Observations, Testing and Security ⋆

Damas P. Gruska[1] and M. Carmen Ruiz[2]

[1] Comenius University, Slovakia
[2] Universidad de Castilla-La Mancha, Spain

Abstract. Testing of security for multi-agent systems is proposed and studied. We assume an attacker, as an agent, who can accumulate and exploit knowledge about other agents. This will be done by observation function and security property called process opacity. Unfortunately, this property is undecidable in general, so we propose its more realistic variant based on tests and testing. Here we consider systems to be secure if they cannot be compromised by a given test or set of tests. In the end, we state a decidability result for security testing.

Keywords: multi-agent systems, process algebras, information flow, security, testing

1 Introduction

Formal methods play an important role to guarantee software quality. On the other side their application is frequently either rather expensive or practically impossible due to complexity issues. On top of these there are properties which cannot be formally verified in general due to their undecidability. In such cases, tests and testing offer a more realistic approach.

In this paper, we address security properties of systems, particularly multi-agent (MAS). We consider an attacker, as one of agents, who can accumulate some knowledge on the behaviour of other agents. We propose corresponding security properties and means how to test these properties. In a sense, we combine formal verification with testing. 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 [Gru13] we have exploited this idea, but we weaken it by requiring that forbidding and the 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]).

⋆ Work supported by the grant VEGA 1/0778/18.

Here we start with security concept called opacity. To explain the opacity principle, let's suppose that we have some property ϕ over sequences of actions. Such property 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. We would like to know whether an observer can deduce the validity of the property ϕ just by partially observing (not all actions are visible) sequences of actions (traces) performed by the given process. The observer cannot deduce the validity of ϕ if there are two traces w, w' such that $\phi(w) \wedge \neg\phi(w')$ holds and the traces cannot be distinguished by the observer. In [Gru15] opacity is modified (the result is called process opacity) in such a way that instead of a process' traces we focus on properties of reachable states and attackers which can see only some process's actions. Hence we assume an intruder who is not primarily interested in whether some sequence of actions performed by a given process has some given property but we consider an intruder who wants to discover whether this process reaches a state which satisfied some given (classified) predicate. It turned out that in this way we could capture many new security flaws. On the other hand some security flaws, particularly important for multi-agent systems, are not covered by this state-based security property neither by its variant called an initial state opacity or infinite studied in [Gru17,GR19b].

In this paper we extend process opacity to reflect attackers which can accumulate some knowledge about other system's behaviour. This approach is particularly appealing for multi-agent systems where one of the agents could be an attacker. 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. There are several ways to overcome these disadvantages i.e. either quantify information flow or put some restrictions on attacker's capabilities. 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. 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 a different approach. We define tests and testing of security properties. Hence instead of requiring general security we require security with respect to a given set of tests. Each test represents a possible scenario of an attacker as well his or her capabilities. For example, an access control system with strong password policy should be considered reasonable secure with respect to a "small" attackers (tests), which can try only a few passwords. 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's security.

In this paper we also exploit an idea of observation function which express a capability of an attacker to accumulate some knowledge over MAS behaviour.

Attacker's observations may not be just simple ones, say that some actions are visible for her and others are not (see [GR19a]). Visibility of a particular action could depend on previous actions as well. Hence the presented testing approach is strictly stronger then that of [Gru11], which is based on simple process's observations.

Contribution of the work can be summarized as follows: 1. definition of security with respect to a given test, 2. modeling observations by processes, 3. modeling predicates by processes, 4. reducing security checking to process's traces checking, 5. decidability results, for practically the most interesting, finite states processes.

The paper is organized as follows. Our working formalism is introduced in Section 2. In Section 3 we describe information flow security properties of interest. In Sections 4 we define tests and testing and in Section 5 we relate testing and security.

2 Working Formalism

As an working formalism we will use Milner's CCS (see [Mil89]). Let A be a se of atomic action symbols not containing symbols 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 $a, b, u, v \ldots$ range over A, $x, y, \ldots$ range over Act, For $s = x_1.x_2.\ldots.x_n, x_i \in Act$ 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. The set of all traces of P will be denoted by $Tr(P)$. By ϵ we will denote the empty sequence of actions, by $Succ(P)$ we will denote the set of all successors of P i.e. $Succ(P) = \{P' | P \xrightarrow{s} P', s \in Act^*\}$. If the set $Succ(P)$ is finite we say that P is a finite state process. We define modified transitions $\stackrel{x}{\Rightarrow}_M$ which "hide" actions from M. Formally, we will write $P \stackrel{x}{\Rightarrow}_M P'$ for $M \subseteq Act$ iff $P \xrightarrow{s_1}\xrightarrow{x}\xrightarrow{s_2} P'$ for $s_1, s_2 \in M^\star$ and $P \stackrel{s}{\Rightarrow}_M$ instead of $P \stackrel{x_1}{\Rightarrow}_M \stackrel{x_2}{\Rightarrow}_M \ldots \stackrel{x_n}{\Rightarrow}_M$. 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 $x \in M$. Note that $\stackrel{x}{\Rightarrow}_M$ is defined for arbitrary action x but in definitions of security properties we will use it for actions (or sequence of actions) not belonging to M. We can extend the definition of $\Rightarrow_M$ for sequences of actions similarly to $\xrightarrow{s}$. By $s|_B$ we will denote the sequence obtained from s by removing all actions not belonging to B.

We define two equivalences which are modifications of trace equivalence and weak bisimulation, respectively (see [Mil89]).

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 $(Act, Act, \rightarrow)$ be a labelled transition system (LTS). A relation $\Re \subseteq Act \times Act$ is called a* M-bisimulation *if it is symmetric and it satisfies*

the following condition: if $(P,Q) \in \Re$ *and* $P \xrightarrow{x} P', x \in Act$ *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 M-bisimulation relating* P *and* Q.

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. Firstly we define observation function on sequences from $Act^\star$.

Definition 3 (Observation). *Let* Θ *be a set of elements called observables. Any function* $\mathcal{O} : Act^\star \to \Theta^\star$ *is an observation function. It is called static /dynamic /orwellian / m-orwellian* ($m \geq 1$) *if the following conditions hold respectively (below we assume* $w = x_1 \dots x_n$*):*

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

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. Note that the same action can be seen differently during an observation (except static observation function) and this express a possibility to accumulate some knowledge by intruder. For example, an action not visible at the beginning could become somehow observable.

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 $Act^\star$ performed by given process. The observer cannot deduce the validity of ϕ if there are two traces $w, w' \in Act^\star$ 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 4 (Opacity). *Given process P, a predicate ϕ over $Act^\star$ 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 $Op^{\phi}_{\mathcal{O}}$.*

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 the both traces are indistinguishable for an observer (which is expressed by an observation function). This means that the observer (intruder) cannot say whether a trace for which the predicate holds has been performed or not.

4 Process Opacity

Now let us assume a different scenario, namely that an intruder is not interested in traces and their properties but he or she tries to discover whether a given process has reached a state with some given property which is expressed by a (total) predicate. This property might be process deadlock, capability to execute only traces with some given actions, capability to perform at the same actions form a given set, incapacity to idle (to perform τ action) etc. We do not put any restriction on such predicates but we only assume that they are consistent with some suitable behavioral equivalence. The formal definition follows.

Definition 5. *We say that the predicate ϕ over processes is consistent with respect to relation $\cong$ if whenever $P \cong P'$ then $\phi(P) \Leftrightarrow \phi(P')$.*

As the consistency relation $\cong$ we could take bisimulation ($\approx_\emptyset$), weak bisimulation ($\approx_{\{\tau\}}$) or any other suitable equivalence.

.

Definition 6 (Process Opacity). *Given process P, a predicate ϕ over processes is process opaque w.r.t. the observation function $\mathcal{O}$ whenever $P \xrightarrow{w} P'$ for $w \in Act^*$ and $\phi(P')$ holds then there exists P'' such that $P \xrightarrow{w'} P''$ for some $w' \in Act^*$ and $\neg\phi(P'')$ holds and moreover $\mathcal{O}(w) = \mathcal{O}(w')$. The set of processes for which the predicate ϕ is process opaque w.r.t. to the $\mathcal{O}$ will be denoted by $POp^{\phi}_{\mathcal{O}}$.*

Schematically, Fig. 1 depicts process opacity. In [Gru15]) process opacity is defined for static observational function, namely a set of public actions which can

$$\begin{array}{lll} P & \stackrel{w}{\Longrightarrow} \quad \phi(P') & \mathcal{O}(w) \\ & & \| \\ P & \stackrel{w'}{\Longrightarrow} \quad \neg\phi(P'') & \mathcal{O}(w') \end{array}$$

Fig. 1. Process opacity with respect to $\mathcal{O}$

be observed and a set of hidden (not necessarily private) actions are assumed. To model such observations we exploit the relation $\stackrel{s}{\Rightarrow}_M$ where actions from M are those ones which could not be seen by the observer.

Definition 7. *Given process P, a predicate ϕ over processes is process opaque w.r.t. the set M if whenever $P \stackrel{s}{\Rightarrow}_M P'$ for $s \in (Act \setminus M)^*$ and $\phi(P')$ holds then there exists P'' such that $P \stackrel{s}{\Rightarrow}_M P''$ and $\neg\phi(P'')$ holds. The set of processes for which the predicate ϕ is process opaque w.r.t. to the M will be denoted by POp^{ϕ}_{M}.*

Schematically, Fig. 2 depicts process opacity e w.r.t. the set M . Note that if $P \cong P'$ then $P \in POp^{\phi}_{M} \Leftrightarrow P' \in POp^{\phi}_{M}$ whenever ϕ is consistent with respect to $\cong$ and $\cong$ is such that it is a subset of the trace equivalence (defined as $\simeq_w$ but instead of $\stackrel{s}{\Rightarrow}_{\{\tau\}}$ we use $\stackrel{s}{\Rightarrow}_{\emptyset}$).

$$\begin{array}{lll} P & \stackrel{s}{\Longrightarrow}_M \quad \phi(P') \\ \\ P & \stackrel{s}{\Longrightarrow}_M \quad \neg\phi(P'') \end{array}$$

Fig. 2. Process opacity with respect to M

A relation between these two security properties is stated by the following lemma.

Lemma 1. *Let $\mathcal{O} : Act \to Act$ such that $\mathcal{O}(x) = \epsilon$ iff $x \in M$ and $\mathcal{O}(x) = x$ otherwise. Then $P \in POp^{\phi}_{\mathcal{O}}$ iff $P \in POp^{\phi}_{M}$.*

Proof. Directly from Definitions 6 and 7.

The process opacity is defined for arbitrary predicates and observation functions. Now we will reformulate it for those ones which can be expressed by process algebras. Firstly we start with observation function $\mathcal{O}$. Suppose that $Act \cap \Theta = \emptyset$. We extend the set of actions A by Θ and we model $\mathcal{O}$ by a special process.

Definition 8. *Process O is called process definition of observation function $\mathcal{O}$ if for every P and $s \in Act^*$ it holds $P \stackrel{s}{\longrightarrow} P'$ iff $(P|O) \setminus A \stackrel{o}{\Rightarrow} (P'|O') \setminus A$ for $o \in \Theta^*$ such that $\mathcal{O}(s) = o$.*

Some observation function cannot be computed at all (we can prove this by defining a function which returns a specific symbol if a given Turing machine halts). But some observation functions can be emulated by finite state processes as it is stated by the following lemma.

Lemma 2. *For any static or m-orwellian observation functions $\mathcal{O}$ there exists finite state process O which is process definition of observation function $\mathcal{O}$.*

Proof. Sketch. Any static or m-orwellian observation function can be simulated by finite-state transducer which can be simulated by finite state process.

Note that also some dynamic and orwellian observation function can be defined by finite state systems in a case that their computation does not need an unlimited memory.

Now we can relate observation functions defined by processes with the definition of process opacity.

Lemma 3. *Let for every $o \in \Theta^*$ such that $(P|O) \setminus A \stackrel{o}{\Rightarrow} (P'|O') \setminus A$ and $\phi(P')$ holds there exists P'' such that $(P|O) \setminus A \stackrel{o}{\Rightarrow} (P''|O') \setminus A$ such that $\neg\phi(P'')$ holds. Then $P \in POp^{\phi}_{\mathcal{O}}$ and vice versa.*

Proof. Directly from Definitions 6 and 8.

An observation function defines what an attacker can see from process behaviour. One attacker could see more that another attacker. There are several ways how to express this situation. In the following definition we suppose that the set of observables for one attacker is a subset of observables of another attacker. This gives an ordering between observation functions.

Definition 9. *Let $\mathcal{O}_1, \mathcal{O}_2$ are two observation functions with the common set of observables Θ. We define ordering between them (denoted by $\prec$) as follows. $\mathcal{O}_1 \prec \mathcal{O}_2$ iff for every $w, w \in Act^*$ we have $\mathcal{O}_1(w)|_U = \mathcal{O}_2(w)$ for some set $U, U \subseteq \Theta$.*

Lemma 4. *Let $\mathcal{O}_1, \mathcal{O}_2$ are two observation functions such that $\mathcal{O}_1 \prec \mathcal{O}_2$. Then for every $w, w \in Act^*$ it holds that $\mathcal{O}_1(w) = \mathcal{O}_1(w')$ implies $\mathcal{O}_2(w) = \mathcal{O}_2(w')$.*

Proof. Let $\mathcal{O}_1(w) = \mathcal{O}_1(w')$. Then also $\mathcal{O}_1(w)|_U = \mathcal{O}_1(w')|_U$ for arbitrary U. Since $\mathcal{O}_1 \prec \mathcal{O}_2$ we have $\mathcal{O}_1(w)|_U = \mathcal{O}_2(w)$ and $\mathcal{O}_1(w')|_U = \mathcal{O}_2(w')$ i.e. $\mathcal{O}_2(w) = \mathcal{O}_2(w')$.

The ordering between observation functions corresponds to stronger and weaker process opacity properties as it is stated by the following lemma.

Lemma 5. *Let $\mathcal{O}_1, \mathcal{O}_2$ are two observation functions such that $\mathcal{O}_1 \prec \mathcal{O}_2$. Then $POp^{\phi}_{\mathcal{O}_1} \subseteq POp^{\phi}_{\mathcal{O}_2}$.*

Proof. Main idea. Let $P \in POp^{\phi}_{\mathcal{O}_1}$ Then from Definitions 6 we know that whenever $P \xrightarrow{w} P'$ for $w \in Act^*$ and $\phi(P')$ holds then there exists P'' such that $P \xrightarrow{w'} P''$ for some $w' \in Act^*$ and $\neg\phi(P'')$ holds and moreover $\mathcal{O}_1(w) = \mathcal{O}_1(w')$. But since $\mathcal{O}_1 \prec \mathcal{O}_2$ we have by the previous lemma also $\mathcal{O}_2(w) = \mathcal{O}_2(w')$ and hence $P \in POp^{\phi}_{\mathcal{O}_2}$.

The ordering on observation functions could be related to the relations $\Rightarrow_M$ as it is stated by the following Lemma.

Lemma 6. *Let O_1, O_2 are process definitions of observation functions $\mathcal{O}_1, \mathcal{O}_2$, respectively and $\mathcal{O}_1 \prec \mathcal{O}_2$. Then for every P and $o \in \Theta^*$ it holds $(P|O_1) \setminus A \stackrel{o}{\Rightarrow} (P'|O'_1) \setminus A$ iff $(P|O_2) \setminus A \stackrel{o}{\Rightarrow}_U (P'|O'_2) \setminus A$ for some $U, U \subseteq \Theta$.*

Proof. Sketch. Since $\mathcal{O}_1 \prec \mathcal{O}_2$ from Definition 9 we have $\mathcal{O}_1(w)|_U = \mathcal{O}_2(w)$ for some set $U, U \subseteq \Theta$. And from Definition 8 we have translation of this fact of this to traces.

Note that there also other possibilities how to order intruders with respect to their capabilities to observe processes behaviour expressed by observation function. For example, observation of one attacker can be included in observations of another one, one attacker cannot see completely something which can be seen by another one etc. We leave investigations of such orderings to future work.

5 Testing

Process opacity is undecidable property, since its special variant (see Definition 7) is undecidable (see [Gru15]). Undecidability results from many factors, one of them is too general notion of observer's capabilities. Now we use a concept of testing, which will help us to define security with respect to given test or set of tests.

A test examines process's capability to perform some set of actions. It is required that every sequence of test's actions should be emulated also by tested process i.e. for a test T and tested process P it should hold $Tr_w(P_T) \subseteq Tr_w(P)$ where P_T is behavior of P under the test T. But since we will later incorporate also visibility of such traces as well as properties of resulting processes we need a more elaborated definition of testing. We suppose that a test communicates with tested process by means of new alternative actions, which are original ones just indexed by a (i.e. alternative actions to x, y, z are depicted as x_a, y_a, y_a). The formal definition follows.

Definition 10. *Process T is called a test for process P iff $Tr_w((T|P[f]) \setminus A_a) \subseteq Tr_w(P)$ where f maps every action of $Sort(P)$ to its alter ego, depicted by index a. Alter age actions will be denoted by A_a.*

In the following example we present a test which examines all process traces.

Example 1. Let $T = \mu X.\sum_{x \in A}(x.\bar{x}_a.X + \bar{x}.x_a.X)$. Then we have $Tr_w((T|P[f]) \setminus A_a) = Tr_w(P)$ for every P and hence T will be called a simple complete test.

Now we are prepared for combining testing with observation functions. It reduces general process opacity to property which requires that for an observer/tester, expressed by T, tested process P is secure.

Definition 11. *Let process O is a process definition of observation function $\mathcal{O}$ and T is the test of process P. We say that P pass test T under O and ϕ (denoted by $P \in POp^{\phi}_{\mathcal{O}}(T)$) iff if for every $o \in \Theta^*$ it holds $((T|P[f])\ A_a|O)\setminus A \stackrel{o}{\Rightarrow} ((T'|P'[f])\ A_a|O')\setminus A$ and $\phi(P')$ holds there exists P'' such that $((T|P[f])\ A_a|O)\setminus A \stackrel{o}{\Rightarrow} ((T'|P''[f])\ A_a|O') \setminus A$ such that $\neg\phi(P'')$ holds.*

Testing by test T and observation function defined by process O is depicted by Fig. 3. Actions of process P are firstly tested by T and its sequence of actions and then the same sequence is taken as an input for O to produce its visible part, i.e. as a sequence of observables from Θ.

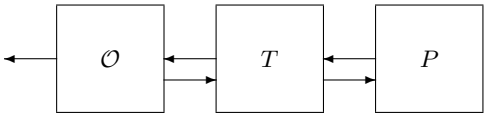

Fig. 3. Testing scenario

If a process P is process opaque with respect to ϕ and $\mathcal{O}$ then it should be process opaque with respect to any test T as its is stated by the following Lemma.

Lemma 7. *Let process O is process definition of observation function $\mathcal{O}$, test T of process P and $P \in POp^{\phi}_{\mathcal{O}}$. Then P pass test T under O.*

Proof. Directly from Definitions 6 and 11.

Now we can show how a natural ordering between tests defines ordering on security of processes with respect to process opacity.

Lemma 8. *Let T_1, T_2 are two tests for process P such that $Tr(T_1) \subseteq Tr(T_2)$. Then if $P \in POp^{\phi}_{\mathcal{O}}(T_2)$ then $P \in POp^{\phi}_{\mathcal{O}}(T_1)$.*

Proof. Sketch. Let $P \in POp^{\phi}_{\mathcal{O}}(T_2)$ i.e. it has to pass test T_2 which represents, basically a set of traces. Since test T_1 produces less traces it has to pass also this test and hence $P \in POp^{\phi}_{\mathcal{O}}(T_1)$.

As a direct consequence of the previous Lemma we have the following one.

Lemma 9. *Let T_1, T_2 are two tests for process P. Then if $P \in POp^{\phi}_{\mathcal{O}}(T_1 + T_2)$ then $P \in POp^{\phi}_{\mathcal{O}}(T_1)$ and $P \in POp^{\phi}_{\mathcal{O}}(T_2)$.*

Proof. The proof follows from the previous Lemma and facts that $Tr(T_1) \subseteq Tr(T_1 + T_2)$ and $Tr(T_2) \subseteq Tr(T_1 + T_2)$.

A property similar to the one expressed by Lemma 8 holds for predicates i.e. security with respect to weaker predicate implies security with respect to stronger one.

Lemma 10. *Let ϕ_1, ϕ_2 are two predicates such that $\phi_1 \Rightarrow \phi_2$ and T is a test for process P. Then if $P \in POp^{\phi_2}_{\mathcal{O}}(T)$ then $P \in POp^{\phi_1}_{\mathcal{O}}(T)$.*

Proof. Let $P \in POp^{\phi_2}_{\mathcal{O}}(T)$ and let $P \xrightarrow{w} P'$ for $w \in Act^*$ and $\phi_1(P')$ holds. Since $\phi_1 \Rightarrow \phi_2$ then also $\phi_2(P')$ holds and since $P \in POp^{\phi_2}_{\mathcal{O}}(T)$ there exists P'' such that $P \xrightarrow{w'} P''$ for some $w' \in Act^*$ and $\neg\phi_2(P'')$ holds and moreover $\mathcal{O}(w) = \mathcal{O}(w')$. Again since $\neg\phi_2 \Rightarrow \neg\phi_1$ then also $\neg\phi_1(P'')$ holds and hence $P \in POp^{\phi_1}_{\mathcal{O}}(T)$.

Now we show how checking of process opacity could be reduced to checking of standard process algebra properties. Firstly we define also a predicate over processes by a special process.

Definition 12. *Predicate ϕ is called process definable if there exists a process P_ϕ such that $\phi(P)$ holds iff $(P|P_\phi) \setminus A \stackrel{\surd}{\Rightarrow}$ where $\surd$ is a new action $\surd \notin A$. Process P_ϕ is called process definition of ϕ.*

Example 2. Let $\phi(Q)$ holds of Q can once perform action a then later action b. Process P_ϕ defined as $P_\phi = \mu X. \sum_{x \neq a} x.X + a.P'$, where $P' = \mu X. \sum_{x \neq b} x.X + b.\surd.Nil$ is process definition of ϕ.

Now we can reduce process opacity checking to checking of trace inclusion as it is stated by the following theorem.

Theorem 1. *Let $O, P_\phi, P_{\neg\phi}$ are process definitions of observation functions $\mathcal{O}$ and predicates $\phi, \neg\phi$, respectively. Then $P \in POp^{\phi}_{\mathcal{O}}$ iff for every $o \in \Theta^*$ it holds that if $o.\surd \in Tr_w(P|O|P_\phi) \setminus A$ then $o.\surd \in Tr_w(P|O|P_{\neg\phi}) \setminus A$.*

Proof. The proof follows from Definitions 6, 8 and 12.

Now, thank to the above mention reduction, we can obtain decidable variant for process opacity.

Theorem 2. *Let $O, T, P_\phi, P_{\neg\phi}$ are finite state process definitions of observation functions $\mathcal{O}$, test and predicates $\phi, \neg\phi$, respectively. Then process opacity is decidable for any finite state test.*

Proof. Sketch. The proof follows from Definitions 6, 8 and 12 and Theorem 1.

The security property process opacity expects an attacker who can just observe process' traces but cannot interact with the process. In many cases this does not cover real attacks and attackers, particularly in the case of multi-agent systems where a possible attacker could be one of the agents. Hence we generalize process opacity in such a way that also every successor of a process has to be process opaque as well. The formal definition follows.

Definition 13 (Persistent Process Opacity). *We say that process P is persistently process opaque w.r.t. the observation function $\mathcal{O}$ and ϕ if $Succ(P) \subseteq PPOp^{\phi}_{\mathcal{O}}$.*

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

Proof. The main idea. Let $P \in PPOp^{\phi}_{\mathcal{O}}$. Then directly from Definition 13 we have that $P \in POp^{\phi}_{\mathcal{O}}$. To show that the inclusion is proper let us assume the following example. Let $P = a.b.b.Nil + a.c.c.Nil$, an observation function $\mathcal{O}$ such that $\mathcal{O}(a) = a, \mathcal{O}(b) = \mathcal{O}(c) = \epsilon$ and predicate ϕ such that $\phi(Q)$ hold iff Q can perform a or b. Then $P \in POp^{\phi}_{\mathcal{O}}$ but $P \notin PPOp^{\phi}_{\mathcal{O}}$ since $b.b.Nil \notin POp^{\phi}_{\mathcal{O}}$.

For persistent process opacity similar properties as for process opacity but some modifications are needed. For example, in Lemma 8 we have to replace trace inclusion by simulation, what is an asymmetric variant of bisimulation.

6 Conclusions

We have presented the security concept called process opacity and its stronger variant persistent process opacity. We show how to model observation functions, which express capability to observe a system as well as capabilities to accumulate some knowledge on its behaviour as well as security predicates by processes. We have proposed a way how to (partially) verify these properties by means of tests and testings. Each test represents a scenario for an attacker as well his or her capabilities. Instead of verifying process opacity we define system security with respect to a given set of tests. We have shown, that under some restriction, this testing is feasible or at least decidable.

As future work, we plan to define and study a minimal set of tests which are necessary to be passed to guarantee some security property. In this way we would simplify overall testing. Moreover we plan to work with different ordering on observation functions as well as tests.

The presented approach allows us to exploit also process algebras enriched by operators expressing other "parameters" (space, distribution, networking architecture, processor or power consumption and so on). Hence we could obtain security properties which have even higher practical value.

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.

[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.

[GM04] Gorrieri R. and F. Martinelli: A simple framework for real-time cryptographic protocol analysis with compositional proof rules. to appear at Science of Computer Programing.

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

[GR19a] Gruska D.P and M. C. Ruiz: Security Testing for Multi-Agent Systems. IWANN 2019, LNCS 11506, Springer, 2019.

[GR19b] Gruska D.P and M. C. Ruiz: Security of Low Level IoT. ICCS 2019, LNCS 11538, Springer, 2019.

[Gru18] Gruska D.P and M. C, Ruiz: Opacity-enforcing for Process Algebras. CS&P'2018, 2018.

[Gru17] Gruska D.P and M. C, Ruiz: Initial process security. in Specification and Verification CS&P'2017, 2017.

[Gru15] Gruska D.P.: Process Opacity for Timed Process Algebra. In Perspectives of System Informatics, LNCS 8974, 2015.

[Gru13] Gruska D.P.: Information flow testing Fundamenta Informaticae. - Vol. 128, No. 1-2 (2013).

[Gru11] Gruska D.P.: Gained and Excluded Private Actions by Process Observations. Fundamenta Informaticae. - Vol. 109, No. 3 (2011).

[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.