*Fundamenta Informaticae 72 (2006) 1–14*
*IOS Press*


# Network Information Flow*

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

**Abstract.** A formal model for an analysis of an information flow in interconnection networks is presented. It is based on timed process algebra which can express also network properties. The information flow is based on a concept of deducibility on composition. Robustness of systems against network timing attacks is defined. A variety of different security properties which reflect different security requirements are defined and investigated.

**Keywords**: security, information flow, timing attack, interconnection network

## 1. Introduction

A number of formal definitions of security properties has been proposed and studied in the literature. ”Information flow security” approach plays an important role among them. It requires that there is no flow from private (confidential) to public level of system behaviour. A concept of non-interference formalizes an absence of information flow in multilevel systems. In general, an intruder can exploit the information flow to discover confidential part of the system behaviour. Attacks against system security can exploit besides observing “public” system actions also other activities, for example timing of actions occurrences (hence timing attacks), properties of interconnection networks etc. Particularly timing attacks 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

*Work supported by the grant VEGA 1/3105/06 and APVV-20-P04805.
†Address for correspondence: Institute of Informatics, Comenius University, Mlynska dolina, 842 48 Bratislava, Slovakia

[Ko96]). This idea was developed in [DKL98] were a timing attack against smart card implementation of RSA was conducted. In [HH99], a timing attack on the RC5 block encryption algorithm, in [SWT01] the one against the popular SSH protocol and in [FS00] the one against web privacy are described.

To study security properties various formalisms are used. *Process algebras* are an important class of such formalisms. They successfully describe the behavior of systems as "communicating systems" and they usually abstract away many real properties of existing systems, such as duration and structure of actions, properties of communication networks, distribution of system components, and so on. If one wants to analyze systems with respect to timing attacks in interconnection networks, a special process algebra which is enriched by time and network reasoning has to be used. The aim of this paper is to formalize a notion of "network timing attacks" by means of a particular (dialect of) process algebra NTiCCS (see [GM01]), called Network Security Process Algebra (nSPA), which is based on Milner's CCS. The presented approach is similar to the one which appeared in papers [FG01] and [FGM00]. In the first of them attacks are defined in general, in the second paper discrete time setting is introduced. Here we add also a structure of interconnection network. By this finer, more descriptive calculus we can express and analyze situations which are otherwise very difficult to handle. Note that most of timing attacks strongly depend on network properties.

We will study information flows in the framework of the property called Bisimulation-based Non-Deducibility on Composition (BNDC for short, proposed in [FGM00, FG01]). It is based on the idea of checking systems against all high level potential interactions, representing every possible high level process i.e. a system is BNDC if for every high level user $A$, the low level view of the behaviour of $P$ is not modified (in terms of bisimulation) by the presence of $A$. We will investigate different locations (nestings) of process $A$ in systems with interconnection networks of heterogeneous structure where some connections might be of a limited capacity and an influence of these nestings on the systems security. Moreover, we will investigate also more general case when an attacker might exploit not only one but several high level processes $A_i$ to perform an attack. These process can cooperate (communicate) to exhibit the information flow.

We will prove that in case that process $A$ does not communicate with the system on the top most level, as it is assumed in the original definition of BNDC, but it can be arbitrary nested in the system, more powerful attacks can be performed (Network BNDC). It will be also proved that the power of attacks can be further increased by exploiting several high level processes $A_i$ (Multiple Network BNDC). Hence we get a hierarchy of security properties. It will depend on security requirements, the system accessibility and so on, which of these properties is the most adequate for a concrete application. Moreover, we will study also persistent variant of BNDC properties which require that also all successors have the same property. For the strongest security property (Persistent Multiple Network BNDC) we will present some decidability and compositionality results.

Later we will concentrate on two different questions. First, we will examine how to distinguish attacks for which elapsing of time plays a crucial role (timing attacks) from those for which an information flow has nothing to do with time. This problem has some practical applications. We might consider "off-line" systems to be secure even if they have not some of BNDC properties. Later we will examine security properties for CCS calculus which are based on timed and networked calculi and corresponding BNDC properties. These properties allow formulate some security requirements for abstract CCS specifications before such details as timing of actions and network embedding are specified.

The paper is organized as follows. In Section 2 we describe the language nSPA. In Section 3 we present various information flow based notions of security for both nSPA and CCS processes.

## 2. Network Security Process Algebra

In this section we introduce Network Security Process Algebra (nSPA) which is based on the *Process algebra for network communication* (see [GM01]) and on the *Security Process Algebra* (see [FG01]). In general, the presented calculus allows modeling of two types of communications (via fast networks, for example local buses, and via shared networks with limited throughput, for example optical networks with wavelength-division multiplexing) and modeling of complex networks combining both types of communications. It is assumed that performing communications via a limited capacity network decreases the number of free communication links. It means that a communication of this kind consumes a communication link, and therefore the communication is possible only if there is free link in the network. After completion of the communication the link is free again. We only require that $n$ links can transmit $n$ pieces of information at the same time. We assume that the current number of free links depends on the current number of communicating processes. This means that, during an execution, a process may have to wait for a free communication link. Now, if there are no free links, a networked process cannot perform any communication via the corresponding (sub) network. At this point we depart from a common assumption of timed calculi that communications are instantaneous (minimal idling property), which simplifies problems of communications to an unrealistic level, and instead we assume a *restricted minimal idling property*. This means that, in case of "limited" communications, an internal communication can idle if there is no free communication link in a network. In general, we assume that the duration of an action might be different from the duration of its communication part, i.e. the time for which the action occupies a link. So, this does not lead to a restriction on the maximal number of concurrently running actions or processes.

Now an atomic action will not represent a communication but just its beginning. We will distinguish two kinds of communications. For communications (of concurrently running processes) via a complete network we will use the standard CCS parallel operator $|$. For communications via a network with a limited number of links we will use the new parallel operators $[. \parallel \ldots\ldots \parallel .]^n_B$. To count the number of busy links, we will indicate the total number of links and the current number of busy links. As regards busy links, we need two pieces of information. The number of busy links and the time needed to finish communications. We will indicate this by a multi-set of positive integers $B = \{n_1, n_2, \ldots, n_k\}$. A process $[P_1 \parallel P_2 \cdots \parallel P_m]^n_B$ has $n$ links at its disposal but at the moment $k$ of them are busy. Number $n_i$ indicates how many time units are needed to complete a communication and to release a link. To express run of time and duration of actions we exploit a special action $t$ as for ticks (for details see [GM01]).

To define the language nSPA, we first presuppose the set of atomic action symbols $A$ not containing symbols $\tau$ and $t$, and such that for every $a \in A$ there exists $\overline{a} \in A$ and $\overline{\overline{a}} = a$. We define $Act = A \cup \{\tau\}, Actt = Act \cup \{t\}$. We assume that $a, b, \ldots$ range over $A$, $u, v, \ldots$ range over $Act$, and $x, y \ldots$ range over $Actt$. We suppose that $L$ is a finite multi-set of positive integers. By $|B|$ we will indicate the cardinality of $B$. Let $B = \{n_1, \ldots, n_k\}$. By $B - 1$ we will mean a multi-set $\{n_i - 1 | 1 \leq i \leq k, n_i \in L, n_i - 1 > 0\}$. If $B$ is empty then $B - 1$ is the empty set. Let $c, d : A \to N$ such that $d(a) \geq c(a)$. By $d(a)$ we will indicate the duration of action $a$ and by $c(a)$ we will indicate the duration of a communication part of $a$. The set of nSPA terms over the signature $\Sigma$ is defined by the following BNF notation:

$$P \ ::= \ X \ \mid \ op(P_1, P_2, \ldots P_n) \ \mid \ \mu X P$$

where $X \in Var$, $Var$ is a set of process variables, $P, P_1, \ldots P_n$ are nSPA terms, $\mu X -$ is the binding

construct, $op \in \Sigma$. Assume the signature $\Sigma = \bigcup_{n\geq 0} \Sigma_n$ where

$$\begin{aligned}
\Sigma_0 &= \{Nil\} \\
\Sigma_1 &= \{x \mid x \in A \cup \{t\}\} \cup \{[S] \mid S \text{ is a relabeling function}\} \cup \{\backslash M \mid M \subseteq A\} \\
\Sigma_2 &= \{|, +, [. \parallel .]^n_B\} \\
\Sigma_j &= [. \parallel . \parallel \ldots\ldots \parallel .]^n_B, j > 2
\end{aligned}$$

with the agreement to write unary action operators in prefix form, the unary operators $[S], \backslash L$ in postfix form, and the rest of operators in infix form. To have terms shorter sometimes the term $Nil$ will be omitted. Relabeling functions, $S : Actt \to Actt$ are such that $\overline{S(a)} = S(\bar{a})$ for $a \in A, S(\tau) = \tau$ and $S(t) = t$. Moreover, $M \subseteq A$, $n \in N^+$ and $L$ is a multi-set of positive integers. The set of CCS terms consists of nSPA terms without $t$ action and $[. \parallel . \parallel \ldots\ldots \parallel .]^n_B$ operators. Closed terms are called processes.

A structural operational semantics of terms will be defined in terms of labeled transition systems. The set of terms represents a set of states, labels are actions from $Actt$. The transition relation $\to$ is a subset of nSPA $\times$ $Actt$ $\times$ nSPA. We write $P \xrightarrow{x} P'$ instead of $(P, x, P') \in \to$ and $P \not\xrightarrow{x}$ if there is no $P'$ such that $P \xrightarrow{x} P'$. The meaning of the expression $P \xrightarrow{x} P'$ is that the term $P$ can evolve to $P'$ by performing action $x$, by $P \xrightarrow{x}$ we will denote that there exists a term $P'$ such that $P \xrightarrow{x} P'$. We define the network transition relation as the least relation satisfying the following inference rules:

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

$$\frac{}{Nil \xrightarrow{t} Nil} \quad A3 \qquad \frac{P \xrightarrow{u} P'}{P \mid Q \xrightarrow{u} P' \mid Q} \quad Pa1$$

$$\frac{P \xrightarrow{u} P'}{Q \mid P \xrightarrow{u} Q \mid P'} \quad Pa2 \qquad \frac{P \xrightarrow{a} P', Q \xrightarrow{\overline{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'} \quad Pa3$$

$$\frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q', P \mid Q \not\xrightarrow{\tau}}{P \mid Q \xrightarrow{t} P' \mid Q'} \quad Pa4 \qquad \frac{P \xrightarrow{u} P'}{P + Q \xrightarrow{u} P'} \quad S1$$

$$\frac{P \xrightarrow{u} P'}{Q + P \xrightarrow{u} P'} \quad S2 \qquad \frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q'}{P + Q \xrightarrow{t} P' + Q'} \quad S3$$

$$\frac{P_i \xrightarrow{u} P'_i, \text{for some } i, 1 \leq i \leq m}{[P_1 \parallel \cdots \parallel P_i \parallel \cdots \parallel P_m]^n_B \xrightarrow{u} [P_1 \parallel \cdots \parallel P'_i \parallel \cdots \parallel P_m]^n_B} \quad N1$$

$$\frac{P_i \xrightarrow{a} P'_i, P_j \xrightarrow{\overline{a}} P'_j, \text{for some } i, j, 1 \leq i, j \leq m, |L| < n}{[P_1 \parallel \cdots \parallel P_i \parallel \cdots \parallel P_j \parallel \cdots \parallel P_m]^n_B \xrightarrow{\tau} [P_1 \parallel \cdots \parallel P'_i \parallel \cdots \parallel P'_j \parallel \cdots \parallel P_m]^n_{B \cup \{c(a)\}}} \quad N2$$

$$\frac{P_i \xrightarrow{t} P'_i, \text{for every } i, 1 \leq i \leq m \text{ and } [P_1 \parallel \cdots \parallel P_m]^n_B \not\xrightarrow{\tau}}{[P_1 \parallel \cdots \parallel P_m]^n_B \xrightarrow{t} [P'_1 \parallel \cdots \parallel P'_m]^n_{B-1}} \quad N3$$

$$\frac{P \xrightarrow{x} P'}{P \setminus M \xrightarrow{x} P' \setminus M}, (x \notin M) \quad Res \qquad \frac{P[\mu XP/X] \xrightarrow{x} P'}{\mu XP \xrightarrow{x} P'} \quad Rec \qquad \frac{P \xrightarrow{x} P'}{P[S] \overset{S(x)}{\rightarrow} P'[S]} \quad Rl$$

Here we mention rules that are new with respect to CCS. Axioms $A2, A3$ allow arbitrary idling. Concurrent processes connected via full connection network can idle only if there is no possibility of an internal communication ($Pa4$). A run of time is deterministic ($S3$). If concurrent processes are connected via a network of a limited capacity, they behave similarly except internal communications and $t$ action. Any internal communication increases the number of busy links. It adds to $L$ the duration of the communication part of the corresponding actions ($N2$). By any $t$ step every member of $L$ is decreased by 1. If any number was equal to 1 then after that step it is removed from $L$. This means that a link gets free. The action $t$ can be performed ($N3$) if there is no possibility of an internal communication either between processes $P_i, P_j$ (there are no two processes ready to communicate or there is no free link for such communication) or there is not a possibility of an internal communication "inside" $P_i$ for every $i$ (it can be easily proved by structural induction that $P \xrightarrow{t}$ implies $P \overset{\tau}{\not\rightarrow}$).

**Definition 2.1. (Bisimulation)** A binary relation $\Re \subseteq$ nSPA $\times$ nSPA is called a *bisimulation* if it is symmetric and if $P\Re Q$ and $P \xrightarrow{x} P', x \in Actt$, then there exists $Q'$ such that $Q \xrightarrow{x} Q'$ and $P'\Re Q'$. Two terms $P, Q$ are *bisimilar*, abbreviated $P \sim Q$, if there exists a bisimulation relating $P$ and $Q$.

First some notation is needed: let $x \in Actt$ then $\hat{x} = x$ if $x \neq \tau$ and $\hat{\tau} = \epsilon$ (empty sequence). We will write $P \overset{x}{\Rightarrow} P'$ if $P(\xrightarrow{\tau})^* \xrightarrow{x} (\xrightarrow{\tau})^* P'$. Now we can define weak bisimulation.

**Definition 2.2. (Weak Bisimulation)** A binary relation $\Re \subseteq$ nSPA $\times$ nSPA is called a *weak bisimulation* if it is symmetric and if $P\Re Q$ and $P \xrightarrow{x} P', x \in Actt$, then there exists $Q'$ such that $Q \overset{\hat{x}}{\Rightarrow} Q'$ and $P'\Re Q'$. Two terms $P, Q$ are *weakly bisimilar*, abbreviated $P \approx Q$, if there exists a weak bisimulation relating $P$ and $Q$.

## 3. Information flow in networks

A well known example of information flow based attacks are so called timing attacks. In this case an attacker can see public actions of a system to be attacked and can measure time of their occurrences as well. From this information (s)he tries to deduce performing of private actions. For example, in [FS00] a timing attack on web privacy is described. The attack compromise the privacy of user's web-browsing histories by allowing a malicious web site to determine whether or not the user has recently visited some other, unrelated, web page $w$. An applet is embedded in the malicious web site and is run by user's browser. The applet first performs a request to a file of $w$, and then performs a new request to the malicious site. So, the malicious site can measure the time elapsed between the two requests and it can infers that $w$ was in the cache of the browser (implying that $w$ has been recently visited by the user).

In general, systems respect the property of privacy if there is no leaking of private information, namely there is no information flow from the high (private) level to the low (public) level of their behaviour. This means that the confidential behavior cannot influence the observable one, or, equivalently, no information on the observable behavior permits to infer information on the secret one. In the case of timing attacks, timing of actions play a crucial role. Moreover, real-time behaviour of a system depends

on its embedding to a particular interconnection network. Some communications are fast, some slower and some might even be delayed if there is no free communication link for them. Actually, the above mentioned web attack was based on the property, that cache communications (via local bus) are faster than communications with remote sites. By measuring time (duration) of communications an attacker could deduce whether the site had been visited or not.

Now, to model timing attacks by means of nSPA calculus, first we divide all actions in two groups, namely public (low level) actions $L$ and private (high level) actions $H$ i.e. $A = L \cup H, L \cap H = \emptyset$ (see [FGM00]) and we suppose that renaming functions respect this partition i.e. $f(H) \subseteq H$ and $f(L) \subseteq L$. We need to distinguish processes which can perform only high level actions (high level users). Let $Sort(P) = \{x | P \xrightarrow{s,x} \text{ for } s \in Actt^\star\}$ and $Succ(P) = \{P' | P \xrightarrow{s} P' \text{ for } s \in Actt^\star\}$. The property Bisimulation-based Non-Deducibility on Composition (BNDC for short, proposed in [FGM00, FG01]) is based on the idea of checking the system against all high level potential interactions, representing every possible high level process i.e. a system is BNDC if for every high level user $A$, the low level view of the behaviour of $P$ is not modified (in terms of bisimulation) by the presence of $A$. The idea of BNDC can be formulated as follows.

**Definition 3.1. (BNDC)** $P \in BNDC$ iff for every $A, Sort(A) \subseteq H \cup \{\tau, t\}$

$$(P|A) \setminus H \approx P \setminus H$$

This definition assumes that the high level process ($A$) communicates with the process $P$ at the top most level and without any limitation on throughput of the interconnection network and an observer cannot distinguish between those two systems if they perform only low level actions.

If we consider network processes, this approach seems to be not fully adequate. The network processes might be of a complex network topology and hence for security analysis we have to admit that the high level process $A$ could be arbitrary nested in the interconnection network.

To formalize this concept we define the set $\mathcal{A}-$nSPA of *attack opened* processes. The set of $\mathcal{A}-$nSPA terms over the signature $\Sigma$ is defined by the following BNF notation:

$$P \ ::= \mathcal{A} \ \mid \ X \ \mid \ op(P_1, P_2, \ldots P_n) \ \mid \ \mu X P$$

where $\mathcal{A}$ is a place holder for an attacker and the rest is the same as in case of nSPA terms. Note that nSPA terms are $\mathcal{A}-$nSPA terms and for $\mathcal{A}-$nSPA terms we can use the same transition system which has been defined for nSPA terms where $\mathcal{A}$ behaves like $Nil$. An attack opened nSPA term is called process if it is closed and it contains exactly one occurrence of the place holder $\mathcal{A}$ which does not occur in a subterm of the form $x.F$. When it is clear from the context we will use the same notations for the set of processes and for the set of terms. For every nSPA process $P$ there exist its "openings" for possible timing attacks. To formalize this statement we need some definitions.

**Definition 3.2.** Attack opened process $P'$ is an opening of process $P$ iff $P \sim P'$.

Clearly there might be several "openings" for a given process $P$, but all of them are bisimilar (and bisimilar with $P$ itself). Each opening represents a possible "entrance" (the placeholder $\mathcal{A}$) for an attacker. By an attacker we mean any process $A$ such that $Sort(A) \subseteq H \cup \{\tau, t\}$ and which cannot change network topology, i.e. $A$ does not contain network parallel operators $[. \parallel \ldots\ldots \parallel .]_B^n$ (from now on we will always suppose that attackers $A$ are of this form). Now we define the network variant (nBNDC) of BNDC.

**Definition 3.3. (Network BNDC)** $P \in nBNDC$ iff for every $P'$, where $P'$ is opening of $P$,

$$(P'[A/\mathcal{A}]) \setminus H \approx P \setminus H$$

for all $A, Sort(A) \subseteq H \cup \{\tau, t\}$ where $P'[A/\mathcal{A}]$ represents the substitution of place holder $\mathcal{A}$ by process $A$.

**Example 3.1.** Let

$$\begin{aligned} P &= [h'.t.Nil \parallel \bar{h}'.t.Nil \parallel h.t.Nil \parallel \bar{h}.t.l.Nil]^1_{\{\emptyset\}} \setminus \{h', \bar{h}', h\} \\ P' &= [h'.t.Nil \parallel \bar{h}'.t.Nil \parallel h.t.Nil \parallel \bar{h}.t.l.Nil \parallel \mathcal{A}]^1_{\{\emptyset\}} \setminus \{h', \bar{h}', h\} \end{aligned}$$

and suppose that all communications take exactly one time unit i.e. $c(h) = c(h') = 1$. The opening $P'$ of process $P$ contains the placeholder for a process which communicates with $P$ via a network of capacity 1, i.e. an ethernet like network which can transmit only one piece of information at the moment. Assume that we have attacker $A$, $A = h.t.Nil$ which is nested in the system $P'$, i.e. $P'[A/\mathcal{A}] = [h'.t.Nil \parallel \bar{h}'.t.Nil \parallel h.t.Nil \parallel \bar{h}.t.l.Nil \parallel h.t.Nil]^1_{\{\emptyset\}} \setminus \{h', \bar{h}', h\}$.

Clearly, system $P$ does not have BNDC property since $P \setminus H$ cannot perform the sequence of actions $\tau.\tau.t.l$ while $(P|A) \setminus H$ can perform it. On the other hand if process $A$ is nested then $P'[A/\mathcal{A}] \setminus H$ cannot be on the level of weak bisimulation distinguished from $P \setminus H$. □

**Example 3.2.** Let

$$\begin{aligned} P &= [h_1.t.Nil \parallel \bar{h}_1.t.Nil \parallel \bar{h}_2.t.Nil + (h_3.t.Nil|\bar{h}_3.t.Nil)]^1_{\{\emptyset\}} \setminus \{h_1, \bar{h}_1, h_3, \bar{h}_3\} \\ P' &= [h_1.t.Nil \parallel \bar{h}_1.t.Nil \parallel \bar{h}_2.t.Nil + (h_3.t.Nil|\bar{h}_3.t.Nil) \parallel \mathcal{A}]^1_{\{\emptyset\}} \setminus \{h_1, \bar{h}_1, h_3, \bar{h}_3\} \end{aligned}$$

and suppose that all communications take exactly one time unit i.e. $c(h_1) = c(h_2) = c(h_3) = 1$. The opening $P'$ of process $P$ contains the placeholder for a process which communicates with $P$ via a network of capacity 1. Assume that we have attacker $A'$, $A' = h_2.t.Nil$ which is nested in the system $P'$.

It is easy to check that process $P$ has the BNDC property i.e. $(P|A) \setminus H \approx P \setminus H$ for any high level process $A$. But if process $A'$ is nested in $P'$ then it can be checked that $P'[A/\mathcal{A}] \setminus H$ can perform the sequence of actions $\tau.t\tau.t$ while $P \setminus H$ cannot perform it. Hence $P$ has not nBNDC property. □

The relation between BNDC and nBNDC properties is expressed in the following theorem.

**Theorem 3.1.** nBNDC $\subset$ BNDC.

**Proof:**
Clearly nBNDC $\subseteq$ BNDC since for every process $P$, $P|\mathcal{A}$ is an opening for $P$. From Example 3.2 we have that nBNDC $\subset$ BNDC. □

According to Theorem 3.1 the nBNDC property is a stronger property than BNDC itself. An intruder nested inside a network might be more powerful than an ordinary, non-tested one. Now we show that if we consider not only one but more intruders nested in different places of a network we can model even stronger attacks with respect to nBNDC. First we modify the concept of openings. From now on

an opening might be a term which might contain not only one but also more placeholders $\mathcal{A}_i$. Hence multiply network BNDC property (mnBNDC) will require that behaviour of process $P$ (from the low level point of view) is not changed if a couple of high level intruders are nested in different places of (a network of) system $P$.

**Definition 3.4. (Multiple Network BNDC)** $P \in mnBNDC$ iff for every $P'$ where $P'$ is an opening of $P$, and for every $n \geq 1$,

$$P'[A_1/\mathcal{A}_1, \ldots A_n/\mathcal{A}_n,] \setminus H \approx P \setminus H$$

for all $A_i, Sort(A_i) \subseteq H \cup \{\tau, t\}$ where $P'[A_1/\mathcal{A}_1, \ldots A_n/\mathcal{A}_n,]$ represents the substitution of place holders $\mathcal{A}_i$ by processes $A_i$.

**Example 3.3.** Let

$$\begin{aligned} P &= [h_1.t.Nil \parallel \bar{h}_1.t.Nil \parallel \bar{h}_2.t.Nil \parallel \bar{h}_2.t.Nil]^2_{\{\emptyset\}} \setminus \{h_1, \bar{h}_1, h_2, \bar{h}_2\} \\ P' &= [\mathcal{A}_1 \parallel h_1.t.Nil \parallel \bar{h}_1.t.Nil \parallel \bar{h}_2.t.Nil \parallel \bar{h}_2.t.Nil \parallel \mathcal{A}_2]^2_{\{\emptyset\}} \setminus \{h_1, \bar{h}_1, h_2, \bar{h}_2\} \end{aligned}$$

and suppose that all communications take exactly one time unit i.e. $c(h_1) = c(h_2) = 1$. The opening $P'$ of process $P$ contains two placeholders and processes communicate via a network of capacity 2. Assume that we have process $A, A'$, $A = h_3.t.Nil$, $A' = \bar{h}_3.t.Nil$ which are nested in the system $P'$.

It is easy to check that process $P$ has the nBNDC property. But if processes $A, A'$ are nested in $P'$ then it can be checked that $P'[A/\mathcal{A}_1, A'/\mathcal{A}_2] \setminus H$ can perform the sequence of actions $\tau.t.\tau.t$ while $P \setminus H$ cannot perform it. Hence $P$ has not mnBNDC property. □

**Theorem 3.2.** mnBNDC $\subset$ nBNDC.

**Proof:**
Clearly mnBNDC $\subseteq$ nBNDC. From Example 3.3 we have that mnBNDC $\subset$ nBNDC. □

In [FR02] Focardi and Rossi defined a security property which is stronger property than BNDC (Persistent BNDC) which allows to deal with possibly dynamic attackers and systems "being secure in every state". We can reformulate this concept for the both Network and Multiply Network BNDC.

**Definition 3.5. (Persistent nBNDC, mnBNDC )** $P \in P\_nBNDC$ $(P\_mnBNDC)$ iff for every $P'$, $P' \in Succ(P)$ we have $P' \in nBNDC(mnBNDC)$.

**Example 3.4.** Let

$$\begin{aligned} P &= l_1.t.h.t.l_2.t.Nil + l_1.((h_1.t.l_2.t.Nil|\bar{h}_1.t.Nil) \setminus \{h_1, \bar{h}_1\} \\ &+ (h_2.t.t.Nil|\bar{h}_2.t.t.Nil) \setminus \{h_2, \bar{h}_2\}) \end{aligned}$$

It can be checked that $P \in mnBNDC$ but $P' \notin mnBNDC$ where $P \xrightarrow{l_1} P'$ and $P' = t.h.t.l_2.t.Nil$. □

From the following theorem we have that also persistent variants of network and multiple network BNDC properties are stronger than their non-persistent variants.

**Theorem 3.3.**

$$P\_nBNDC \subset nBNDC,$$

$$P\_mnBNDC \subset mnBNDC.$$

**Proof:**
Clearly $P\_nBNDC \subseteq nBNDC, P\_mnBNDC \subseteq mnBNDC$ From Example 3.4 we have that $P\_nBNDC \subset nBNDC, P\_mnBNDC \subset mnBNDC$. □

The definitions of persistent BNDC properties contain three universal quantifications (over all possible intruders, openings and successors). To avoid them we exploit an idea introduced by Bossi, Focardi, Piazza and Rossi ([BFPR03]). First we introduce a low level observation equivalence (with respect to relation $\asymp$) which relates processes indistinguishable from the low level point of view.

**Definition 3.6. (Equivalence on Low Actions)** Let $\asymp$ be an equivalence relation over processes. We say that two processes $P$ and $Q$ are $\asymp$-equivalent on low actions, denoted by $P \asymp^l Q$, if $P \setminus H \asymp Q \setminus H$.

Now we can recall a notion of generalized unwinding condition. Roughly speaking, it requires that each high level action can be "simulated" in such a way that it is impossible for a low level user to infer which high level actions have been performed. All high level actions are required to be simulated in a way which is transparent to a low level user.

**Definition 3.7. (Generalized Unwinding)** Let $\asymp$ be an equivalence relation and $\longmapsto$ be a binary relation on processes. The unwinding class $(\mathcal{W}, \asymp^l, \longmapsto)$ is defined as

$(\mathcal{W}, \asymp^l, \longmapsto) = \{P \in nSPA | \forall Q \in Succ(P)$ if $Q \stackrel{h}{\rightarrow} R$ then $\exists R'$ such that $Q \longmapsto R'$ and $R \asymp^l R'\}$.

**Theorem 3.4.** $P \in P\_mnBNDC$ iff $P \in (\mathcal{W}, \approx^l, \stackrel{\hat{\tau}}{\Rightarrow})$.

**Proof:**
Let $P \in (\mathcal{W}, \approx^l, \stackrel{\hat{\tau}}{\Rightarrow})$ and let $P' \in Succ(P)$. So if $P' \stackrel{h}{\rightarrow} P_1$ then $P' \stackrel{\hat{\tau}}{\Rightarrow} P_2$ and $P_1 \approx^l P_2$. Let $S = \{(P'_s[A_1/\mathcal{A}_1, \ldots A_n/\mathcal{A}_n,]) \setminus H, P_s \setminus H)\}$ where $P_s \in Succ(P), P'_s$ is its opening and $Sort(A_i) \subseteq H \cup \{\tau, t\}$. It can be checked by case analysis that $S$ is a weak bisimulation up to $\approx$ and then the appropriate processes are weakly bisimilar.

Now, let $P \in P\_mnBNDC$. Then for all $P_s \in Succ(P)$ we have $P_s \in mnBNDC$. Hence for any opening $P'_s$ of $P_s$ and every $A_i$ we have $P'_s[A_1/\mathcal{A}_1, \ldots A_n/\mathcal{A}_n,] \setminus H \approx P_s \setminus H$. Let $P_s \stackrel{h}{\rightarrow} P_1$. Then it easy to see that we can choose an opening and $A = \bar{h}.Nil$ such that $P'_s[A/\mathcal{A}] \setminus H \stackrel{\tau}{\rightarrow} P_1 \setminus H$. Since $P'_s[A/\mathcal{A}] \setminus H \approx P_s \setminus H, P_s \stackrel{\hat{\tau}}{\Rightarrow} P_2 \setminus H$ and $P_1 \setminus H \approx P_2 \setminus H$. □

Due to Theorem 3.4 we can reduce the problem of verifying $P\_mnBNDC$ property to the problem of checking a weak bisimulation up to high level actions. In case of finite state processes, this can be done either adopting the model-checking technique or using a strong bisimulation checker (see [FR02]). Hence we have the following theorem.

**Theorem 3.5.** $P\_mnBNDC$ property is decidable for finite processes.

Processes which satisfy $P\_mnBNDC$ have some important properties with respect to a bottom-up design of complex systems.

**Theorem 3.6. (Compositionality)** Let $P, P_i \in P\_mnBNDC$ for $i = 1, \ldots, n$. Then

$x.P \in P\_mnBNDC$ for $x \in L \cup \{t, \tau\}$

$P_1|P_2 \in P\_mnBNDC$

$[P_1 \parallel \cdots \parallel P_m]^n_B \in P\_mnBNDC$

$P[f] \in P\_mnBNDC$

$P \setminus M \in P\_mnBNDC$

**Proof:**
By Theorem 3.4. We will prove the first three cases which are the most interesting.

(1) Let $P \in P\_mnBNDC$ and let $Q \in Succ(x.P)$ for $x \in L \cup \{t, \tau\}$ such that $Q \stackrel{h}{\rightarrow} R$ for $h \in H$. Clearly, $Q \in Succ(P)$ and then $\exists R'$ such that $Q \longmapsto R'$ and $R \asymp^l R'$. The only successor of $x.P$ which is not also a successor of $P$ is $x.P$ itself but this process cannot perform any high level action $h$. Hence $x.P \in P\_mnBNDC$.

(2) Let $P_1, P_2 \in P\_mnBNDC$ and let $Q \in Succ(P_1|P_2)$. Clearly $Q$ is of the form $Q = P_1'|P_2'$. Let $Q \stackrel{h}{\rightarrow} R$. Without lost of generality we can assume that $R = P_1''|P_2'$. Since $P_1 \in P\_mnBNDC$ and $P_1' \in Succ(P_1)$ there exists $P'''$ such that $P_1' \longmapsto P_1'''$ and $P_1'' \asymp^l P_1'''$. Clearly $Q \longmapsto P_1'''|P_2'$ and $P_1''|P_2' \asymp^l P_1'''|P_2'$.

(3) Let $P_i \in P\_mnBNDC$ for $i = 1, 2, \ldots, n$ let $Q \in Succ([P_1 \parallel \cdots \parallel P_m]^n_B)$. Clearly $Q$ is of the form $Q = [P_1' \parallel \cdots \parallel P_m']^{k'}{}_B$. Let $Q \stackrel{h}{\rightarrow} R$. Without lost of generality we can assume that $R = [P_1'' \parallel \cdots \parallel P_m']^{k'}{}_B$. Since $P_1 \in P\_mnBNDC$ and $P_1' \in Succ(P_1)$ there exists $P'''$ such that $P_1' \longmapsto P_1'''$ and $P_1'' \asymp^l P_1'''$. Clearly $Q \longmapsto [P_1''' \parallel \cdots \parallel P_m']^{k'}{}_B$ (see inference rule $N1$ for operator $[. \parallel \ldots\ldots \parallel .]^n_B$) and $[P_1''' \parallel \cdots \parallel P_m']^{k'}{}_B \asymp^l [P_1'' \parallel \cdots \parallel P_m']^{k'}{}_B$. □

**Remark.** Note that $P\_mnBNDC$ property is not compositional with respect to nondeterministic choice, i.e. there are processes $P_1, P_2 \in P\_mnBNDC$ such that $P_1 + P_2 \notin P\_mnBNDC$. For example let $P_1 = l.Nil, P_2 = h.Nil, l \in L, h \in H$ it is easy to see that $P_1 + P_2 \notin P\_mnBNDC$ but $P_1, P_2 \in P\_mnBNDC$.

Assume that there exist processes $A_i$ such that $P'[A_1/\mathcal{A}_1, \ldots A_n/\mathcal{A}_n,] \setminus H \not\approx P \setminus H$. It is still not clear whether this is the case of "ordinary" or of timing (multiple network) attack. To distinguish these two cases we define t-weak bisimulation ($\approx_t$) in the style of Definition 2.2 but which will neglect not only $\tau$ actions but also $t$ actions. We will write $P \stackrel{x}{\Rightarrow}_t P'$ if $P(\stackrel{\tau,t}{\rightarrow})^* \stackrel{x}{\rightarrow} (\stackrel{\tau,t}{\rightarrow})^* P'$. Now we can define t-weak bisimulation.

**Definition 3.8. (t-weak Bisimulation)** A binary relation $\Re \subseteq$ nSPA $\times$ nSPA is called a *t-weak bisimulation* if it is symmetric and if $P \Re Q$ and $P \stackrel{x}{\rightarrow} P', x \in Actt$, then there exists $Q'$ such that $Q \stackrel{\hat{x}}{\Rightarrow}_t Q'$ and $P' \Re Q'$. Two terms $P, Q$ are *t-weakly bisimilar*, abbreviated $P \approx_t Q$, if there exists a t-weak bisimulation relating $P$ and $Q$.

Now we can formulate the definition of the timing attack.

**Definition 3.9. (Timing Attack)** Processes $A_i, i = 1, \ldots, n$ represent a timing attack for $P$ if

$$P'[A_1/\mathcal{A}_1, \ldots A_n/\mathcal{A}_n,] \setminus H \not\approx P \setminus H$$

and, moreover, there exist $A'_i$, $A'_i \approx_t A_i$ such that

$$P'[A'_1/\mathcal{A}_1, \ldots A'_n/\mathcal{A}_n,] \setminus H \approx P \setminus H.$$

In other words an attack is a timing attack if time matters. Note that the attacks described in Example 3.2 and 3.3 are the timing attacks. To check whether system (process) $P$ is open to timing attacks is even more complex than checking persistent nBNDC and mnBNDC properties due to infinitely many t-weak bisimilar processes to every attack $A_i$. But again in case of finite state processes this complexity can be significantly reduced. We do not need to check the whole infinite set of t-bisimilar processes to processes $A_i$. Let $d$ is the maximal size (by size we mean a number of operators used in a process) of processes $P, A_1, \ldots, A_n$. Then it is enough to check only process $A'_i$ which are built in the following way: first all $t$ actions are removed from $A_i$ and then sequences of $t^k$ actions, $k = 0, \ldots, d$ are gradually put before each action and $Nil$ operator. There is a finite number of such possible attackers ($O((d+1)^{n.(d+1)})$ and it is easy to see that for the second part of Definition 3.9 it is enough to check only these processes. Hence we have the following theorem.

**Theorem 3.7.** It is decidable to check whether processes $A_i, i = 1, \ldots, n$ represent a timing attack for finite process $P$.

It is important to distinguish between timing and non-timing attacks. For example, a system which is off-line or has very slow connection with its environment can be considered to be safe despite the fact that there exists a corresponding timing attack.

nSPA processes can be seen as refinements of timed CCS ones and these again can be seen as refinements of pure CCS ones. If we take the basic specification of a system expressed as CCS process $P$ we might add to it "timed information" and get a timed (say, TiCCS) process $P_t$. Then, again, if we know the implementation (embedding) of the process $P_t$ into a particular network architecture, we can add this information to $P_t$ and get a networked (nSPA) process $P_{nt}$.

For each of these calculi we have the appropriate BNDC property. But if a process seems to be secure on some level (in the sense of the BNDC property) it can lose this property in a lower level where more details of its behaviour can be expressed. In this sense it is meaningful to define a variant of BNDC property which is satisfied by a process if its more concrete variant exhibits that property on a lower level. Hence, we can say that a process $P$ has BNDC$_t$ property if any of its timed versions (i.e. for arbitrary timing of its actions) has $P\_BNDC$ property on the level of TiCCS. And similarly a process $P$ has BNDC$_{nt}$ property if any nSPA process $P_{nt}$ which is its timed and networked refinement has $P\_mnBNDC$ property.

**Definition 3.10.** Let $P$ be CCS process. We say that $P$ has $BNDC_t$ property iff $P_t \in P\_BNDC$ for every timed refinement $P_t$ of $P$, i.e. for every process from which by removing all $t$ actions we get $P$. We say that $P$ has $BNDC_{nt}$ property iff $P_{nt} \in P\_mnBNDC$ for every $P_{nt}$, where $P_{nt}$ is its timed and

network refinement of $P$, i.e by removing all $t$ actions from $P_{nt}$ and replacing all parallel operators by $|$ we get $P$. By BNDC, BNDC$_t$ and BNDC$_{nt}$ we will denote corresponding subsets of CCS processes.

These three properties for CCS processes reflect different requirements for system security. To compare them we cannot work with labeled transition system as it was defined in Section 2. The problem is the rule $Pa4$ which forbids idling of process $P|Q$ in case that the processes $P$ and $Q$ can communicate. We have to replace this rule by new one which allows idling also in this case. The reason is such that in the case when the information about networking is lacking we need to allow idling also when two processes can communicate since there might be an obstacle for their communication - no free communication link.

Let $\rightarrow_a$ be a transition relation which is defined similarly as $\rightarrow$ except the rule $Pa4$ which is replaced by the new rule

$$\frac{P \stackrel{t}{\rightarrow}_a P', Q \stackrel{t}{\rightarrow}_a Q'}{P \mid Q \stackrel{t}{\rightarrow}_a P' \mid Q'}$$

and all bisimulations and corresponding BNDC properties based on the labeled transition system with the relation $\rightarrow_a$ will be denoted by index $a$.

**Theorem 3.8.** Let $P$ be CCS process and $P_t$ its timed refinement. Then $P \approx_{at} P_t$.

**Proof:**
By structural induction. We show that $S = \{(P, P_t) | P \in CCS$ and $P_t$ is timed refinement of $P\}$ is bisimulation. The only interesting case is that of the parallel composition. Let $P = P_1|P_2$ and let $P_1|P_2 \stackrel{x}{\rightarrow}_a P_1' \mid P_2'$. By case analyze it can be shown that every $x$ step can be simulated by $x$ step of $P_t$ and the resulting processes belong to $S$. On the other side if timed refinemt of $P$ preform $x$ step, $P$ can simulate $x$ step as well (thanks to the rules *A2, A3, S3, N3* and the new rule for the parallel composition) and again the resulting processes belong to $S$. □

According to the previous theorem the relation $\approx_{at}$ really represents the abstraction of timed behaviour, i.e. process $P$ behaves in the same way as its timed refinement $P_t$ if a run of time is abstracted. Hence we get the following hierarchy of CCS security properties defined with help of the network and timed caculi.

**Theorem 3.9.** BNDC$_{nta} \subset$ BNDC$_{ta} \subset$ BNDC.

**Proof:**
From the Theorem 3.8 we have BNDC$_{nta} \subseteq$ BNDC$_{ta} \subseteq$ BNDC. The rest of the proof we get by slight modifications of processes from Example 2 and 3. □

As regards decadibility of BNDC$_{nta}$ and BNDC$_{ta}$ by the similar arguments as they were used for Theorem 3.7 we can prove the following theorem.

**Theorem 3.10.** Properties BNDC$_{nta}$ and BNDC$_{ta}$ are decidable for finite state CCS processes.

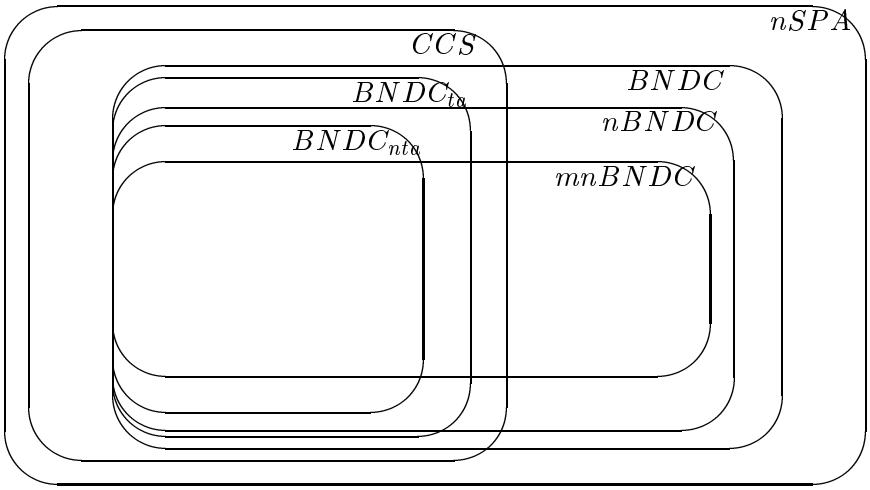


Figure 1. Semantics Hierarchy

Now we have a rich hierarchy (see Fig. 1) of various BNDC based security properties for both CCS and nSPA processes. It depends on security requirements, system accessibility, a phase of system design process and so on which one should be applied for the system design. The final decision depends on a designer which one should be exploited. The designer might have additional knowledge about system architecture, say, possible weak points from security point of view. This knowledge might be exploited in both choice of appropriate BNDC property and choice of the most critical part of the system which should be checked formally.

## 4. Conclusions and further work

Timing attacks can “break” systems which are often considered to be “unbreakable”. More precisely, the attacks usually do not break system algorithms themselves but rather their bad, from security point of view, implementations. For example, such implementations, due to different optimizations, could result in dependency between time of computation and data to be processed, and as a consequence systems might become open to timing attacks. An attacker can deduce from time information also some information about private data, despite the fact and safe algorithms were used. Hence the importance of their study for privacy. In this paper we have presented a formal model which can describe timing attacks in computer networks. This approach enables us to formulate not only a question whether a system is robust with respect to timing attacks but also other questions: how precise must be the measure of time to perform a successful attack, how to modify the system in such a way that attacks will be not possible etc. Note that there exists a software tool for the Process algebra for network communications (see [M94]) which can be used to check some of these properties. Theorem 3.4 allows us to reduce the problem of verifying P_mnBNDC, BNDC$_{nta}$ and BNDC$_{ta}$ properties to the problem of checking relation $\approx^l$. In the case of finite-state processes this can be done by model-checking technique for characteristic mu-calculus formulae for finite-state processes.

In [BMPR05] a concept of secure contexts is presented in the framework of SPA (recall that this process algebra contains no time and network information). The secure contexts are similar to here introduced process openings, which can express both time and network properties. Reasoning about the secure contexts can be interpreted from two points: security for the processes and security for the context. In this paper we have followed only the later of these interpretations.

In general, nBNDC and mnBNDC properties are rather strong as they require that process $P$ is safe against attacks wherever an attackers $A_i$ are placed. If a designer can decide which parts of the system are open for an attacker placing, it is enough to check only the appropriate openings. In other words, a system might be considered to be safe even it does not enjoy mnBNDC or nBNDC property but there are other obstacles to put attackers $A_i$ in such positions from where it could violate system safeness. To check this kind of "limited" nBNDC and mnBNDC properties the designer has to specify the set of "reasonable" openings for $P$. We see our work as a first step towards an analysis of timing attacks on privacy. Further study will concern more efficient decision algorithms, more elaborated "nesting" of attackers in the network, different types of attackers, other compositionally properties, and so on.

## References

[BMPR05] Bossi A., D. Macedonio, C. Piazza and S. Rossi. Information Flow in Secure Contexts. Journal of Computer Security, 13(3), 391-422, IOS Press, 2005.

[BFPR03] Bossi A., R. Focardi, C. Piazza and S. Rossi. Refinement Operators and Information Flow Security. Proc. of SEFM'03, IEEE Computer Society Press, 2003.

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

[FS00] Felten, E.W., and M.A. Schneider: Timing attacks on web privacy. Proc. $7^{th}$ ACM Conference on Computer and Communications Security, 25–32, 2000.

[FG01] Focardi, R. and R. Gorrieri: Classification of Security Properties. Part I: Information Flow. Foundations of Security Analysis and Design, LNCS 2171, 331-396, 2001.

[FGM00] Focardi, R., R. Gorrieri, and F. Martinelli: Information flow analysis in a discrete-time process algebra. Proc. $13^{th}$ Computer Security Foundation Workshop, IEEE Computer Society Press, 2000.

[FR02] Focardi, R. and S. Rossi: Information flow security in Dynamic Contexts. Proc. of the IEEE Computer Security Foundations Workshop, 307-319, IEEE Computer Society Press, 2002.

[GM01] Gruska D.P. and A. Maggiolo-Schettini. Process Algebra for Network Communication, Fundamenta Informaticae, 45(4), 359-378, 2001.

[HH99] Handschuh H. and Howard M. Heys: A timing attack on RC5. Proc. Selected Areas in Cryptography, LNCS 1556, Springer, Berlin, 1999, 306-318.

[Ko96] Kocher P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems. Proc. Advances in Cryptology - CRYPTO'96, LNCS 1109, Springer, Berlin, 1996, 104-113.

[M94] Martinelli, F.: "McTACTL: A Tool to Verify NTiCCS Specifications", *Technical Report TR-21/94*, Dipartimento di Informatica, Università di Pisa, 1994.

[SWT01] Song. D., D. Wagner, and X. Tian: *Timing analysis of Keystrokes and SSH timing attacks.* In 10th USENIX Security Symposium, 2001.