Analysis for Real-time Intransitive Information Flow Security Properties

Yong Huang
College of Computer Science and Technology, Zhejiang University, China
Email: hy970531@163.com

Lingdi Ping, Shanping Li, Xuezeng Pan
College of Computer Science and Technology, Zhejiang University, China
Email: {ldping, shan, xzpan}@cs.zju.edu.cn

***Abstract*—Real-time information flow security properties such as timed noninterference provide assurances that some time dependent information flows may not become possible. However, with transitive noninterference formulation, it is difficult to deal with intransitive flow policies like channel control and secure downgrading of information with time constraints. In this paper, we introduce the notion of trust domain into *Timed Secure Process Algebra* (tSPA), extending intransitive noninterference to real-time systems. Based on weak timed bisimulation equivalence, some security properties for intransitive flow are reformulated in a real-time setting, in particular one property which is persistent, meaning that if a system is secure then all of its reachable states are secure too. Furthermore, we prove that such persistent intransitive timed property is compositional, which is thus possible to alleviate the state space explosion problem caused by the interleaving of all the possible executions of parallel processes. Finally, we provide one case study showing that it is possible to model and analyze the real-time system through our approach.**

***Index Terms*—real-time information flow, trust domain, intransitive noninterference, timed bisimulation equivalence, timing covert channels**

I. INTRODUCTION

Information flow analysis is an important approach for studying computer systems security problem, and many security properties such as noninterference [1] have been proposed in the literature to capture the intuition of information flow. The aim of noninterference is to prevent any possible flow from the confidential level to the public one. However, the notion of noninterference is based on transitive flow policies. Hence, noninterference cannot deal with intransitive flow policies which are necessary in order to capture common concepts like channel control, or secure downgrading [2, 4]. A typical example for such concepts is that a high confidential domain should not directly communicate with a public network unless the confidential data of the high confidential domain has been properly encrypted by an encryption algorithm, which guarantees that the low level users cannot get the information from the encrypted data.

As a matter of fact, intransitive security policies can be well represented by the notion of intransitive non-interference, which avoid the interference of information flow with regard to the various secure domains of the system. Flow relations among these domains are not transitive. The common intuition behind such policies is that information can be transmitted from the *high* domain to the *low* domain through a *trusted* domain, but not directly from *high* one to *low* one.

Though the first really treatment of intransitive noninterference flow is given by Haigh and Young [3], the better satisfactory formal definition on the same subject has been proposed by Rushby [4], followed by Pinsky [5] and Roscoe [6]. For nondeterministic systems, Mantel [2] presents a novel notion based on event system for intransitive flow. In [7], Bossi proposes a general framework for formalizing different noninterference properties in terms of process algebras. Recently, some researchers (e.g., [8, 17-19]) have focused on mapping intransitive noninterference to a language-based setting. However, the approaches mentioned above are restricted to deterministic systems [4-6, 17-19] or nondeterministic systems [2, 7, 8]. In fact, most of these properties proposed there are based on analysis of information flow that does not take into consideration aspects of time or probability, and therefore they are not useful to analyze the existence of probabilistic or timing covert channels. To overcome this, some significant work has been done in order to extend the study by considering either time (see, e.g., [11, 20-21]) or probability (see, e.g., [9-10, 14-15]).

In this paper we utilize the *Timed Secure Process Algebra* (tSPA, for short) language [11], which is a timed extension of *Secure Process Algebra* (SPA, for short) [12], to model real-time systems. Process algebras [13] provide well-developed theory tool to handle the interactions between a system and its users. Therefore the use of process algebras to analyze information flow security properties has received increased attention in recent years [7]. In [11], Focardi uses tSPA to define several security properties for the analysis of timed

noninterference. In this article we also chose the process-algebra as our analysis tool.

The main contribution of this paper is that we introduce the notion of trusted domain into tSPA and extend the intransitive noninterference theory in a real-time setting. In this paper, our discussion on intransitive noninterference security properties is based on weak timed bisimulation equivalence that provides a more suitable notion of observation to real system. Moreover, we propose one intransitive security property which is persistent, in the sense that if a system is secure then all of its reachable states are secure too. The strongest security property, called *Intransitive Timed Strong Bisimulation Strong Nondeterministic Noninterference* (*I_tSBSNNI*) is powerful to cope with information leakage from real-time system, e.g., timing covert channels.

The rest of the paper is organized as follows. In section II, we recall the tSPA language and the notion of weak timed bisimulation over tSPA terms. In section Ⅲ, the concept of trusted domain is introduced into tSPA, which is used as the tool for analyzing several intransitive noninterference properties of real-time systems. Section Ⅳ illustrates a case study in a time setting. Finally, Section Ⅴ discusses related work and Section Ⅵ gives a conclusion.

II. BASIC NOTIONS

In this section we review the syntax and semantics of the tSPA language and the definition of timed weak bisimulation [11].

In [11], a set of *visible* actions is denoted with $\mathcal{L}$, ranged over by l, such that $\mathcal{L}=I\cup O$ where $I=\{a,b,c\ldots,\}$ is a set of *input* actions and $O=\{\overline{a},\overline{b},\overline{c},\ldots,\}$ is a set of *output* actions. The special type τ denotes an internal action, which is not visible outside the system. Meanwhile, let *tick* be the special action used to model time elapsing. Thus the complete set of *actions* is denoted by $Act=I\cup O\cup\{\tau\}\cup\{tick\}$, ranged over by $\alpha,\beta,\ldots$, while $\mathcal{L}\cup\{\tau\}$, ranged over by a, $b,\ldots$. In order to specify multilevel systems, tSPA partitions the set of visible actions into high level actions and low level ones. Hence, the set of $\mathcal{L}$ is partitioned into two sets, Act_H and Act_L, of high-level and low-level action types.

The syntax of tSPA terms (or processes) is given by the following definition:

$$E ::= 0 \mid \alpha.E \mid E_1+E_2 \mid E_1 \| E_2 \mid E/L \mid E\backslash L \mid l(E) \mid Z$$

Where $\alpha\in Act$, $L\subseteq\mathcal{L}$ and Z is a constant that must be associated with a definition $Z\triangleq E$. Intuitively, 0 represents the terminated or deadlocked term; $\alpha.E$ is the classical prefix operator, and in particular $tick.E$ represents a process willing to let one time unit pass; E_1+E_2 expresses a nondeterministic choice between two processes; $E_1\| E_2$ is the parallel composition of processes that can proceed in an asynchronous way but they must synchronize on complementary actions to make a communication , which is represented by an internal action τ; E/L is the hiding operator that turns actions of type a into internal actions τ; $E\backslash L$ denotes the restriction of the actions with type in the set $L\cup\overline{L}$; $l(E)$ allows process E to wait indefinitely.

Let us call ε the set of tSPA terms, ranged over E and F. Let $sort(E)$ be the set of types of the actions syntactically occurring in E. The sets of high-level terms and low-level ones are defined as $\varepsilon'_H=\{E\in\varepsilon \mid sort(E)\subseteq Act_H\cup\{\tau\}\cup\{tick\}\}$ and $\varepsilon'_L=\{E\in\varepsilon \mid sort(E)\subseteq Act_L\cup\{\tau\}\cup\{tick\}\}$. Note that $\varepsilon'_H\cap\varepsilon'_L=\varnothing$. Since there exists terms that include both high and low level actions, we have the fact that $\varepsilon'_H\cup\varepsilon'_L\subset\varepsilon$. The formal semantics of tSPA terms is given in terms of *labeled transition system* (LTS) $\left(\varepsilon, Act, \xrightarrow{\alpha}_{\alpha\in Act}\right)$, where $\xrightarrow{\alpha}_{\alpha\in Act}$ is the minimal relation between tSPA processes induced by the operational rules of Fig.1.

In tSPA, there exist the following relations between terms: $E\stackrel{\tau}{\Rightarrow}E'$ if $E\xrightarrow{\tau}{}^{*}E'$ (where $\xrightarrow{\tau}{}^{*}$ denotes the reflexive and transitive closure of the relation); for $\alpha\neq\tau$, $E\stackrel{\alpha}{\Rightarrow}E'$ if $E\stackrel{\tau}{\Rightarrow}\xrightarrow{\alpha}\stackrel{\tau}{\Rightarrow}E'$. Let $Der(E)$ be the set of all reachable states from E, i.e., the set of processes that can be reached through the transition relations.

Here we elaborate two features of tSPA. The first one is time determinacy which ensures that the time elapsing never leads a process to two different states (i.e., *tick*-deterministic); another one is the so-called maximal progress assumption, guaranteeing that internal actions have high priority over the elapsing of time. The proof of the following lemma can be easily given by examining the operational rules as shown in Fig.1.

Lemma 1 For every tSPA process E, we have:

- If $E\xrightarrow{tick}E'$ and $E\xrightarrow{tick}E''$ then $E'=E''$;
- If $E\xrightarrow{tick}$ then $E\not\xrightarrow{\tau}$.

In the following, it is helpful to define the so-called weakly time alive processes that do allow time to pass.

Definition 1 A process E is directly weakly time alive iff $E\stackrel{tick}{\Rightarrow}$, while it is weakly time alive iff for all $E'\in Der(E)$, we have E' is directly weakly time alive.

In this paper, the behavioral equivalences we utilize is the timed version of weak bisimulation, which is described in [9] and formally stated as follows.

Definition 2 An equivalence relation $R\subseteq\varepsilon\times\varepsilon$ is a timed weak bisimulation if and only if, whenever, then for all $(E,F)\in R$, we have

- If $E\xrightarrow{a}E'$, then there exists F' s.t $F\stackrel{a}{\Rightarrow}F'$ and $(E',F')\in R$;
- If $E\xrightarrow{tick}E'$, then there exists F' s.t $F\stackrel{tick}{\Rightarrow}F'$ and $(E',F')\in R$.

Two terms $E,F\in\varepsilon$ are weakly timed bisimulation equivalent, denoted $E\approx_t F$, if there exists a timed weak bisimulation R containing the pair (E,F).

$$PREFIXING: \quad \frac{}{\alpha.E \xrightarrow{\alpha} E}$$

$$CHOICE: \quad \frac{E_1 \xrightarrow{a} E'_1}{E_1 + E_2 \xrightarrow{a} E'_1} \quad \frac{E_2 \xrightarrow{a} E'_2}{E_1 + E_2 \xrightarrow{a} E'_2}$$

$$\frac{E_1 \xrightarrow{tick} E'_1 \quad E_2 \xrightarrow{tick} E'_2}{E_1 + E_2 \xrightarrow{tick} E'_1 + E'_2}$$

$$PARALLEL: \quad \frac{E_1 \xrightarrow{a} E'_1}{E_1 \| E_2 \xrightarrow{a} E'_1 \| E_2} \quad \frac{E_2 \xrightarrow{a} E'_2}{E_1 \| E_2 \xrightarrow{a} E_1 \| E'_2}$$

$$\frac{E_1 \xrightarrow{l} E'_1 \quad E_2 \xrightarrow{\bar{l}} E'_2}{E_1 \| E_2 \xrightarrow{\tau} E'_1 \| E'_2}$$

$$\frac{E_1 \xrightarrow{l} E'_1 \quad E_2 \xrightarrow{\bar{l}} E'_2 \ \forall l \in \mathcal{L} \ \neg(E_1 \xrightarrow{l} \wedge E_2 \xrightarrow{\bar{l}})}{E_1 \| E_2 \xrightarrow{\tau} E'_1 \| E'_2}$$

$$CONSTANT: \quad \frac{Z \triangleq E \quad E \xrightarrow{\alpha} E'}{Z \xrightarrow{\alpha} E'}$$

$$RESTRICTION: \quad \frac{E \xrightarrow{\alpha} E'}{E \setminus L \xrightarrow{\alpha} E' \setminus L} (\alpha \notin L \cup \bar{L})$$

$$HIDING: \quad \frac{E \xrightarrow{\alpha} E'}{E \setminus L \xrightarrow{\alpha} E' \setminus L} (\alpha \notin \{L \cup \bar{L} \cup \{tick\}\})$$

$$\frac{E \xrightarrow{l} E' \ l \in L \cup \bar{L}}{E \setminus L \xrightarrow{\tau} E' \setminus L} \quad \frac{E \xrightarrow{tick} E' \ \forall l \in L \cup \bar{L} \ E \not\xrightarrow{\alpha}}{E \setminus L \xrightarrow{tick} E' \setminus L}$$

$$IDLING: \quad \frac{E \not\xrightarrow{tick} \ E \not\xrightarrow{\tau}}{I(E) \xrightarrow{tick} I(E)} \quad \frac{E \xrightarrow{tick} E'}{I(E) \xrightarrow{tick} I(E')} \quad \frac{E \xrightarrow{a} E'}{I(E) \xrightarrow{a} E'}$$

Figure 1. Operational semantics for timed SPA

III. Security Properties

In this section, we extend the intransitive noninterference theory to the tSPA language and analyze intransitive real-time information flow security properties based on timed weak bisimulation equivalence.

A. Extending the tSPA Language

The universal notion of intransitive noninterference is that information can be transmitted from the *high* level to the *low* level via a *trusted* domain, but not directly from *high* to *low*. To extend intransitive noninterference to real-time system, the concept of trusted domain need to be introduced into the tSPA language, which can be performed just by extending tSPA with a set of trusted actions to simulate the behavior of trusted domain. Therefore, we partition the set of $\mathcal{L}$ into the sets Act_H, Act_T and Act_L, of high, trusted and low level action types.

In this paper, our discussion is based on the assumption that the trusted part of the system is so strong that attackers cannot simulate the trusted actions. For example, a trusted computing PC provides enough secure protection to store the encryption keys via the Trusted Platform Module (TPM for short), and the attacker cannot get these keys. So it is reasonable to assume that attackers cannot simulate the trusted actions which are in Act_T. Meanwhile, we assume that the trusted actions cannot be observed by the low level users.

B. Timed Intransitive Noninterference

For real-time systems, the general intuition behind all information flow properties is strictly related to the basic notion of timed noninterference, which says that a system is secure if what the low level users can observe does not rely on what the high level actions can be done. A formalization of such a property in the context of the tSPA language is expressed by *Timed Bisimulation Strong Nondeterministic Noninterference* (*tBSNNI*) [11]. We define the timed intransitive noninterference property by extending *tBSNNI* with trusted actions. Our property allows the low level users to deduce information about high level users' activity through trusted actions.

Definition 3 (*I_tBSNNI: Intransitive Timed Bisimulation Strong Nondeterministic Noninterference*) $E \in \varepsilon$ is *I_tBSNNI* if and only if

$$(E / Act_H) \setminus Act_T \approx_t E \setminus Act_H Act_T$$

It is worthwhile noting that in term E / Act_H, only the high actions are hidden; however, the trusted actions are not. It indicates that the *I_tBSNNI* property does not impose restrictions on the trusted actions and allows the flows from trusted actions to low actions. It is easy to prove that $(E / Act_H) \setminus Act_T$ and $(E / Act_H) \setminus Act_H Act_T$ are isomorphic, because the high actions of E are first hidden to internal actions. Due to this and by *I_tBSNNI* definition, we have E / Act_H and E are timed bisimulation equivalent on low actions ($\setminus Act_H Act_T$). So *I_tBSNNI* does not care about the trusted and high actions, and thus allows the flows from high actions to trusted actions. In summary, the *I_tBSNNI* property can capture the secure information flow from the high level to the low level via the trusted level. We show this by providing a simple example.

Example 1 Consider two processes $E_1 \triangleq h.t_1.tick.l_1.0 + h.t_2.tick.0 + t_3.tick.l_2.0$ and $E_2 \triangleq h.t_1.tick.l_1.0 + l_2.0$. Over this paper, we consider $h, h', \ldots$ to be high level action types, $l, l', \ldots$ to be low level types and $t, t', \ldots$ to be trusted level types. From the low-level perspective, if the action l_1 is executed, then a low user could infer that the action h has been executed. Although the high level interferes with the low level, it is controlled by the trusted level action t_1. Thus *I_tBSNNI* admits this flow. It is easy to prove that E_1 is *I_tBSNNI* secure since $(E_1 / Act_H) \setminus Act_T \approx_t \tau.0 \approx_t E_1 \setminus Act_H Act_T$.

The process E_2 is not *I_tBSNNI* secure, because a low user can deduce that the action *h* has not been executed if the action l_2 is executed. This flow is not controlled by trusted actions of the system. Formally, we have $(E_2 / Act_H) \setminus Act_T \approx_t \tau.0 + l_2.0 \not\approx_t l_2.0 \approx_t E_2 \setminus Act_H Act_T$.

C. Non Deducibility on Compositions

In this section, we present a timed security property stronger than *I_tBSNNI*. We first recall the *Timed Bisimulation-based Non Deducibility on Compositions* (*tBNDC*, for short) security property [11], which is based

on the idea of inspecting if the high users (here we simply refer to as the malicious attackers), modify the low-level response time of the system. Intuitively, a system E is *tBNDC* secure if what a low level user observes on the system must be invariable with respect to the composition with every high level process Π. Contrary to SPA, in the tSPA model, the process Π must be restricted to *weakly time alive* processes that can perform only action in $Act_H \cup \{\tau, tick\}$. Here, we define ε'_H as the set of such processes.

To deal with intransitive flow policies, we present the *Intransitive tBNDC* property. In Section *A*, we have assumed that attackers cannot simulate the behavior of a trusted part of a system. For this reason, like *tBNDC*, only every high level (not including trusted level) attacker $\Pi \in \varepsilon'_H$ is considered in our definition of *Intransitive tBNDC*.

Definition 4 (*I_tBNDC: Intransitive Timed Bisimulation-based Non Deducibility on Compositions*) $E \in \varepsilon$ is *I_tBNDC* if and only if $\forall \Pi \in \varepsilon'_H$,

$$E \setminus Act_H A\,ct_T \approx_t (E \| \Pi) \setminus Act_H A\,ct_T$$

The *I_tBSNNI* property is not strong to detect potential deadlocks due to communications between a system and its high level users, and property *I_tBNDC* overcomes such shortcomings.

Example 2 Let us consider a process $E \triangleq h.h.l.0 + l.0 + h.t.l'.0$. It is provable that E is *I_tBSNNI* secure. Indeed, we have that $(E / Act_H) \setminus Act_T \approx_t l.0 \approx_t E \setminus Act_H\ Act_T$. However, E is not *I_tBNDC* secure. It is sufficient to consider $\Pi \triangleq l(\overline{h}.0)$. In this case the high level process Π executes the high-level action $\overline{h}$, and the process E finds itself in the state $E' \triangleq h.l.0$ (after the first h is executed). Because of the hand-shake nature of tSPA communication, E' must wait for a corresponding external input which could never arrive and a deadlock is caused by the high process Π. It is straightforward to see that $((E \| \Pi) / Act_H) \setminus Act_T \approx_t \tau.0 + l.0 \not\approx_t l.0 \approx_t E \setminus Act_H\ Act_T$.

The following theorem shows that *I_tBNDC* is at least as strong as *I_tBSNNI*.

Theorem 1 *I_tBNDC* $\subset$ *I_tBSNNI*.

Proof: Let E be a process. We need to prove that if $E \in$ *I_tBNDC*, then $E \in$ *I_tBSNNI*. Given $H \subseteq Act_H$ which is the set $\{h_1, \ldots, h_n\}$ containing the types of the high-level actions occurring syntactically in E, we define $\Pi \triangleq \overline{h_1}.\Pi + (\overline{h_2}.\Pi + (\ldots + (\overline{h_{n-1}}.\Pi + \overline{h_n}.\Pi)\ldots)$ as a process which only executes the high-level input actions with type in H. Like the proof of proposition 1 in the appendix of paper [9], we can prove that $(E / Act_H) \setminus A\,ct_T \approx_t (E \| \Pi) \setminus Act_H A\,ct_T$. Due to this and by *I_tBNDC* definition, we have $(E / Act_H) \setminus A\,ct_T \approx_t (E \| \Pi) \setminus Act_H\ A\,ct_T \approx_t E \setminus Act_H Act_T$. Therefore, we have that $(E / Act_H) \setminus Act_T \approx_t E \setminus Act_H Act_T$, which means $E \in$ *I_tBSNNI*.

The inclusion is strict because in Example 2 $E \triangleq h.h.l.0 + l.0 + h.t.l'.0$ is *I_tBSNNI* but not *I_tBNDC*.

It is necessary to notice that, even if E can be *I_tBNDC*, every derivative from E may not be secure. This is the case when uncontrolled flows occur after the first trusted action. Since the *I_tBNDC* property does not inspect all the reachable states, such uncontrolled flows could not be revealed. The following example illustrates this case.

Example 3 Consider the process $E \triangleq h.t_1.l_1.\overline{h}.l_2.0 + t_2.l_3.0$. It is easy to prove that E is *I_tBNDC* secure. If we consider any possible attacker $\Pi \in \varepsilon'_H$, we have that $E \setminus Act_H A\,ct_T \approx_t 0 \approx_t (E \| \Pi) \setminus Act_H Act_T$. Suppose that it reaches the state $\overline{h}.l_2.0$ (after performing action h, t_1 and l_1). Now we have a direct causal relationship between the high level input $\overline{h}$ and the low level output l_2, denoting a direct insecure information flow. However, it is clear that *I_tBNDC* cannot detect this flow because such flow occurs after the trusted action t_1.

D. Persistent Security Properties

Due to the usage of the universal quantification, *I_tBNDC* is difficult to check. In this section, a persistent property stronger than *I_tBNDC*, namely *I_tSBSNNI*, is presented. The basic ideal is to require every state which is reachable by the system still satisfies the *I_tBSNNI* property. If this holds, we can prove that every possible reachable state is warranted to be secure.

Definition 5 (*I_tSBSNNI: Intransitive Timed Strong Bisimulation Strong Nondeterministic Noninterference*) $E \in \varepsilon$ is *I_tSBSNNI* if and only if $\forall E'$ reachable from E, $E' \in I\_tBSNNI$.

That *I_tSBSNNI* is a proper information flow security property of real-time processes, which can expose the potential secure problem that *I_tBSNNI* can not discover.

Example 4 Consider again the *I_tBNNI* process $E_1 \triangleq t_1.tick.l_1.0 + h.h.l_2.0 + l_2.0$. It is clear that $E' \triangleq h.l_2.0$ reachable from E_1 is not *I_tBSNNI* and this gives evidence that E_1 is not *I_tSBSNNI*. We can repair E_1 as follows: $E_1 \triangleq t_1.tick.l_1.tick.0 + h(\tau.0 + h.t.l_2.0) + l_2.0$, which may be proved to satisfy the *I_tSBSNNI* property.

Theorem 2 *I_tSBSNNI* $\subset$ *I_tBNDC*.

Proof: Let E be a process satisfying *I_tSBSNNI*. The proof that E is also *I_tBNDC* is performed by showing that the following relation is a timed weak bisimulation (up to) $R = \{(E' \setminus Act_H Act_T, (E' \| \Pi) \setminus Act_H Act_T) \mid E' \in Der(E), \Pi \in \varepsilon'_H\}$. The proof proceeds by checking all possible cases. Note that if $E' \in Der(E)$ and $E' \xrightarrow{a} E''$ then $E'' \in Der(E)$ and so this condition of the relation R is always satisfied. Let us consider the pair $(E' \setminus Act_H Act_T, (E' \| \Pi) \setminus Act_H Act_T) \in R$. Let us first consider the moves of $E' \setminus Act_H Act_T$.

- If $E' \setminus Act_H Act_T \xrightarrow{a} E'' \setminus Act_H Act_T$ ($a \neq tick$), then $(E' \| \Pi) \setminus Act_H Act_T \xrightarrow{a} (E'' \| \Pi) \setminus Act_H Act_T$, and so $(E'' \setminus Act_H Act_T, (E'' \| \Pi) \setminus Act_H Act_T) \in R$.
- If $E' \setminus Act_H Act_T \xrightarrow{tick} E'' \setminus Act_H Act_T$, then necessarily $E' \xrightarrow{tick}$ and consequently, by lemma 1 $E' \not\xrightarrow{\tau}$.

The following fact holds: for whatever sequence of high actions $\alpha_1,\ldots,\alpha_n$ s.t. $E' \stackrel{\alpha 1}{\Rightarrow} E_1 \ldots E_{n-1} \stackrel{\alpha n}{\Rightarrow} E_n$, we have $E'/Act_H \stackrel{\tau}{\Rightarrow} E_n/Act_H$ and $(E_n/Act_H)\setminus Act_T \approx_t (E'/Act_H)\setminus Act_T \approx_t E'\setminus Act_H Act_T \approx_t E_n \setminus Act_H Act_T$. Since $E' \xrightarrow{tick}$ and $(E'/Act_H)\setminus Act_T \approx_t E'\setminus Act_H Act_T$, we have that there exists a sequence of high actions $\alpha_1,\ldots,\alpha_n$ s.t. $E' \stackrel{\alpha 1}{\Rightarrow} E_1 \ldots E_{n-1} \stackrel{\alpha n}{\Rightarrow} E_n$, $E_n \xrightarrow{\alpha}$ for no action $h \in H$ and $E_n/Act_H \stackrel{tick}{\rightarrow} \stackrel{\tau}{\Rightarrow} E'''/Act_H$, with $(E'''/Act_H)\setminus Act_T \approx_t E'''\setminus Act_H Act_T \approx_t E''\setminus Act_H Act_T$. Thus, if we consider $(E'\|\Pi)\setminus Act_H Act_T$, then, we may have a sequence $(E'\|\Pi)\setminus Act_H Act_T \stackrel{\tau}{\Rightarrow}(E_i\|\Pi')\setminus Act_H Act_T$, with $1 \le i \le n$ s.t. if $E_i \xrightarrow{h}$ then $\Pi \not\xrightarrow{h}$. Now, from the previous fact, it follows that $(E_i/Act_H)\setminus Act_T \approx_t E_i \setminus Act_H Act_T \approx_t E'\setminus Act_H Act_T$ and so $E_i \setminus Act_H Act_T \stackrel{tick}{\Rightarrow} E_{iv}\setminus Act_H Act_T$, with $E_{iv}\setminus Act_H Act_T \approx_t E''\setminus Act_H Act_T$, since $E'\setminus Act_H Act_T \xrightarrow{tick} E''\setminus Act_H Act_T$. This means that $(E'\|\Pi)\setminus Act_H Act_T \stackrel{tick}{\Rightarrow}(E_{iv}\|\Pi'')\setminus Act_H Act_T$. Since $E''\setminus Act_H Act_T \approx_t E_{iv}\setminus Act_H Act_T \approx_t (E_{iv}\|\Pi'')\setminus Act_H Act_T$, the thesis follows.

Similarly, we can prove the moves from $(E'\|\Pi)\setminus Act_H Act_T$ with the same technique above.

- If $(E'\|\Pi)\setminus Act_H Act_T \xrightarrow{\tau}(E''\|\Pi')\setminus Act_H Act_T$, with $E' \xrightarrow{h} E''$ and $\Pi \xrightarrow{h} \Pi'$, then $(E'/Act_H)\setminus Act_T \xrightarrow{\tau}(E''\|\Pi')\setminus Act_H Act_T$. As $E' \in I\_tBSNNI$, we have $(E'/Act_H)\setminus Act_T \approx_t E'\setminus Act_H Act_T$; therefore, to match the τ transition above, there exists E_1 such that $E'\setminus Act_H Act_T \stackrel{\tau}{\Rightarrow} E_1 \setminus Act_H Act_T$ with $E_1 \setminus Act_H Act_T \approx_t (E''/Act_H)\setminus Act_T$; furthermore, also $E'' \in I\_tBSNNI$, therefore $(E''/Act_H)\setminus Act_T \approx_t E''\setminus Act_H Act_T$. In summary, up to weak timed bisimulation, we have that $(E''\setminus Act_H Act_T, (E''\|\Pi)\setminus Act_H Act_T) \in R$.
- If $(E'\|\Pi)\setminus Act_H Act_T \xrightarrow{tick}(E''\|\Pi')\setminus Act_H Act_T$, with $E' \xrightarrow{tick} E''$ and $\Pi \xrightarrow{tick} \Pi'$, then $E'\setminus Act_H Act_T \xrightarrow{tick} E''\setminus Act_H Act_T$. Similarly, we have $(E''\setminus Act_H Act_T, (E''\|\Pi)\setminus Act_H Act_T) \in R$.

From Example 4 it shows that *I_tSBSNNI* is strictly stronger than *I_tBNDC*. Thus we get *I_tSBSNNI* $\subset$ *I_tBNDC*

E. Compositionality of Security Properties

As described in [11], the compositionality property is one of the most interesting features of *tSBSNNI*. In this section, the proposition below shows that the intransitive timed extension of this property also has this desirable feature.

Theorem 3 Whenever $E, F \in I\_tSBSNNI$, then also $E \| F \in I\_tSBSNNI$.

Proof: We need to prove that if $E, F \in I\_tSBSNNI$ then for all of their derivatives E' and F', we have that $E' \| F' \in I\_tBSNNI$, i.e., that $((E'\|F')/Act_H)\setminus Act_T \approx_t (E'\|F')\setminus Act_H Act_T$. Note that **Theorem** 3, also all of the derivatives of a *I_tSBSNNI* process are in *I_tSBSNNI*. Hence, we can actually obtain the result by showing that the following relation is a timed weak bisimulation: $R = \{(((E\|F)/Act_H)\setminus Act_T, (E\|F)\setminus Act_H Act_T) \mid E, F \in I\_tSBSNNI\}$.

We can prove this fact by checking the possible cases. Assume $(((E\|F)/Act_H)\setminus Act_T, (E\|F)\setminus Act_H Act_T) \in R$. Consider the possible moves from $((E\|F)/Act_H)\setminus Act_T$. (Here, we consider only the most interesting cases).

- If $(E\|F)/Act_H)\setminus Act_T \xrightarrow{\tau}(E'\|F')/Act_H)\setminus Act_T$, then if $E \xrightarrow{a} E'$, $F \xrightarrow{\bar{a}} F'$ with $a \in \mathcal{L}$, we have $(E\|F)\setminus Act_H Act_T \xrightarrow{\tau}(E'\|F')\setminus Act_H Act_T$ and $(((E'\|F')/Act_H)\setminus Act_T, (E'\|F')\setminus Act_H Act_T) \in R$.
- If $(E\|F)/Act_H)\setminus Act_T \xrightarrow{\tau}(E'\|F)/Act_H)\setminus Act_T$, with $E \xrightarrow{h} E'$, then since $(E/Act_H)\setminus Act_T \approx_t E\setminus Act_H Act_T$, there exist E_1 s.t. $E\setminus Act_H Act_T \stackrel{\tau}{\Rightarrow} E_1 \setminus Act_H Act_T$ and $E_1 \setminus Act_H Act_T \approx_t (E'/Act_H)\setminus Act_T \approx_t (E_1/Act_H)\setminus Act_T$, the last equality holding because E_1 is *I_tBSNNI*. Hence, we also have $(E\|F)\setminus Act_H Act_T \stackrel{\tau}{\Rightarrow}(E_1\|F)\setminus Act_H Act_T$ and we can now prove that $((E'\|F)/Act_H)\setminus Act_T \approx_t R \approx_t (E_1\|F)\setminus Act_H Act_T$ .it is sufficient to note that $(((E_1\|F)/Act_H)\setminus Act_T, (E_1\|F)\setminus Act_H Act_T) \in R$. Obviously, we get $(((E'\|F)/Act_H)\setminus Act_T, (E'\|F)\setminus Act_H Act_T) \in R$.
- If $(E\|F)/Act_H)\setminus Act_T \xrightarrow{tick}(E'\|F')/Act_H)\setminus Act_T$, then $E \xrightarrow{tick} E'$, $F \xrightarrow{tick} F'$. Since $E, F \in I\_tSBSNNI$, it must hold for all $h \in Act_H \cup Act_T$ that $E \not\xrightarrow{h}$, and the same is also true for F. Hence, $(E\|F)\setminus Act_H Act_T \xrightarrow{tick}(E'\|F')\setminus Act_H Act_T$ and $(((E'\|F')/Act_H)\setminus Act_T, (E'\|F')\setminus Act_H Act_T) \in R$.

Similarly, we can prove the moves from $(E\|F)\setminus Act_H Act_T$ with the same technique above.

- If $(E\|F)\setminus Act_H Act_T \xrightarrow{tick}(E'\|F')\setminus Act_H Act_T$, we have that $E \xrightarrow{tick} E'$ and $F \xrightarrow{tick} F'$. Meanwhile, we get $E\setminus Act_H Act_T \xrightarrow{tick} E'\setminus Act_H Act_T$ and $F\setminus Act_H Act_T \xrightarrow{tick} F'\setminus Act_H Act_T$. Since $E \in I\_tSBSNNI$, we have $(E/Act_H)\setminus Act_T \approx_t E\setminus Act_H Act_T$, from which it follows $(E/Act_H)\setminus Act_T \stackrel{\tau}{\Rightarrow}(E_1/Act_H)\setminus Act_T \xrightarrow{tick} (E_1'/Act_H)\setminus Act_T \stackrel{\tau}{\Rightarrow}(E_2/Act_H)\setminus Act_T \approx_t E'\setminus Act_H Act_T$. for $(F/Act_H)\setminus Act_T$, we have the same result that $(F/Act_H)\setminus Act_T \stackrel{\tau}{\Rightarrow}(F_1/Act_H)\setminus Act_T \xrightarrow{tick}(F_1'/Act_H)\setminus Act_T \stackrel{\tau}{\Rightarrow}(F_2/Act_H)\setminus Act_T \approx_t F'\setminus Act_H Act_T$. At last, we get $((E\|F)/Act_H)\setminus Act_T \approx_t (E/Act_H)\setminus Act_T \| (F/Act_H)\setminus Act_T \stackrel{\tau}{\Rightarrow}((E_1\|F_1)/Act_H)\setminus Act_T \xrightarrow{tick}(E_1'/Act_H)\setminus Act_T \| (F_1'/Act_H)\setminus Act_T \stackrel{\tau}{\Rightarrow}(E_2/Act_H)\setminus Act_T \| (F_2/Act_H)\setminus Act_T \approx_t (E'/Act_H)\setminus Act_T \| (F'/Act_H)\setminus Act_T \approx_t ((E'\|F')/Act_H)\setminus Act_T$; therefore, based on timed bisimulation, we have $(((E'\|F')/Act_H)\setminus Act_T, (E'\|F')\setminus Act_H Act_T) \in R$.

Fig. 2 reports the final diagram summarizing all the intransitive timed noninterference properties we have presented so far.

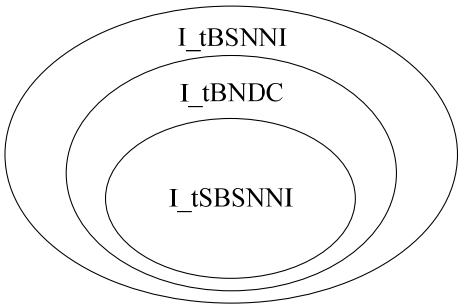

Figure 2. The inclusion diagram for intransitive real-time information flow security properties

Ⅳ. One Case Study

In this section we will provide a case study showing that our approach can be employed to model and analyze security properties of real systems. Here, we consider a timed version of an example from [15], in which a routing mechanism of the OSI network layer is modeled. An internetworking node(termed Interface Message Processor, IMP for short) is modeled by a process which decides on which output link an arriving packet should be sent, depending on the destination of that packet. There are two possible destinations, *a* and *b*, where *a* is occupied to transmit public data (the actions handing packets destined to *a* express the low-level observable behavior of the system), while *b* is utilized to transmit confidential data (the actions handing packets destined to *b* express the high-level observable behavior of the system).

In real application, Virtual Private Network (VPN for short) is usually used by companies or organizations to communicate confidentially over a public network. VPN is a private communications network founded at the network layer. In this section, we model the function of VPN, as well the definition for IMP [15]. Here, confidential packets (those destined to *b*) are properly encrypted first, and thus they are allowed to be transmitted over the low level channel destined to *a*. Fig. 3 shows the algebraic specification of our modified IMP.

Similarly, modified *IMP* is composed of the parallel composition of three components: term *Arrivals* modeling the incoming traffic, term *Router* modeling the core of the IMP, and term *Channels* modeling the outgoing channels. The behavior of each component of *IMP* is explained in [15]. The actions handling packets destined to *a* are low level actions, while the actions handling packets destined to *b* are high level actions. The trusted action *encrypting* represents the encryption phase. Therefore, we have that $Act_H = \{waitb, transmb, avail\_chb, receiveb, sendb, busyb, acceptb, idle\}$, $Act_T = \{encrypting\}$ and all the other actions are low level actions.

$$IMP \triangleq (Arrivals \,||\, Router \,||\, Channels\ ) \setminus \{idle\}$$
$$Arrivals \triangleq Arrivala \,||\, Arrivalb$$
$$Channels \triangleq Channela \,||\, Channelb$$
$$Router \triangleq (Queues \,||\, Switch) \,||\, Idle$$
$$Queues \triangleq Queuea \,||\, Queueb$$
$$Switch \triangleq (Managera \,||\, Routinga) \,||\, (Managerb \,||\, Routingb)$$
$$Arrivalx \triangleq receivex.Arrivalx + waitx.Arrivalx$$
$$Queuex \triangleq \overline{receivex}.\overline{acceptx}.Queuex + \overline{idle}.Queuex$$
$$Managerx \triangleq accetpx.Manager'x + \overline{idle}.Managerx$$
$$Manager'x \triangleq sendx.Managerx + busyx.Manager'x$$
$$Channelx \triangleq avail\_chx.Channelx + transmx.Channelx$$
$$x = \{a, b\}$$
$$Routinga \triangleq \overline{senda}.Routing'a + \overline{avail\_cha}.Routinga$$
$$Routing'a \triangleq \overline{transma}.Routinga + \overline{busya}.Routing'a$$
$$Routingb \triangleq \overline{sendb}.Routing'b + \overline{avail\_chb}.Routingb$$
$$Routing'b \triangleq (\overline{transmb}.Routingb + encrypting.\overline{transma}.Routingb) + \overline{busyb}.Routing'b$$
$$Idle \triangleq l(idle.Idle)$$

Figure 3. Modified IMP model

Intuitively, we observe that a low level packet destined to *a* received via term *Arrivala* which models a low level ingoing channel, handled by term *Router* which is the core of the IMP, and then sent on a low level outgoing channel. The standard noninterference property [15] requires that all the high-level components of the router cannot interference the execution of the actions handling such a low level packet. But in our modified IMP model, the high level component of the router (term $Routing'b$) can introduce trusted interactions between the low level and the high level. By enabling actions *transma*, *transmb* and *encrypting* in term $Routing'b$, we can see that a high level packet can be either directly moved through the high level channel, or first encrypted by the trusted action e*ncrypting* and then delivered over the low level channel. Here, we refine the model by introducing $l(E)$ construct, which is a timeout mechanism to resolve the deadlock problem. It is possible to prove that these potential high-level deadlocks may be used to construct time covert channels.

It is clear that a low level user may observe more packets sent over the low level outgoing channel than packets arrived at the low level ingoing channel, because some high level packets will be encrypted and then sent on the low level outgoing channel. The low level user cannot decrypt these encrypted packets. So our security properties permit such intransitive timed information flow from the high level to the low level through the trusted level. It can be formally verified that process *IMP* is *I_tBNDC* secure, but not *I_tSBSNNI* secure.

Ⅴ. Related Work

In order to provide a formal foundation for the specification and analysis of security policies and models, the notion of *noninterference* has first been introduced by Goguen and Meseguer in [1]. While recognizing the

inability of standard noninterference to model intransitive flow policies, they present several extensions to the basic formulation on the subject based on a *unless* construct in [16]. However, this construct does not deal with intransitive information flow, since it accepted some intuitively insecure systems as being secure. The first satisfactory formal definition of intransitive information flow is introduced by Rushby [4]. Using deterministic state machines as model tool, Pinsky [5] unifies the notions of transitive and intransitive noninterference based on *purge functions*. In [6], Roscoe and Goldsmith provide an approach in terms of deterministic CSP to reveal some insecure flows which are not revealed by Rushby's approach. Besides above approaches for deterministic systems ([6] can deal with severely restricted nondeterminism), reference [2] proposes novel definitions to cope with intransitive flow which is suited for nondeterministic systems. In [7] researchers introduce a general unwinding framework for formalizing traditional and intransitive noninterference properties. Based on weak probabilistic bisimulation equivalence, Jiang [10] extends intransitive noninterference to rephrase some properties for probabilistic systems mentioned in [15].

Related literature on real-time systems includes [11], [20] and [21]. In [20], an approach based on CSP, extended with a special event to denote the passage of time, is used for analyzing cryptographic protocols. The semantic model is rather similar (even if maximal progress is not assumed); we consider that, in the light of [22], also the intransitive noninterference theory developed in this paper can be adapted to that setting. Similarly, the notion in [20] of using a real-time calculus for the analysis of time-dependent properties of cryptographic protocols can be adapted also to our setting. In [21], the model of timed automata [22] is used as a basis for the analysis of information flow properties, based on trace semantics. The information flow properties studied there seem to be less limitative than *tBNDC* for they deal on the recognition of given patterns on the execution of high and low actions. The first generalization of noninterference to real-time systems was defined by Focardi and Gorrieri [11]. Their property is suited to capture time-dependent behaviors. However, it does not deal with the cases such as channel control, information flow filtering, or explicit downgrading. Our security properties (e.g., *I_tBSNNI*, *I_tBNDC*, *I_tSBSNNI*) can overcome this defect.

VI. Conclusion

In this paper, we concentrate on extending the intransitive noninterference theory to the discrete time case. Our main aim is to adopt a timed process algebra style in the definition of intransitive flow properties for real-time system. We first introduce the idea of trusted domain into *Timed Secure Process Algebra* and then may offer some time dependent information flows that can not be revealed by untimed noninterference. Our properties, namely *I_tBSNNI*, *I_tBNDC* and *I_tSBSNNI,* are all based on timed bisimulation equivalence used to distinguish between different process behaviors.

It is worthwhile noticing that the *I_tSBSNNI* security property is persistent, which requires that all of a process's derivatives satisfy the same security property. Moreover, one of the most inspiring features of *I_tSBSNNI* is that it is compositional. This is useful for alleviating the so-called *state-explosion* problem caused by the parallel executions of processes.

Acknowledgment

We would like to thank Dr. Li Jiang and Dr. Guoyong Cai as well as the reviewers for useful discussions and comments. This work was supported in part by a grant from the National High Technology Research and Development Program (863) of china (No. 2006AA01Z431); the Major Program of Science and Technique Foundation of Zhejiang Province (No. 2007C11068&2007C11088).

References

[1] J.A. Goguen and J. Meseguer, "Security policies and security models", *In Proc. 5th IEEE Symposium on security & privacy*, pp. 11–20, 1982.

[2] H. Mantel, "Information flow control and applications – bridging a gap", In *Proc. 10th Symposium on Formal Methods Europe (FME 2001)*, volume 2021 of *LNCS*, pp. 153–172. Springer-Verlag, 2001.

[3] J.T. Haigh and W.D. Young. "Extending the noninterference version of MLS for SAT", *IEEE Transactions on Software Engineering*, SE-13(2), pp. 141–150, February 1987.

[4] J. Rushby, "Noninterference, transitivity, and channel-control security", Technical report, Computer Science Laboratory, SRI International, 1992.

[5] S. Pinsky, "Absorbing covers and intransitive noninterference", In *Proc. 16th IEEE Symposium on Security & Privacy*, pp. 102–113, 1995.

[6] A. Roscoe and M. Goldsmith, "What is intransitive noninterference?", In *Proc. 12th IEEE Computer Security Foundations Workshop (CSFW)*, pp. 226–238, 1999.

[7] A. Bossi, C. Piazza, and S. Rossi, "Modelling downgrading in information flow security", In *Proc. 17th IEEE Computer Security Foundations Workshop (CSFW)*, pp. 187–201, June 2004.

[8] H. Mantel and D. Sands, "Controlled downgrading based on intransitive noninterference", In *Proc. Asian Symp. on Programming Languages and Systems*, volume 3302 of *LNCS*, pp. 129–145. Springer-Verlag, November 2004.

[9] M. Backes and B. Pfitzmann, "Intransitive Non-Interference for cryptographic purposes", In *Proc. of the IEEE Symposium on Security and Privacy (SSP'03)*, pp. 140–152. IEEE Computer Society Press, 2003.

[10] L. Jiang, L.D. Ping, X.Z. Pan, "Extending Intransitive Noninterference with Probabilities in Information Flow Security", in *2nd International Multi-Symposiums on Computer and Computational Sciences*, Iowa, USA. pp. 337–343, IEEE Computer Society Press, 2007.

[11] R. Focardi, R. Gorrieri, and F. Martinelli, "Real-Time Information Flow Analysis", *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 1, pp. 20–35, January 2003.

[12] R. Focardi and R. Gorrieri, "Classification of Security Properties (Part I: Information Flow)", In R. Focardi and R.

Gorrieri, editors, *Proc. of Foundations of Security Analysis and Design (FOSAD'01)*, volume 2171 of *LNCS*, pp. 331–396. Springer-Verlag, 2001.
[13] R. Milner, *Communication and Concurrency*, Prentice-Hall, 1989.
[14] C. Baier, H. Hermanns, "Weak Bisimulation for Fully Probabilistic Processes", *In Proc. Of 9th Int. Conf. on Computer Aided Verification*, volume 1254 of *LNCS*, pp. 119–130. Springer-Verlag, 1997.
[15] A. Aldini, M. Bravetti, R. Gorrieri, "A process-algebraic approach for the analysis of probabilistic noninterference", *Journal of Computer Security*, vol. 12, no. 2, pp. 191–245, 2004.
[16] J. A. Goguen and J. Meseguer, "Unwinding and inference control", in *Proc. 5th IEEE Symposium on Security & Privacy*, pp. 75–86, 1984.
[17] S. Chong and A. C. Myers, "Security policies for downgrading", in *ACM Conference on Computer and Communications Security*, pp. 198–209, October 2004.
[18] N. Broberg and D. Sands, "Flow locks: Towards a core calculus for dynamic flow policies", in *Proc. European Symp. on Programming*, volume 3924 of LNCS, pp. 180–196. Springer-Verlag, 2006.
[19] A. Askarov and A. Sabelfeld, "Gradual release: Unifying declassification, encryption and key release policies", in *Proc. IEEE Symp. on Security and Privacy*, pp. 207–221, May 2007.
[20] S.Schneider, "Analysing time-dependent security properties in CSP using PVS," in *Procs. ESORICS*, vol. LNCS 1895, pp. 222–237, 2000.
[21] R. Lanotte, A. Maggiolo-Schettini, and S. Tini, "Privacy in real-time systems", in *Proc. MTCS'01*, Electronic Notes in Theoretical Computer Science, vol. 52, no. 3, pp. 1–11, 2001.
[22] R. Alur and D. L. Dill, "A theory of timed automata," *Theor. Comput. Sci.*, vol. 126, pp. 183–235, 1994.

Yong Huang received the B.S. and M.S. degrees from Nanjing University of Science and Technology, Nanjing, China, in 2001 and 2004, respectively.

He is is currently a doctoral candidate of computer science, Zhejiang University, Hangzhou, China. His research activity mainly focuses on information security, formal methods, and trusted software development.

Lingdi Ping received the B.S. degree from Zhejiang University, Hangzhou, China, in 1969.

She is currently a Professor of computer science, Zhejiang University, Hangzhou, China. Her research activity mainly focuses on information security, theory of concurrency, and secure operation system.

Shanping Li received the B.S. and Ph.D. degree from Zhejiang University, Hangzhou, China, in 1982 and 1987, respectively.

He is currently a Professor of computer science, Zhejiang University, Hangzhou, China. In 1992, He was a senior researcher engaged in a major program of Oxford University and Leeds University. Now, his research activity mainly focuses on information security, and secure operation system.

Xuezeng Pan received the B.S. degree from Zhejiang University, Hangzhou, China, in 1969.

He is currently a Professor of computer science, Zhejiang University, Hangzhou, China. His research activity mainly focuses on information security, formal methods, VLSI design, and secure operation system.