Security in a Model for Long–running Transactions

Damas P. Gruska[*]
Institute of Informatics, Comenius University
Mlynska dolina, 842 48 Bratislava, Slovakia
Email: gruska@fmph.uniba.sk

Andrea Maggiolo–Schettini, Paolo Milazzo
Dipartimento di Informatica, Università di Pisa
Largo B. Pontecorvo 3, 56127 Pisa, Italy
Email: {maggiolo,milazzo}@di.unipi.it

Abstract. Communicating Hierarchical Transaction-based Timed Automata have been introduced to model systems performing long–running transactions. Here, for these automata a security concept is introduced, which is based on a notion of opacity and on the assumption that an attacker can not only observe public system activities, but also cause abortion of some of them. Different intruder capabilities as well as different kinds of opacity are defined and the resulting security properties are investigated. Security of long–running transactions is defined by the mentioned notion of opacity and conditions for compositionality are established.

1. Introduction

Opacity is one of the strongest security concepts as, with its help, many other security properties can be expressed (see [3]). Its origin can be traced to a concept of non-interference (see [7]), which assumes the absence of any information flow between private and public system 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,

[*] Work supported by European Science Foundation program AutoMathA and by the grant VEGA 1/3105/06.

Address for correspondence: Damas P. Gruska, Institute of Informatics, Comenius University, Mlynska dolina, 842 48 Bratislava, Slovakia.

computational models and nature or "quality" of observations. All reformulations try to capture important aspects of system behaviour with respect to possible attacks against systems security, and often are tailored to some types of attacks.

Timing attacks 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 [10]). This idea was developed in [5] where a timing attack against smart card implementation of RSA was conducted. In [9], a timing attack on the RC5 block encryption algorithm, in [13] the one against the popular SSH protocol and in [6] the one against web privacy are described.

To perform different kinds of timing attacks attackers might exploit different capabilities. For example, for some attacks it is enough if an attacker can only observe the system to be attacked. For other attacks an attacker has to communicate with the system via public actions, either directly or by means of an embedded auxiliary system. Some attacks exploit the brute force of an attacker who can interrupt some system activities (by resetting system components, breaking communication links, etc). Particularly sensitive to such type of attacks are systems performing so called long–running transactions (LRTs). A LRT is composed by atomic activities that should be executed completely. Atomicity means that they are either successfully executed or no effect is observed if their execution fails. Partial executions of a LRT are not desirable, and, if they occur, they must be compensated for. Therefore, all the activities A_i in a LRT have a compensating activity B_i that can be invoked to recover from the effects of a successful execution of A_i if some failure occurs later. Hence from the computational point of view the system is robust with respect to abortion of some of its activities. However these abortions may lead to some information flow between classified and public system activities.

In [11] we have introduced Communicating Hierarchical Transaction–based Timed Automata (CHTTAs) to model LRTs. In this paper we investigate information flow based attacks for systems described with CHTTAs and attackers that can not only passively observe public system activities but also actively cause abortion of system activities. We model information flow by the notion of opacity for which we give different definition depending on the assumed capabilities of attackers. The introduced concepts are used to investigate security of LRTs. We study under which conditions opacity of LRTs can be established compositionally.

In Section 2 we recall CHTTAs. In Section 3 we study opacity of CHTTAs, reformulating it for intruders who can also abort system activities. In Section 4 we discuss the application to LRTs. In Section 5 we conclude.

2. Communicating Hierarchical Timed Automata

Let us assume a finite set of communication channels $\mathcal{C}$ partitioned into a set $\mathcal{C}_{Pub}$ of *public* channel and a set $\mathcal{C} \setminus \mathcal{C}_{Pub}$ of *private* channels. As usual, we denote with $a!$ the action of sending a signal on channel a and with $a?$ the action of receiving a signal on a. Let Σ_C denote the set of all possible sending and receiving actions on channels in $C \subseteq \mathcal{C}$.

Let us assume a finite set X of positive real variables called *clocks*. A *valuation* over X is a mapping $v : X \rightarrow \mathbb{R}^{\geq 0}$ assigning real values to clocks. Let V_X denote the set of all valuations over X. For a valuation v and a time value $t \in \mathbb{R}^{\geq 0}$, let $v + t$ denote the valuation such that $(v + t)(x) = v(x) + t$, for each clock $x \in X$.

The set of *constraints* over X, denoted $\Phi(X)$, is defined by the following grammar, where ϕ ranges over $\Phi(X)$, $x \in X$, $c \in \mathbb{Q}$ and $\sim \in \{<, \leq, =, \neq, >, \geq\}$:

$$\phi ::= x \sim c \,|\, \phi \wedge \phi \,|\, \neg\phi \,|\, \phi \vee \phi \,|\, true$$

We write $v \models \phi$ when *the valuation v satisfies the constraint ϕ*. Formally, $v \models x \sim c$ iff $v(x) \sim c$, $v \models \phi_1 \wedge \phi_2$ iff $v \models \phi_1$ and $v \models \phi_2$, $v \models \neg\phi$ iff $v \not\models \phi$, $v \models \phi_1 \vee \phi_2$ iff $v \models \phi_1$ or $v \models \phi_2$, and $v \models true$.

Let $B \subseteq X$; with $v[B]$ we denote the valuation resulting after resetting all clocks in B. More precisely, $v[B](x) = 0$ if $x \in B$, $v[B](x) = v(x)$, otherwise. Finally, with $\mathbf{0}$ we denote the valuation such that $\mathbf{0}(x) = 0$ for all $x \in X$.

Definition 2.1. A Transaction-based Timed Automaton (TTA) is a tuple $A = (\Sigma, X, S, Q, q_0, \delta)$, where:

- $\Sigma \subseteq \Sigma_{\mathcal{C}}$ is a finite set of labels;
- X is a finite set of clocks;
- S is a finite set of superstates;
- $Q = L \cup S \cup \{\odot, \otimes\}$, where L is a finite set of basic states and $\odot$ and $\otimes$ represent the special states commit and abort, respectively;
- $q_0 \in L$ is the initial state;
- $\delta \subseteq (L \times \Sigma \cup \{\tau\} \times \Phi(X) \times 2^X \times Q) \cup (S \times \{\boxdot, \boxtimes\} \times Q)$ is the set of transitions.

A TTA is said to be *flat* when $S = \emptyset$.

Superstates are states that can be refined to automata (*hierarchical composition*). Note that from superstates in S only transitions with labels in $\{\boxdot, \boxtimes\}$ can be taken. We assume that $\odot$ and $\otimes$ are the final states of a TTA.

We now introduce CHTTAs as an extension of TTAs allowing superstate refinement and parallelism.

Definition 2.2. Let $\Sigma_{Pub} = \{a!, a? \mid a \in \mathcal{C}_{Pub}\}$ and $\mathcal{A} = \{A^1, \ldots, A^n\}$ be a finite set of TTAs, with $A^i = (\Sigma^i, X^i, S^i, Q^i, q_0^i, \delta^i)$ and such that there exists m ($m < n$) such that A^j is flat if and only if $j \geq m$. A *Communicating Hierarchical Transaction-based Timed Automaton* (CHTTA$_{\mathcal{A}}^{\Sigma_{Pub}}$) is given by:

$$\mathrm{CHTTA}_{\mathcal{A}}^{\Sigma_{Pub}} ::= \langle A^i, \mu \rangle \quad | \quad \mathrm{CHTTA}_{\mathcal{A}}^{\Sigma_{Pub}} || \mathrm{CHTTA}_{\mathcal{A}}^{\Sigma_{Pub}}$$

where μ is a *hierarchical composition function* $\mu : S^i \to \mathrm{CHTTA}_{\{A^{i+1},\ldots,A^n\}}^{\Sigma_{Pub}}$.

Parallelism allows concurrent execution of automata. Hierarchical composition allows refining superstates. Automata executed in parallel may communicate by synchronizing transitions labeled with a sending and a receiving action on the same channel. The set Σ_{Pub} contains sending and receiving actions on public channels. These actions may belong to the alphabets of TTAs in $\mathcal{A}$. Communications

Figure 1. Example of CHTTA.

performed using non public channels are only allowed between components inside the same superstate or at top–level. Communication performed by using public channels have no restrictions.

Note that, by definition of $\mathcal{A}$ and μ, cyclic nesting is avoided. In the following, if it does not give rise to ambiguity, we may write CHTTA instead of CHTTA$_{\mathcal{A}}^{\Sigma_{Pub}}$. Finally, if A is a flat TTA, in $\langle A, \mu \rangle$ μ is an empty function.

Example 2.1. In Figure 1 we show a simple example of CHTTA. Superstates of the CHTTA are depicted as boxes and basic states as circles; initial states are represented as vertical segments. Transitions are represented as labeled arrows in which labels τ and constraints $true$ are omitted. Containment in boxes represents hierarchical composition, while parallel composition is represented by juxtapositions. The CHTTA in the figure is formally defined as $\langle (\emptyset, \emptyset, \{s_1\}, \{q_0, s_1, \odot, \otimes\}, q_0, \delta), \mu \rangle$, where $\delta = \{(q_0, \tau, true, \emptyset, s_1), (s_1, \boxdot, \odot), (s_1, \boxtimes, \otimes)\}$, and $\mu(s_1) = A_1 || A_2$. Automata A_1 and A_2 are defined as $A_1 = \langle (\{a!, b?\}, \{x\}, \emptyset, \{q_0, q_1, \odot, \otimes\}, q_0, \delta_1)$ and $A_2 = \langle (\{a?, b!\}, \emptyset, \emptyset, \{q_0, q_2, \odot, \otimes\}, q_0, \delta_2)$, respectively, with $\delta_1 = \{(q_0, a!, true, \{x\}, q_1), (q_1, b?, x < 5, \emptyset, \odot), (q_1, \tau, x \geq 5, \emptyset, \otimes)\}$ and $\delta_2 = \{(q_0, a?, true, \emptyset, q_2), (q_2, b!, true, \emptyset, \odot)\}$.

Configurations of CHTTAs are pairs $tc = (c, \nu)$ where c, the *untimed configuration*, represents the currently active states, and ν, the *composed valuation*, represents the current clock valuations. The configuration of a CHTTA without parallel components, when the currently active state is a basic state, is a pair (q, v) with q the currently active state, and v the automaton clock valuation. We represent with $q.c$ the configuration where q is a superstate and c is the untimed configuration of $\mu(q)$, and with $v.\nu$ the composed valuation where v is the clock valuation of the automaton having q as superstate and ν is the composed valuation of the clocks of $\mu(q)$. We denote with $c_1; c_2$ the untimed configuration of the parallel composition of two CHTTAs having c_1 and c_2 as untimed configurations. Analogously, we denote with $\nu_1; \nu_2$ the composed valuation of the parallel composition of two CHTTAs having ν_1 and ν_2 as composed valuations. Formally, the set of configurations $Conf(A)$ of a CHTTA A is inductively defined as follows:

- if $A = \langle (\Sigma, X, S, Q, q_0, \delta), \mu \rangle$, then $Conf(A) = \{(Q \setminus S) \times V_X\} \cup \{(q.c, v.\nu) \mid q \in S \wedge v \in V_x \wedge (c, \nu) \in Conf(\mu(q))\}$;
- if $A = A_1 || A_2$ then $Conf(A) = \{(c_1; c_2, \nu_1; \nu_2) \mid (c_1, \nu_1) \in Conf(A_1) \wedge (c_2, \nu_2) \in Conf(A_2)\}$.

For a composed valuation ν and a time value $t \in \mathbb{R}^{\geq 0}$, let $\nu + t$ denote the composed valuation such that $(v + t)(x) = v(x) + t$, for each valuation v in ν.

The initial configuration of A, denoted $Init(A) \in Conf(A)$, is the configuration (c, ν) such that each state occurring in c is an initial state and each valuation occurring in ν is $\mathbf{0}$.

We give a semantics of CHTTAs as a labeled transition system where states are pairs (A, tc) with $A \in$ CHTTA$_{\mathcal{A}}^{\Sigma_{Pub}}$ and $tc \in Conf(A)$, and labels are in $\mathbb{R}^{>0} \cup \bigcup_i \Sigma^i \cup \{\tau\}$. In order to simplify the semantics we introduce a notion of structural equivalence for pairs (A, tc), accounting for commutativity and associativity of parallelism. The relation $\approx$ is the least equivalence relation satisfying $(A_1 || A_2, tc_1; tc_2) \approx$

$(A_2||A_1, tc_2; tc_1)$ and $(A_1||(A_2||A_3), tc_1; (tc_2; tc_3)) \approx ((A_1||A_2)||A_3, (tc_1; tc_2); tc_3)$. Moreover, given an untimed parallel configuration $c = c_1; \ldots ; c_n$ we use the following notations: $c \approx \odot$ if $\forall i.c_i = \odot$, and $c \approx \otimes$ if $\exists i.c_i = \otimes \wedge \forall i \neq j.c_j \in \{\odot, \otimes\}$.

Definition 2.3. (Semantics of CHTTAs)
Given $A \in \mathrm{CHTTA}_{\mathcal{A}}^{\Sigma_{Pub}}$, the semantics of a A is the least labeled transition relation $\xrightarrow{\alpha}$ over $\{A\} \times Conf(A)$ closed with respect to structural equivalence and satisfying the following rules:

$$\frac{t \in \mathbb{R}^{>0}}{(A,(c,\nu)) \xrightarrow{t} (A,(c,\nu+t))} \tag{T}$$

$$\frac{(q,\alpha,\phi,B,q') \in \delta \quad v \models \phi \quad q' \notin S}{(\langle A,\mu\rangle,(q,v)) \xrightarrow{\alpha} (\langle A,\mu\rangle,(q',v[B]))} \tag{C1}$$

$$\frac{(q,\alpha,\phi,B,q') \in \delta \quad v \models \phi \quad q' \in S \quad Init(\mu(q')) = (c,\nu)}{(\langle A,\mu\rangle,(q,v)) \xrightarrow{\alpha} (\langle A,\mu\rangle,(q'.c,v[B].\nu))} \tag{C2}$$

$$\frac{(\mu(q),(c,\nu)) \xrightarrow{\alpha} (\mu(q),(c',\nu')) \quad \alpha \in \Sigma_{Pub} \cup \{\tau\}}{(\langle A,\mu\rangle,(q.c,v.\nu)) \xrightarrow{\alpha} (\langle A,\mu\rangle,(q.c',v.\nu'))} \tag{C3}$$

$$\frac{(A_1,(c_1,v)) \xrightarrow{\alpha} (A_1,(c_1',v')) \quad \alpha \in \Sigma_{Pub} \cup \{\tau\}}{(A_1||A_2,(c_1;c_2,v)) \xrightarrow{\alpha} (A_1||A_2,(c_1';c_2,v'))} \tag{P1}$$

$$\frac{(A_1,(c_1,v)) \xrightarrow{a!} (A_1,(c_1',v')) \quad (A_2,(c_2,v')) \xrightarrow{a?} (A_2,(c_2',v''))}{(A_1||A_2,(c_1;c_2,v)) \xrightarrow{\tau} (A_1||A_2,(c_1';c_2',v''))} \tag{P2}$$

$$\frac{c \approx \odot \quad (q,\boxdot,q') \in \delta \quad q' \notin S}{(\langle A,\mu\rangle,(q.c,v.\nu)) \xrightarrow{\tau} (\langle A,\mu\rangle,(q',v))} \tag{Com1}$$

$$\frac{c \approx \odot \quad (q,\boxdot,q') \in \delta \quad q' \in S \quad Init(\mu(q')) = (c',\nu')}{(\langle A,\mu\rangle,(q.c,v.\nu)) \xrightarrow{\tau} (\langle A,\mu\rangle,(q'.c',v.\nu'))} \tag{Com2}$$

$$\frac{c \approx \otimes \quad (q,\boxtimes,q') \in \delta \quad q' \notin S}{(\langle A,\mu\rangle,(q.c,v.\nu)) \xrightarrow{\tau} (\langle A,\mu\rangle,(q',v))} \tag{Ab1}$$

$$\frac{c \approx \otimes \quad (q,\boxtimes,q') \in \delta \quad q' \in S \quad Init(\mu(q')) = (c',\nu')}{(\langle A,\mu\rangle,(q.c,v.\nu)) \xrightarrow{\tau} (\langle A,\mu\rangle,(q'.c',v.\nu'))} \tag{Ab2}$$

where $A = (\Sigma, X, S, Q, q_0, \delta)$ except for rule (T) where A is any CHTTA.

Rule (T) allows the elapsing of time for a generic CHTTA A. We note that the time t is the same for any TTA composing A. Rules (C1) and (C2) describe the behavior of a flat TTA. From a configuration (q, v), the step is performed due to a transition (q, α, ϕ, B, q') such that the condition ϕ is satisfied by v. After the step, the flat TTA is in the configuration composed by state q' and where clocks in B are reset. If q' is a superstate (rule (C2)), then the CHTTA $\mu(q')$ becomes active inside q'. The synchronization step is described by rule (P2). The relation $\approx$ allows CHTTAs that are not neighbors in the parallel composition to communicate. Rules (C3) and (P1) allow expanding the step of a TTA which is a component of a CHTTA. Rule (C3) deals with the hierarchical composition and rule (P1) deals with the parallel composition. The label of the step is either τ or a public channel. Hence, thanks to rule (P2), communication between TTAs in parallel is allowed both for private and public channels, while for TTAs in different superstates the communication is allowed only if the channel is public. Moreover, we note that the step we are expanding cannot be a time step. Hence, since time steps can be performed only by the root, the time elapsed is the same for each TTA composing the CHTTA we are considering.

Each execution of a superstate terminates with either a commit or an abort state. Rules (Com1) and (Com2) deal with the case in which the commit of the superstate takes the TTA to a basic state or to a superstate, respectively, and rules (Ab1) and (Ab2) deal with the case in which the abort of the superstate takes the TTA to a basic state or to a superstate, respectively.

Given a string $w = \alpha_1 \dots \alpha_m$, we will write $(A, (c, \nu)) \overset{w}{\Longrightarrow} (A, (c', \nu'))$ to denote the existence of a sequence of steps $(A, (c, \nu)) \xrightarrow{\alpha_1} \dots \xrightarrow{\alpha_m} (A, (c', \nu'))$. We denote with $\Lambda = \mathbb{R}^{>0} \cup \Sigma_{Pub} \cup \{\tau\}$ the set of labels of the transition system that does not include communcations on private channels. The set Λ is the alphabet of the language accepted by a CHTTA.

Definition 2.4. (Accepted Language)
Let A be a CHTTA, $\mathcal{L}(A)_{\odot} = \{w \in \Lambda^\star \mid (A, Init(A)) \overset{w}{\Longrightarrow} (A, (\odot, \nu'))\}$ and $\mathcal{L}(A)_{\otimes} = \{w \in \Lambda^\star \mid (A, Init(A)) \overset{w}{\Longrightarrow} (A, (\otimes, \nu'))\}$. The language accepted by A is $\mathcal{L}(A) = \mathcal{L}(A)_{\odot} \cup \mathcal{L}(A)_{\otimes}$.

We denote with $\mathcal{L}^p(A)$ the set of all prefixes of elements in $\mathcal{L}(A)$, namely $\mathcal{L}^p(A) = \{w \mid w.w' \in \mathcal{L}(A) \text{ for some } w'\}$. Moreover, we denote with $\mathcal{L}(A, \Sigma_V)$ and $\mathcal{L}^p(A, \Sigma_V)'$ the subsets of $\mathcal{L}(A)$ and $\mathcal{L}^p(A)$, respectively, whose elements are string composed only by symbols in $\mathbb{R}^{>0} \cup \Sigma_V \cup \{\tau\}$.

3. Information Flow in CHTTAs

In this section we will formalize a notion of attacks on system security that are based on an information flow between invisible (private) and visible (public) system activities. We assume that an attacker is just an eavesdropper who can see a part of the system behaviour and tries to deduce from this observation some classified information. In the case of timing attacks, time of occurrences of observed events plays a crucial role, namely, timing of actions represents a fundamental information.

To formalize the attacks we do not divide actions into public and private ones at the system description level, as it is done for example in [8, 4], but we use a more general concept of observation. This concept was recently exploited in [2] and [3] in a framework of Petri Nets and transition systems, respectively, where opacity is defined with the help of observations. First we reformulate a notion of observation function.

Definition 3.1. (Observation)
Let Θ be as set of channels and $\Lambda_\Theta = \mathbb{R}^{>0} \cup \Sigma_\Theta$ be a set of elements called *observables*. Any function $obs : \Lambda^\star \to \Lambda_\Theta^\star$ 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 $obs' : \Lambda \to \Lambda_\Theta \cup \{\epsilon\}$ such that for every $w \in \Lambda^\star$ it holds $obs(w) = obs'(x_1) \dots obs'(x_n)$,
- dynamic if there is a mapping $obs' : \Lambda^\star \to \Lambda_\Theta \cup \{\epsilon\}$ such that for every $w \in \Lambda^\star$ it holds $obs(w) = obs'(x_1).obs'(x_1.x_2) \dots obs'(x_1 \dots x_n)$,

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

In the case of the static observation function each action is observed independently from its context. In case of the dynamic observation function an observation of an action depends on the previous ones, in case of the orwellian and m-orwellian observation function an observation of an action depends on the all and $m-1$ previous and subsequent 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 behaviour and we will alternatively use both the terms observation and 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 a predicate ϕ over sequences. We would like to know whether the observer can deduce the validity of the property ϕ just by observing a sequence from $\mathcal{L}^p(A)$. The observer cannot deduce the validity of ϕ if there are two sequences $w, w' \in \mathcal{L}^p(A)$ such that $\phi(w), \neg\phi(w')$ and the sequences cannot be distinguished by the observer i.e. $obs(w) = obs(w')$. We formalize this concept by the notion of opacity.

Definition 3.2. (Opacity)
Given a CHTTA A, a predicate ϕ over $\mathcal{L}^p(A)$ is opaque w.r.t. the observation function obs if for every sequence $w, w \in \mathcal{L}^p(A)$ such that $\phi(w)$ holds, there exists a sequence $w', w' \in \mathcal{L}^p(A)$ such that $\neg\phi(w')$ holds and $obs(w) = obs(w')$. The set of CHTTAs for which the predicate ϕ is opaque with respect to obs will be denoted by Op^ϕ_{obs}.

The notion of opacity is rather general. With its help many other security properties can be defined (anonymity, non-interference etc.) [3]. On the other side opacity, is undecidable even for the simplest possible observation function, namely for the constant one, and for finite state processes.

Theorem 3.1. Opacity for CHTTA is undecidable.

Proof:
Let us consider an instance of the Post Correspondence Problem with (u_i, v_i) for $i = 1, \ldots, n$. Let us assume that $\{1, \ldots, n\} \subseteq \mathcal{C}_{Pub}$. Let A be a CHTTA consisting of a flat TTA with $\Sigma = \{1!, \ldots, n!\}$, two states q_0 and $\odot$, and a set of transitions $\delta = \{(q_0, \tau, true, \emptyset, \odot)\} \cup \{(q_0, i!, true, \emptyset, q_0) \mid i \in 1, \ldots, n\}$. Let $obs(w) = \epsilon$ for every $w \in \mathcal{L}^p(A)$. We define $\phi(i_1! \ldots i_m!)$ with i_j, $1 \leq j \leq m$, in $\{1, \ldots, n\}$ to be true iff $u_{i_1} \ldots u_{i_m} \neq v_{i_1} \ldots v_{i_m}$. Now, the opacity of ϕ with respect to obs would mean that there exists another sequence $j_1! \ldots j_k!$ such that $\neg\phi(j_1! \ldots j_k!)$ holds, but this would imply $u_{j_1} \ldots u_{j_k} = u_{j_1} \ldots u_{j_k}$, namely a solution of the Post Correspondence Problem. □

Hence there is the need of formalizing a variant of opacity which is decidable but still practically useful, i.e. such that with its help basic security notions could be still expressed.

The undecidability of opacity has two main causes: the first is that the notions of observation functions are very powerful (both dynamic and orwellian ones consider a potentially infinite memory to store actions and subsequently to compute observations), the second is that the predicate ϕ might be difficult to compute. We overcome these obstacles by expressing both an opacity function and predicate ϕ by CHTTAs. First we start with predicate ϕ. We say that the predicate is expressible by an automaton if there exists an automaton such that for every sequence w, $\phi(w)$ holds whenever sequence w is accepted by the automaton. The formal definition is the following.

Definition 3.3. A predicate ϕ over $\Lambda^\star$ is expressible by automaton A_ϕ if $\phi(w)$ holds iff $w \in \mathcal{L}(A_\phi)$. A predicate is *a-expressible* if such an automaton exists.

Example 3.1. Many security concepts are based on an information flow between private and public system activities. Roughly speaking, there is not an information flow if for every sequence of system actions which contains a private action there exists a sequence of actions which does not contain any private action and the both sequences cannot be distinguished by an observer. These concepts can be formalized by opacity when we consider predicate ϕ such that $\phi(w) = true$ iff w contains a privates action. It is easy to see that such the predicate is expressible by the simple automaton A_ϕ which after any action from the set of private actions can reach only states which are final.

Note that the class of a-expressible predicates is very rich and covers more types of predicates that the simple ones mentioned in Example 3.1. By a-expressible predicates we can express rather sophisticated properties which take into account not only a presence of a single private action but also order of actions, their public context, their timing, and so on.

Now we explain how observation function obs can be expressed by an automaton. We assume that sets $\Sigma_\mathcal{C}$ and Σ_Θ (the set of observable actions) have no common elements, and that τ cannot be observed, namely $obs(\tau) = \epsilon$. We will say that the observation function is expressible by automaton if there exists an automaton A such that every sequence accepted by A is obtained from w and o such that $obs(w) = o$. The formal definition is the following, where $x|_y$ denotes the restriction of the sequence x to the set of symbols y.

Definition 3.4. Let $\Theta \cap \mathcal{C} = \varnothing$. An observation function $obs : \Lambda^\star \to \Lambda_\Theta^\star$ is expressible by automaton A_{obs} if for every $w \in \Lambda^\star$ we have $obs(w) = o$ iff there exists $w_o \in \mathcal{L}(A_{obs})$ such that $w_o|_\Lambda = w$ and $w_o|_{\Lambda_\Theta} = o$. We say that an observation function is *a-expressible* if there exists such an automaton.

This definition assumes that an observer (defined by observation function) can always see elapsing of time what is a natural restriction. On the other side a-expressible observation functions cover both static and m-orwellian ones, which represent the most important class of observation functions from the practical point of view.

Now we explain how we define a restricted version of opacity. We assume that predicates ϕ and $\neg\phi$, and that the observation function obs are expressible by A_ϕ, $A_{\neg\phi}$ and A_{obs}, respectively. Moreover, given a CHTTA A, we denote with A_f the CHTTA such that $\mathcal{L}(A_f) = \mathcal{L}^p(A)$. The idea is to compose A_f, A_{obs} and A_{phi} in parallel in order to simultaneously test whether a string w belongs to $\mathcal{L}(A_f)$, assess whether $\phi(w)$ holds and obtain the corresponding observation. If $\phi(w)$ holds, we can replace A_ϕ with $A_{\neg\phi}$ in the parallel composition and test whether there exists w' such that $\neg\phi(w')$ holds with the same observation. In order to allow the three automata of the parallel composition to be executed autonomously (without communicating each other) and in a synchronized manner (in order to ensure that they are working on the same string) we rename all the actions of A_ϕ, $A_{\neg\phi}$ and A_{obs} with different sets of actions and include an additional automaton G to the parallel composition which accepts the language $(\alpha.\alpha_\phi.\alpha_{obs})^\star$ where α_ϕ and α_{obs} are the actions of A_ϕ and A_{obs} corresponding to action α of A_f.

Now let us consider automaton $A_{A,\phi,obs} = (((G||A_f)||A_\phi)||A_{obs})$. From its construction we have that if o belongs to $\mathcal{L}(A_{A,\phi,obs}, \Sigma_\Theta)$ then $o|_{\Lambda_\Theta}$ is an observation of some word w for which $\phi(w)$ holds.

Theorem 3.2. Let $o \in \mathcal{L}(A_{A,\phi,obs}, \Sigma_\Theta)$, there exists $w \in \mathcal{L}^p(A)$ s.t. $\phi(w)$ holds and $obs(w) = o|_{\Lambda_\Theta}$.

Proof:
Let $o \in \mathcal{L}(A_{A,\phi,obs}, \Sigma_\Theta)$. From the construction of $A_{A,\phi,obs}$ we know that for a sequence of type $(\alpha.\alpha_\phi.\alpha_{obs})^\star$ generated by G, all automata A_f, A_ϕ, A_{obs} reached final states and hence the corresponding sequence of type $(\alpha)^\star$ was accepted by A_f (i.e. it belongs to $\mathcal{L}^p(A)$), the corresponding sequence of type $(\alpha_\phi)^\star$ was accepted by A_ϕ (i.e. ϕ holds) and the corresponding sequence from $(\alpha_{obs})^\star$ was accepted by A_{obs}. Sequence o contains also actions τ resulting from the communications among G, A_f, A_ϕ, A_{obs} but they are not taken into account. The proof follows immediately from the definition of A_{obs}. □

Now we define the reduced opacity (r-opacity) property.

Definition 3.5. Let A be a CHTTA. We say that A is *r-opaque* with respect to predicate ϕ expressible by A_ϕ and predicate $\neg\phi$ expressible by $A_{\neg\phi}$, respectively and with respect to observation function *obs* expressible by A_{obs} iff

$$\mathcal{L}((A_{A,\phi,obs}, \Sigma_\Theta)|_{\Lambda_\Theta} \subseteq \mathcal{L}(A_{A,\neg\phi,obs}, \Sigma_\Theta)|_{\Lambda_\Theta}.$$

We denote the set of CHTTAs r-opaque with respect to $\phi, \neg\phi, obs$ as $r\text{-}Op^\phi_{obs}$.

Theorem 3.3. $r\text{-}Op^\phi_{obs} \subset Op^\phi_{obs}$.

Proof:
Let $A \in r\text{-}Op^\phi_{obs}$ and let $w \in \mathcal{L}^p(A)$ such that $\phi(w)$ holds. Then, since $\mathcal{L}((A_{A,\phi,obs}, \Sigma_\Theta)|_{\Lambda_\Theta} \subseteq \mathcal{L}(A_{A,\neg\phi,obs}, \Sigma_\Theta)|_{\Lambda_\Theta}$ we have by Def. 3.5 and Th. 3.2 that there exists $w' \in \mathcal{L}^p(A)$ s.t. $\neg\phi(w)$ holds and $obs(w) = obs(w)$, i.e. $A \in Op^\phi_{obs}$. □

Property $r\text{–}Op^\phi_{obs}$ can be reduced to the language inclusion problem of Timed Automata. First we recall from [11] the following theorem which states that for any CHTTA there is a flat automaton which can perform the same sequences of actions. As a consequence we have that the class of CHTTAs is equivalent to the class of Timed Automata.

Theorem 3.4. Let A be a CHTTA. it holds that $(A, (c_0, v_0)) \overset{w}{\Longrightarrow} (A, (c_n, v_n))$ iff $(A', (c_0, v'_0)) \overset{w}{\Longrightarrow} (A', (c_n, v'_n))$ where $A' = Flat(A)$.

Moreover, for every Timed Automaton A we can construct automaton A^τ such that A accepts word w iff A' accepts word w' which is obtained from w by removing all actions τ. Now, since it is easy to see that the restrictions in Def. 3.5 remove only occurrences of τ, from Th. 3.4 we get the following property.

Theorem 3.5. The property r-opacity can be reduced to the language inclusion problem for automata $A^{\tau}_{A,\phi,obs}$ and $A^{\tau}_{A,\neg\phi,obs}$, i.e to the problem $\mathcal{L}((A^{\tau}_{A,\phi,obs}, \Sigma_\Theta) \subseteq \mathcal{L}(A^{\tau}_{A,\neg\phi,obs}, \Sigma_\Theta)$.

In general, the language inclusion problem for Timed Automata is not decidable [1]. However, it is decidable for many interesting classes of automata.

Theorem 3.6. The property r-opacity is decidable if automaton $A^{\tau}_{A,\neg\phi,obs}$ is deterministic or if automaton $A^{\tau}_{A,\phi,obs}$ has at most one clock or if 0 is the only constant appearing among its clock constraints.

Proof:
According to Th. 3.5, r-opacity can be reduced to the language inclusion problem $\mathcal{L}((A^{\tau}_{A,\phi,obs}, \Sigma_\Theta) \subseteq \mathcal{L}(A^{\tau}_{A,\neg\phi,obs}, \Sigma_\Theta)$. This problem is decidable if automaton $A^{\tau}_{A,\neg\phi,obs}$ is deterministic (see [1]) or if automaton $A^{\tau}_{A,\phi,obs}$ has at most one clock (see [12]) or if 0 is the only constant appearing among its clock constraints of automaton $A^{\tau}_{A,\phi,obs}$ (see [12]). □

Till now we have omitted the discussion about abortions as a tool for performing timing attacks. Suppose that some abortion could be provoked by an intruder. This means that ⊠ becomes an input non-public action and to distinguish different occurrences of such actions we will use indexes. More precisely, we assume that there might be actions that cannot be aborted by the intruder and actions that can be aborted. It is a task of a designer of system A to identify those "weak" places and replace ⊠ by $⊠_i?$. We will call such resulting automaton an *abortion-opened* automaton and we will denote it by A_a. The intruder forces an abortion of a corresponding activity by performing $⊠_i!$. Note that for actions $⊠_i?, ⊠_i!$ only the rule P2 from Definition 2.3 will be applied. Hence we model every intruder as an automaton I that can perform only transitions labeled by $⊠_i!$. We will call an intruder *trivial* either if it cannot abort any action or it can always abort any action. Now we can define r-opacity with respect to some intruder I.

Definition 3.6. Let A be a CHTTA. We say that A is r-opaque with respect to observation automaton A_{obs}, intruder I and automata A_ϕ and $A_{\neg\phi}$ iff $(A_a||I)$ is r-opaque with respect to observation automaton A_{obs} and automata A_ϕ and $A_{\neg\phi}$ for every abortion-opened CHTTA A_a obtained from A.

The set of CHTTAs which are r-opaque with respect to $A_{obs}, I, A_\phi, A_{\neg\phi}$ will be denoted by $r\text{-}Op^{\phi}_{Iobs}$.

The relationship between $r\text{-}Op^{\phi}_{obs}$ and $r\text{-}Op^{\phi}_{Iobs}$ is in the following theorem.

Theorem 3.7. $r\text{-}Op^{\phi}_{Iobs} \subseteq r\text{-}Op^{\phi}_{obs}$.

Proof:
Let $A \in r\text{-}Op^{\phi}_{Iobs}$. That means that $(A_a||I)$ is r-opaque for every abortion-opened CHTTA A_a obtained from A. Hence $(A_a||I) \in r\text{-}Op^{\phi}_{obs}$ for any A_a and hence also for such A_a that intruder I cannot abort any action of A_a, i.e. $(A_a||I)$ and A perform the same sequences of actions, and therefore we get $A \in r\text{-}Op^{\phi}_{obs}$ what proves that $r\text{-}Op^{\phi}_{Iobs} \subseteq r\text{-}Op^{\phi}_{obs}$. □

Note that the inclusion from Theorem 3.7 is proper if the intruder is non-trivial and predicate ϕ expresses, for example, the property that a sequence contains the private action h (see Fig. 2a).

Note that for $r\text{-}Op^{\phi}_{Iobs}$ we have similar property as the one holding for r-opacity (see Theorem 3.6). As an extreme case we might consider a situation when at any time any activity can be aborted. This

Figure 2. Examples of abortion–opened CHTTAs.

might be modeled by replacing every $\boxtimes$ by $\boxtimes?$ and by automaton which can at any time perform $\boxtimes!$. Instead of this we can model this type of attacks simply by putting $\boxtimes$ among public and visible actions. We say that action x is visible with respect to observation function iff $obs(u.x.v) \neq obs(u.v)$.

Definition 3.7. Let A be a CHTTA. We say that A is ar-opaque with respect to observation automaton A_{obs} for which $\boxtimes$ is visible and automata A_ϕ and $A_{\neg\phi}$ iff

$$\mathcal{L}((A_{A,\phi,obs}, \Sigma_\Theta \cup \boxtimes)|_{\Sigma_\Theta \cup \mathbb{R}^{>0}} \subseteq \mathcal{L}(A_{A,\neg\phi,obs}, \Sigma_\Theta \cup \boxtimes)|_{\Sigma_\Theta \cup \mathbb{R}^{>0}}.$$

We denote the set of CHTTAs ar-opaque with respect to $A_{obs}, A_\phi, A_{\neg\phi}$ as $ar\text{-}Op^{\phi}_{obs}$.

Theorem 3.8. $ar\text{-}Op^{\phi}_{obs} \subseteq r\text{-}Op^{\phi}_{Iobs}$.

Proof:
If $A \in ar\text{-}Op^{\phi}_{obs}$ all possible abortions are visible but there is no information flow. Hence, $A \in r\text{-}Op^{\phi}_{Iobs}$ as only some of abortions are visible by I. □

The inclusion in Theorem 3.8 is proper for nontrivial intruders (see Fig. 2b).

As regards timing of actions it is not clear from the above mentioned security concepts whether possible information flow is due to time information contained in observations or not, namely whether there is a danger of timing attack or not. To formalize this concept, let us assume an untimed version obs_t of observation obs i.e. $obs_t(w) = obs_t(w_t) = obs(w_t)$, where w_t is obtained from w by removing all timing information $x \in \mathbb{R}^{>0}$. By $\mathcal{L}^p(A)_t$ we will denote sequences from $\mathcal{L}^p(A)$ from which timing information is removed.

Now we can formalize a notion of being opened to timing attacks.

Definition 3.8. (Opening for Timing Attacks)
Let A be CHTTA. We say that A is opened to timing attacks with respect to predicate ϕ over $\mathcal{L}^p(A)$ and the observation function *obs* if for every sequence w, $w \in \mathcal{L}^p(A)$ such that $\phi(w)$ holds, there exists a sequence $w' \in \mathcal{L}^p(A)$ such that $\neg\phi(w')$ holds and $obs_t(w) = obs_t(w')$ and there exists a sequence $w \in \mathcal{L}^p(A)$ such that $\phi(w)$ holds, but there is not a sequence $w' \in \mathcal{L}^p(A)$ such that $\neg\phi(w')$ holds and $obs(w) = obs(w')$.

In order to define a restricted version of the above notion we consider opacity for the case when elapsing of time is not observed (tr-opacity). It is denoted by $tr\text{-}Op^{\phi}_{obs}$ and it is formally defined as r-opacity (see Def. 3.5) but with both the languages restricted only to $\Theta \cup \mathbb{R}^{>0}$. Decidability of tr-opacity follows immediately from decidability of the language inclusion for untimed languages [1].

Now we can define the property "Restricted Opening for Timing Attacks".

Definition 3.9. (Restricted Opening for Timing Attacks)
Let A be a CHTTA. We say that A is opened to timing attacks with respect to observation automaton A_{obs} and automata A_ϕ and $A_{\neg\phi}$ iff $A \in tr\text{-}Op^{\phi}_{obs}$ and $A \notin r\text{-}Op^{\phi}_{obs}$.

The decidability of this property follows from Th. 3.6 and the decidability of tr-opacity.

4. Opacity of Long–Running Transactions

A *long–running transaction* (LRT) is a composition of *atomic activities* that are either successfully executed (*committed*) or no effect is observed if their execution fails (*aborted*). Partial executions of an LRT are not desirable, and, if they occur, they must be compensated for. Hence, all activities A_i in an LRT have a compensating activity B_i that can be invoked to repair from the effects of a successful execution of A_i if some failure occurs later. Compensations are assumed to always complete their execution successfully (they never abort).

Transactional activities (including compensations) can be composed sequentially and in parallel. Given activities $A_1, \ldots, A_n \in \text{CHTTA}_{\mathcal{A}}^{\Sigma_{Pub}}$ and compensations $B_1, \ldots, B_n \in \text{CHTTA}_{\mathcal{A}}^{\Sigma_{Pub}}$, we can define a language for LRTs as follows:

$$T \ ::= \ A_i \Rsh B_i \quad \Big| \quad T \cdot T \quad \Big| \quad T || T \,.$$

The LRT $A \Rsh B$ denotes the association of the atomic activity A with the compensation B. Given two LRTs T_1 and T_2, with $T_1 \cdot T_2$ we denote their sequential composition and with $T_1 || T_2$ their parallel composition.

In the sequential composition of n transactional activities $A_1 \Rsh B_1 \cdot \ldots \cdot A_n \Rsh B_n$, either the entire sequence $A_1, \ldots, A_n$ is executed or the compensated sequence $A_1, \ldots, A_i, B_i, \ldots, B_1$ is executed for $i < n$. The first case means that all activities in the sequence completed successfully, and the second one stands for the abort of activity A_{i+1}; hence, all the already completed activities $A_1, \ldots, A_i$ are recovered by executing the compensations $B_i, \ldots, B_1$. The sequential composition is associated with a overall compensation to be used for further composition. Such a compensation prescribes the execution of $B_n, \ldots, B_1$ in the order.

In the parallel composition of n transactional activities $A_1 \Rsh B_1 || \ldots || A_n \Rsh B_n$, all the atomic activities are assumed to be executed concurrently, and the whole transaction terminates when all of them complete. If some A_i aborts, then compensation activities should be invoked for the activities that completed successfully. In this latter case, the result of the whole transaction is “abort”. The overall compensation of a parallel composition prescribes the concurrent execution of all the compensations $B_1, \ldots, B_n$.

In [11] we have defined the function $[\![\cdot]\!]$ which maps any LRT into an LRT of the form $A \Rsh B$. Such a function transforms sequential and parallel compositions of LRTs into suitable composition of the CHTTAs of their components. As a consequence, A and B describe the overall behaviour and the overall compensation, respectively, of the considered LRT. Assessing opacity of a predicate on the execution of an LRT can be reduced to assessing opacity of the same predicate on the CHTTAs A and B given by the function $[\![\cdot]\!]$.

Definition 4.1. **(ϕ–opacity)**
Given an LRT T such that $[\![T]\!] = A \Rsh B$ and a predicate ϕ over $\mathcal{L}^p(A) \cup \mathcal{L}^p(B)$, T is *ϕ–opaque* with respect to an observation function *obs* if and only if both ϕ restricted to $\mathcal{L}^p(A)$ and ϕ restricted to $\mathcal{L}^p(B)$ are opaque with respect to obs.

Now, we want to study under which conditions ϕ–opacity may be established compositionally, namely may be deduced by the ϕ–opacity of components.

Given two LRTs T_1 and T_2 such that $[\![T_1]\!] = A_1 \Rsh B_1$, $[\![T_2]\!] = A_2 \Rsh B_2$ and $[\![T_1 \cdot T_2]\!] = A \Rsh B$ as defined in [11], it is easy to see that the languages accepted by A and B can be constructed by those accepted by A_1, A_2, B_1 and B_2, namely $\mathcal{L}(A) = \mathcal{L}(A)_\odot \cup \mathcal{L}(A)_\otimes$ and $\mathcal{L}(B) = \mathcal{L}(B)_\odot \cup \mathcal{L}(B)_\otimes$, where

$$\mathcal{L}(A)_\odot = \mathcal{L}(A_1)_\odot \cdot \mathcal{L}(A_2)_\odot \qquad \mathcal{L}(A)_\otimes = \mathcal{L}(A_1)_\otimes \cup \mathcal{L}(A_1)_\odot \cdot \mathcal{L}(A_2)_\otimes \cdot \mathcal{L}(B_1)$$
$$\mathcal{L}(B)_\odot = \mathcal{L}(B_2)_\odot \cdot \mathcal{L}(B_1)_\odot \qquad \mathcal{L}(B)_\otimes = \varnothing$$

with $\cdot$ the usual concatenation of languages. Similarly, if $[\![T_1 || T_2]\!] = A' \Rsh B'$, it is easy to see that $\mathcal{L}(A') = \mathcal{L}(A')_\odot \cup \mathcal{L}(A')_\otimes$ and $\mathcal{L}(B') = \mathcal{L}(B')_\odot \cup \mathcal{L}(B')_\otimes$, where

$$\mathcal{L}(A')_\odot = \mathcal{L}(A_1)_\odot \oplus \mathcal{L}(A_2)_\odot$$
$$\mathcal{L}(A')_\otimes = \mathcal{L}(A_1)_\otimes \oplus \mathcal{L}(A_2)_\otimes \cup (\mathcal{L}(A_i)_\odot \oplus \mathcal{L}(A_j)_\otimes) \cdot \mathcal{L}(B_i)$$
$$\mathcal{L}(B')_\odot = \mathcal{L}(B_2)_\odot \oplus \mathcal{L}(B_1)_\odot \qquad \mathcal{L}(B')_\otimes = \varnothing$$

with $i, j \in \{1, 2\}, i \neq j$, $\cdot$ the concatenation and $\oplus$ the usual shuffle operator.

We say that a predicate ϕ over the language $\mathcal{L}(A)$ is *decomposable* if and only if $\forall w_1, w_2 \in \mathcal{L}(A).\phi(w_1 \oplus w_2) \implies \phi(w_1) \vee \phi(w_2)$, where $w_1 \oplus w_2$ denotes any possible shuffling of w_1 and w_2 (including $w_1.w_2$), and we say that ϕ is *compositional* if and only if $\forall w_1, w_2 \in \mathcal{L}(A).\phi(w_1) \wedge \phi(w_2) \implies \phi(w_1 \oplus w_2)$.

We show some examples of LRTs whose opacity cannot be established by composition. Let us consider an LRT $T = A \Rsh B$ such that $\mathcal{L}(A)_\odot = \{a!, b!\} \cup \mathbb{R}^{>0}$ and $\mathcal{L}(A)_\otimes = \mathcal{L}(B) = \mathbb{R}^{>0}$. Assume that $\phi(w) = \mathit{true}$ if and only if w contains exactly two occurrences of $a!$, and $\mathit{obs}(x) = x$ for any $x \in \{a!, b!\} \cup \mathbb{R}^{>0}$. It is easy to see that T is ϕ-opaque with respect to *obs*, but ϕ-opacity does not hold for $T \cdot T$. Assume now $\phi'(w) = \mathit{true}$ if and only if w contains at least one occurrence of $a!$, and obs' mapping each pair of consecutive symbols x, y of w to $c!$ if $x = y = a!$ and to ϵ otherwise (for example, $\mathit{obs}'(a!a!b!a!a!a!) = c!c!c!$). Also in this case T is ϕ'-opaque with respect to obs', but ϕ'-opacity does not hold for $T \cdot T$. We note that obs' could be expressed either as a dynamic, or as an orwellian, or as an m-orwellian ($m > 1$) observation. Finally, let us consider LRTs $T_1 = A_1 \Rsh B_1$ and $T_2 = A_2 \Rsh B_2$ such that $\mathcal{L}(A_1)_\odot = \{a!, b!\} \cup \mathbb{R}^{>0}, \mathcal{L}(A_1)_\otimes = \{b!\} \cup \mathbb{R}^{>0}, \mathcal{L}(A_2)_\odot = \{b!\} \cup \mathbb{R}^{>0}$ and $\mathcal{L}(A_2)_\otimes = \mathcal{L}(B_1) = \mathcal{L}(B_2) = \mathbb{R}^{>0}$. Assume ϕ' as above and obs'' such that $\mathit{obs}''(a!) = \epsilon$ and $\mathit{obs}''(b!) = b!$. Both T_1 and T_2 are ϕ'-opaque with respect to obs'', but ϕ'-opacity does not hold for $T_1 \cdot T_2$.

The examples show that we cannot expect that ϕ-opacity is compositional when the predicate ϕ is not decomposable (as in the first example), or the observation function is not static (as in the second example), or one of the CHTTAs of the components, say A, is opaque because it accepts two strings w and w' such that $\phi(w)$ and $\neg\phi(w')$ hold with $\mathit{obs}(w) = \mathit{obs}(w')$, but with $w \in \mathcal{L}(A)_\odot$ and $w' \in \mathcal{L}(A)_\otimes$ (as in the third example). Similar examples can be given to consider the parallel composition of LRTs and LRTs with non–trivial compensations.

We shall show that by restricting predicates, observations and LRTs as the above examples suggest, we are able to prove compositionality of ϕ-opacity. We first need a new concept and two lemmata.

Definition 4.2. (cϕ-opacity)
Given an LRT T with $[\![T]\!] = A \Rsh B$, T is *coherently ϕ-opaque* (cϕ-opaque) with respect to *obs* if and only if for all $w \in \mathcal{L}(A) \cup \mathcal{L}(B)$ such that $\phi(w)$ holds, there exists $w' \in \mathcal{L}(A) \cup \mathcal{L}(B)$ with $\mathit{obs}(w) = \mathit{obs}(w')$ and w, w' are both either in $\mathcal{L}(A)_\odot$, or in $\mathcal{L}(A)_\otimes$, or in $\mathcal{L}(B)_\odot$, or in $\mathcal{L}(B)_\otimes$.

Lemma 4.1. Given a predicate ϕ over a language $\mathcal{L}(A)$, if ϕ is decomposable, then $\neg\phi$ is compositional.

Proof:
ϕ is decomposable means $\forall w_1, w_2 \in \mathcal{L}(A).\phi(w_1 \oplus w_2) \implies \phi(w_1) \vee \phi(w_2)$. Now, assume that $\neg\phi$ is not compositional, namely there exist w_1' and w_2' such that $\neg\phi(w_1') \wedge \neg\phi(w_2')$ holds but $\neg\phi(w_1' \oplus w_2')$ does not. This means that $\phi(w_1' \oplus w_2')$ holds, and by the decomposability of ϕ we obtain that $\phi(w_1') \vee \phi(w_1')$ holds, which is a contradiction. □

Lemma 4.2. If an observation obs is static, $obs(w_1 \oplus w_2) = obs(w_1) \oplus obs(w_2)$.

Proof:
By definition of static observation. □

Theorem 4.1. If ϕ is a decomposable predicate, obs is static observation function and T_1, T_2 are cϕ-opaque, then both $T_1 \cdot T_2$ and $T_1 || T_2$ are cϕ-opaque.

Proof:
Let us assume $[\![T_1]\!] = A_1 \Rsh B_1$, $[\![T_2]\!] = A_2 \Rsh B_2$ and either $[\![T_1 \cdot T_2]\!] = A \Rsh B$ or $[\![T_1 || T_2]\!] = A \Rsh B$. The decomposability of ϕ implies that for all $w \in \mathcal{L}^p(A)$ such that $\phi(w)$ holds w results from the composition of strings $w_1, \ldots, w_n (1 \leq n \leq 3)$ such that $w_1 \in \mathcal{L}^p(A_1) \cup \mathcal{L}^p(A_1) \cup \mathcal{L}^p(B_1) \cup \mathcal{L}^p(B_2)$ and $\phi(w_2)$ holds for some i, $1 \leq i \leq n$. The cϕ-opacity of T_1 and T_2 ensures that for each string w_i such that $\phi(w_i)$ holds there exists w_i' accepted by the same CHTTA and such that $\neg\phi(w_i')$ holds with $obs(w_i) = obs(w_i')$. Now we can reconstruct a string w' by choosing either element w_i or w_i', for each $i \in \{1, \ldots, n\}$, depending whether $\neg\phi(w_i)$ or $\neg\phi(w_i')$ holds, respectively. By Lemma 4.1 we have that $\neg\phi(w')$ holds, and by Lemma 4.2 we have that $obs(w) = obs(w')$. We can reason analogously as regards $\mathcal{L}^p(B)$. Hence, we have the cϕ-opacity of the composition of T_1 and T_2. □

5. Conclusions

In a previous paper we have introduced Communicating Hierarchical Transaction-based Timed Automata (CHTTAs) to model systems performing long–running transactions. In this paper we have introduced for these systems a concept of security which is based on the notion of opacity of CHTTAs. We have given various definitions of opacity and compared their expressiveness. We have studied under which conditions security of long–running transactions can be established compositionally.

References

[1] Alur, R., Dill, D.: A theory of timed automata, *Theoretical Computer Science* **126**, 1994.

[2] Bryans, J., Koutny, M., Ryan, P.: Modelling non-deducibility using Petri Nets. *Proc. of the 2nd Int. Workshop on Security Issues with Petri Nets and other Computational Models (WISP'04)*, 2004.

[3] Bryans, J., Koutny, M., Mazare L., Ryan, P.: Opacity generalised to transition systems, *Proc. of Formal Aspects in Security and Trust (FAST'06)*, LNCS 3866, Springer, Berlin, 2006

[4] Busi, N., Gorrieri, R.: Positive non-interference in elementary and Trace Nets. *Proc. of Appl. and Theory of Petri Nets*, LNCS 3099, Springer, Berlin, 2004.

[5] Dhem, J.F., Koeune, F., Leroux, P.A., Mestre, P., Quisquater, J.J., Willems, J.L.: 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.

[6] Felten, E.W., Schneider, M.A.: Timing attacks on web privacy, *Proc. of the* 7^{th} *ACM Conference on Computer and Communications Security*, 2000.

[7] Goguen, J.A., Meseguer, J.: Security policies and security models, *Proc. of IEEE Symposium on Security and Privacy*, 1982.

[8] Gorrieri, R., Martinelli, F.: A simple framework for real–time cryptographic protocol analysis with compositional proof rules. *Science of Computer Programming* **50**, 2004.

[9] Handschuh, H., Heys, H.M.: A timing attack on RC5, *Proc. Selected Areas in Cryptography*, LNCS 1556, Springer, Berlin, 1999.

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

[11] Lanotte, R., Maggiolo–Schettini, A., Milazzo, P., Troina, A.: Modeling long-running transactions with Communicating Hierarchical Timed Automata, *Proc. of Formal Methods for Open Object-Based Distributed Systems (FMOODS'06)*, LNCS 4037, Springer, Berlin, 2006.

[12] Ouaknine, J., Worrell, J.: On the language inclusion problem for timed automata: closing a decidability gap, *Proc. of the 19th Annual IEEE Symposium on Logic in Computer Science (LICS'04)*, 2004.

[13] Song. D., Wagner, D., Tian, X.: Timing analysis of Keystrokes and SSH timing attacks *Proc. of the 10th USENIX Security Symposium*, 2001.