96]). This idea was developed in [DKL98] where a timing attack against smart card implementation of RSA was conducted.

We can extend our framework so that we can model also timing attacks and we can distinguish them from ordinary attacks. Here we formulate the property for simulation opacity but the same can be done also for strong simulation opacity.

Definition 11. *We say that process P is jeopardized by timing attack on validity of ϕ with a given observation function iff $(P|O_{\phi}) \setminus A \not\prec (P|O_{\neg\phi}) \setminus A$ and $(P|O_{\phi}) \setminus A \prec_{\{t\}} (P|O_{\neg\phi}) \setminus A$.*

Example 2. Let $\phi^{n,m}(s)$ for $1 < n < m$ holds iff $s = s_1.h.s_2.h'.s_3, h, h' \in H$ such that $n \leq |s_2|_{\{t\}}| \leq m$ and $s_1, s_2, s_3 \in (L \cup \{t\})^*$, i.e. $\phi^{n,m}(s)$ holds if s contains two private actions from H and time elapsing between their occurrences is between n and m time units and observation function see just low level actions and elapsing of time. Then the following process

$$O_{\phi}\mu X.(\sum_{x \in L} x.o_x.X + \sum_{x \in H} x.F')$$

where

$$F' = \mu X.(\sum_{x \in L} x.o_x.X + t.F_1),$$

$$F_i = \mu X.(\sum_{x \in L} x.o_x.X + t.F_{i+1})$$

for $i < n$ and

$$F_i = \mu X.(\sum_{x \in L} x.o_x.X + t.F'_{i+1})$$

for $i = n$,

$$F'_i = \mu X.(\sum_{x \in L} x.o_x.X + \sum_{x \in H} x.x^g.F'' + t.F'_{i+1})$$

for $i < m$ and

$$F'' = \mu X.(\sum_{x \in L} x.o_x.X + \sum_{x \in H} x.o.X + \sum_{x \in L} x.o_x.O_\phi + \sum_{x \in H} x.o.O_\phi)$$

is the process definition of predicate $\phi^{n,m}$. Similarly, for predicate $\neg\phi^{n,m}$ we can construct an appropriate finite state process. Clearly, timed proces are jeopardize by timing attacks on validity of $\phi^{n,m}$.

5 Conclusions

We have presented generalization of opacity called simulation opacity and we have elaborated it in timed process algebra setting. This concept offers not only an uniform framework for security theory but can be used to model more elaborated security properties than traditional ones and moreover, by careful choice of processes expressing predicated and observations we can obtain properties which can be effectively checked (note that in general, opacity is undecidable). By this concept we can also naturally model security with respect to limited time length of an attack, with a limited number of attempts to perform an attack and so on.

The presented approach allows us to use also other types of process algebras enriched by operators expressing also other properties (space, distribution, networking architecture, processor or power consumption and so on) and in this way also other types of attacks which exploit these information to detect information flow through various covert channels can be described.

Our approach limits us to predicates and observation functions (i.e. observers) which can be expressed by process algebra processes. In fact, this restriction does not represent any real limitation. Practically, all predicates and observation function of interest (used in known attacks) can be described by finite state processes and there is even no need to exploit full universal power of process algebras. In other words, it has no practical meaning to consider predicates and observation functions which cannot be effectively computed.

References

[BKR04] Bryans J., M. Koutny and P. Ryan: Modelling non-deducibility using Petri Nets. Proc. of the 2nd International Workshop on Security Issues with Petri Nets and other Computational Models, 2004.

[BKMR06] Bryans J., M. Koutny, L. Mazare and P. Ryan: Opacity Generalised to Transition Systems. In Proceedings of the Formal Aspects in Security and Trust, LNCS 3866, Springer, Berlin, 2006.

[BG04] Busi N. and R. Gorrieri: Positive Non-interference in Elementary and Trace Nets. Proc. of Application and Theory of Petri Nets 2004, LNCS 3099, Springer, Berlin, 2004.

[CPS90] Cleaveland R, J. Parrow and B. Steffen: A semantics-based verification tool for finite-state systems. Proc of Protocol specification, testong and verification, Elsevier Science Publishers, 1990.

[DKL98] Dhem J.-F., F. Koeune, P.-A. Leroux, P. Mestre, J.-J. Quisquater and J.-L. Willems: A practical implementation of the timing attack. Proc. of the Third Working Conference on Smart Card Research and Advanced Applications (CARDIS 1998), LNCS 1820, Springer, Berlin, 1998.

[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.

[GM04] Gorrieri R. and F. Martinelli: A simple framework for real-time cryptographic protocol analysis with compositional proof rules. Science of Computer Programing, Volume 50, Issues 13, 2004.

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

[Gro90] Groote, J. F.: "Transition Systems Specification with Negative Premises". Baeten, J.C.M. and Klop, J.W. (eds.), *CONCUR'90*, Springer Verlag, Berlin, LNCS 458, 1990.

[Gru12] Gruska D.P.: Informational analysis of security and integrity. Fundamenta Informaticae, vol. 120, Numbers 3-4, 2012.

[Gru12a] Test based security. Concurrency, Specification and Verification CS&P 2012, Vol. 1, Berlin, 2012.

[Gru11] Gruska D.P.: Gained and Excluded Private Actions by Process Observations. Fundamenta Informaticae, Vol. 109, Number 3, 2011.

[Gru10] Gruska D.P.: Process algebra contexts and security properties. Fundamenta Informaticae, vol. 102, Number 1, 2010.

[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.

[KS83] Kanellakis, P. C. and S.A. Smolka: CCS expressions, finite state processes, and three problems of equivalence. Proc. of the second annual ACM symposium on Principles of distributed computing, ACM, 1983.

[Ko96] Kocher P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems. Proc. Advances in Cryptology - CRYPTO'96, LNCS 1109, Springer, Berlin, 1996.

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