International Journal of Innovative
Computing, Information and Control
Volume 8, Number 1(A), January 2012

ICIC International ©2012 ISSN 1349-4198
pp. 255–267

COMPARISON OF MEASURING INFORMATION LEAKAGE FOR FULLY PROBABILISTIC SYSTEMS

Donghong Sun[1,3], Yunchuan Guo[2], Lihua Yin[2] and Changzhen Hu[3]

[1]Network Research Center
Tsinghua University
Haidian District, Beijing 100084, P. R. China
sundh105@tsinghua.edu.cn

[2]Research Center of Information Security
Institute of Computing Technology
Chinese Academy of Sciences
Beijing 100190, P. R. China

[3]Beijing Institute of Technology
No. 5, South Zhongguancun Street, Haidian District, Beijing 100081, P. R. China

Received August 2010; revised December 2010

Abstract. *Quantifying implicit information leakage is important, especially for fully probabilistic systems (FPS). Although many quantitative methods have been proposed (including methods based on mutual information, on α-mutual information, on relative entropy and on pure probability, etc.), there has been little work analyzing their consistency and accuracy. In order to perform this analysis, these methods must be modeled with a uniform approach. In this paper, a light probabilistic process algebra (PPA-Lite) is presented, and some existing quantitative methods are uniformly characterized by using PPA-Lite. Further, their relationships are analyzed by proof and simulation, respectively. The results show that (1) most of methods concur in determining whether information is leaked; (2) the method based on (α-) mutual information is the most accurate if the distribution of the sent information is known. If not, the method based on relative entropy is the most accurate.*

Keywords: Fully probabilistic system, Information leakage, Probabilistic process algebra, Quantitative measurement

1. **Introduction.** The measurement and quantitative analysis of systems is very important [1, 2, 3, 4, 5]. Especially for fully probabilistic systems (FPS), quantifying the implicit leakage of information is of great importance. Briefly, a process A can implicitly leak information to a process B if all the following conditions are satisfied: (1) A shares resources with B, (2) A can modify some attributions of the resources, and (3) B can observe these modifications. The TCSEC (Trusted Computer Security Evaluation Criteria, also known as the Orange Book, issued by the United States' National Security Agency) requires analysis of implicit information leakage for a system to be classified as B2 or above. Also, CC (Common Criteria for Information Technology Security Evaluation) and GB17859 have a similar stipulation.

Ideally, it should be guaranteed that any confidential information cannot be implicitly leaked. This is, however, rarely satisfied by any system. In fact, many systems (such as operating systems [6, 7], database systems [8, 9], network protocols [10, 11, 12] and hardware systems [13]) can implicitly leak information. Even a system which requires a high level of security may implicitly leak information. Taking an ATM (Automatic Teller Machine) as an example, we assume that a thief steals a bank card, and that he has to

guess its PIN number in order to withdraw cash from the ATM. If he fails to guess the correct PIN number after three attempts, the ATM will dispense no cash, so the ATM is secure. However, from the point of view of information leakage, the ATM has implicitly leaked information because the thief obtains the following knowledge: the guessed PIN number is not correct. In fact, almost all systems can implicitly leak information, so it is a reasonable assumption that a system is secure if its amount of information leakage is below a certain threshold. This requires us to quantify information leakage.

In order to quantify information leakage, methods and the corresponding metrics are needed. TCSEC and GB17859 use bandwidth as a measure of implicit information leakage. On the surface, this seems reasonable. However, adopting the bandwidth as a measure is hard to implement, because the bandwidth of information leakage of a given system is affected by various factors, e.g., noise and delay. This causes difficulty in accurately measuring the bandwidth. As shown in [7], there has been little work in successfully measuring implicit information leakage. Apart from bandwidth-based methods, many other methods have been proposed. These fall into three categories: methods based on information theory, methods based on pure probability and the other methods.

The measurements based on information theory include five sub-categories: entropy-based methods, relative-entropy-based methods, mutual-information-based methods, channel-capacity-based methods and α-mutual information-based methods. To the best of our knowledge, Denning [14] is the first work to adopt entropy to measure information leakage. Following the work of Denning, many information-theory-based methods have been put forward. Clark, Millen and Gray [15, 16, 17, 18, 19, 20] respectively adopt entropy, mutual information and capacity channel methods to analyze information leakage. In contrast with [14, 16, 17], M. Clarkson [21] proposed a new perspective to quantitatively measure information leakage by analyzing the attack's belief. For methods based on pure probability, Pierro [22] believes that if another party cannot distinguish the two subjects, then the amount if information leaked is 0. Similarly, Aldini [23, 24] identities and quantifies information leakage by probabilistic bi-simulation.

In addition to the above, there are other studies. For example, Lowe [25] measures information leakage by counting refusals. Malacaria [26] also provides a framework for quantifying how much information can be leaked with a given model. [15, 27] provide an overview.

Although many novel ways to measure implicit information leakage have been proposed, their accuracy varies. For example, Moskowitz [28] points out that it is not accurate to measure information leakage only using channel capacity. And Clark [15] shows that there is a problem with Denning's way.

Thus, while many methods have been put forward, very little work analyzes their accuracy. In addition, much research concentrates on a non-deterministic system rather than a fully probabilistic system (FPS) and measurement for an FPS is more difficult. In this paper, we analyze the consistency and accuracy of different approaches which can be used to measure implicit information leakage for an FPS. First, a light probabilistic process algebra (PPA-Lite) is presented and employed to model an FPS, and the existing qualitative methods (including methods based on mutual information, on α mutual information, on relative entropy and on pure probability, etc.) are formally modeled by using the PPA-Lite. Further, their relationships are analyzed by proof and simulation, respectively. The results show that (1) most of methods concur in determining whether information is leaked; (2) the method based on (α-) mutual information is the most accurate if the distribution of the sent information is known. If not, the method based on relative entropy becomes the most accurate.

This paper is structured as follows. Section 2 intuitively introduces implicit information leakage in an FPS and presents some related concepts from information theory. Section 3 discusses the syntax and the formal semantics of PPA-Lite which will be used to model an FPS. Section 4 discusses how to model an FPS and its information leakage via PPA-Lite. In Section 5, PPA-Lite is used to formally and uniformly describe the existing measurement methods of implicit information leakage. Different methods are theoretically compared and a simulation is given in Section 6. Finally, we summarize and conclude the paper.

2. **Preliminaries.**

In this section, we will review some notions of information theory which will be used in the following and then give an intuitive description of implicit information leakage in an FPS.

2.1. **Probabilistically implicit information leakage.**

We provide here an intuitive description of probabilistic and implicit information leakage. Consider a system which has two types of users: high security level users (HU) and low security level users (LU), and a security policy stipulating that HU are prohibited from transmitting any information to LU.

If HU and LU share some resources, HU may probabilistically change their attributions of these resources and LU can observe this change. In this way, HU can indirectly transmit messages to LU, and information can be probabilistically and implicitly leaked.

Example 2.1. *Assume a probabilistic scheduling algorithm, which has two types of users HU and LU. HU is the scheduler who can schedule the transaction A and B, and LU can observe the scheduled A and B. HU and LU have a prior agreement: if HU wants to transmit bit 0 to LU, he will schedule A with probability 0.7 and B with 0.3; otherwise if HU wants to transmit bit 1, he will schedule A with probability 0.1 and B with probability 0.9. So, if LU observes that A and B are scheduled with probability 0.8 and 0.2 respectively, he can infer that the transmitted bit is 0; and if LU observes that A and B are scheduled with probability 0.2 and 0.8 respectively, he can infer that the transmitted bit is 1; so the scheduling algorithm can leak information.*

2.2. **Related notions** [30, 31]**.**

Let X, Y be random variables defined over space $\mathcal{X}$, $\mathcal{Y}$, respectively. Let x and y represent values of these variables, with probability mass function $p(x) = Pr\{X = x\}$.

Definition 2.1. *The entropy H(X) of a discrete random variable X is defined by*

$$H(X) = -\sum_{x\in\mathcal{X}} p(x)\log_2 p(x).$$

The log is to the base 2 and entropy is expressed in bits.

Definition 2.2. *Given two probability mass functions $p(x)$ and $q(x)$, the relative entropy or Kullback-Leibler distanced $D_{KL}(p,q)$ is defined as*

$$D_{KL}(p,q) = \sum_{x\in\mathcal{X}} p(x)\log\frac{p(x)}{q(x)}$$

In the above definition, we follow the convention that $0\log\frac{0}{0} = 0$, $0\log\frac{0}{q} = 0$ and $p\log\frac{p}{0} = \propto$.

Definition 2.3. *Let X, Y be two random variables with a joint probability mass function $p(x,y)$ and marginal probability mass functions $p(x)$ and $p(y)$. The mutual information $D_{MI}(X;Y)$ and α-mutual information $D_{\alpha MI}(X;Y)$ are respectively defined as:*

$$D_{MI}(X;Y) = D_{KL}(p(x,y), p(x)p(y))$$

$$D_{\alpha MI}(X;Y) = \frac{1}{\alpha - 1} \log \left(\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \frac{p(x,y)^{\alpha}}{((px)p(y))^{\alpha - 1}} \right)$$

where $\alpha > 0$ and $\alpha \neq 1$.

3. **Probabilistic Process Algebra.** As described in the above section, probabilistic information leakage almost cannot be avoided in a system. This requires us to quantify its information leakage. To achieve this goal, many quantification methods have been proposed, but the accuracy of these methods varies. For example, in Example 2.1, the amount of information leakage based on relative entropy is 1:49 bits, but the amount based on mutual information is 0.29 bits (We assume the proportion of 0's and 1's is the same). In order to compare these methods, the first thing we must do is to uniformly model them. Although many methods can be used to model information leakage, it is not easy to implement them. In this section, we analyze the natural features of probabilistic information leakage, eliminate some calculus which is not suited for probabilistic information leakage and present a light probabilistic process algebra (PPA-Lite).

3.1. **Syntax.** Let $\mathcal{A}$ be a set of observable and countable actions, composed of two disjoint subsets of actions: $\mathcal{H} = \{h, h_0, h_1, \ldots\}$ (called the high-level actions, which are only executed by HU) and $\mathcal{L} = \{l, l_0, l_1, \ldots\}$ (called the low-level actions, which are only executed by LU). That is, $\mathcal{A} = \mathcal{H} \cup \mathcal{L}$ and $\mathcal{H} \cap \mathcal{L} = \phi$. In the following, we use a, b, c to denote a single observable action and τ ($\tau \notin \mathcal{A}$) to denote an unobservable action. Let $Act = \mathcal{A} \cup \tau$. The syntax of PPA-Lite is defined in Figure 1.

P::=	x	variable
	$\pi.P$	prefix
|	$\sum_{i \in I} [p_i] P_i$	probabilistic choice
|	$P \backslash A$	restriction
|	P/A	hiding
|	$rec\ x.P$	recursion
|	$\underline{0}$	stop

FIGURE 1. Syntax

In Figure 1, $\pi \in Act$, $A \subseteq \mathcal{A}$ and $\pi.P$ represent the execution of π followed by P; $\sum_{i \in I} [p_i] P_i$ is the probabilistic choice, where $p_i \in (0,1)$ is a probability with the constraint of $\sum_{i \in I} p_i = 1$; $P \backslash A$ restricts the execution of actions in A; P/A hides the execution of actions in A; $\underline{0}$ denotes a stop term, such as a deadlock.

To model the probabilistic leakage of information, *restriction* and *hiding* operations are very important, because a system can not implicitly leak any information if and only if LU can not determine wether HU executes the *restriction* or the *hiding*. Although many studies can be used to model information leakage, they do not address a fully probabilistic system. Our PPA-Lite is a pared-down version of [32, 33], but it can be used to effectively model the information leakage of a fully probabilistic system and analyze existing quantification methods.

Before introducing the formal semantics of PPA-Lite, we give its intuitive meaning. The process $[p]a.P_1 + [1-p]b.P_2$ has two options: The first is to execute a with probability p, then proceed as P_1; The second is to execute b with probability $1-p$, then proceed as P_2. The process $a.b.c.\underline{0}\backslash\{b\}$ will not execute the action b, i.e., $a.b.c.\underline{0}\backslash\{b\} \equiv a.\underline{0}$. The process $([0.5]a.\underline{0}+[0.5]b.\underline{0})\backslash\{b\}$ will execute a with probability 1, because b is prohibited from being executed, i.e., $([0.5]a.\underline{0}+[0.5]b.\underline{0})\backslash\{b\} \equiv a.\underline{0}$. The hiding operator P/A will turn all actions in A into τ, and its probability will be preserved. For example, $([0.5]a.\underline{0}+[0.5]b.\underline{0})/\{b\} \equiv ([0.5]a.\underline{0}+[0.5]\tau.\underline{0})$. The process $rec\ x.P$ is a recursion operator. For example, $P = a.b.P$ will execute a, and then b the execution will be repeated and never stop. In the next section, an execution of a hiding (restriction) operator is called an *event*.

$$
\begin{array}{ll}
prefix & \pi.P \xrightarrow{\pi,1}_1 P \\
choice & \dfrac{P_i \xrightarrow{\pi,p}_k P_i'}{\sum_{i\in I}[q_i]P_i \xrightarrow{\pi,p.q_i}_{i,k} P_i'} \\
restriction & \dfrac{P \xrightarrow{\pi,p}_k P'}{P\backslash A \xrightarrow{\pi,p/(1-v(P,A))}_k P'\backslash A}, \quad \pi \notin A \\
hiding1 & \dfrac{P \xrightarrow{\pi,p}_k P'}{P/A \xrightarrow{\tau,p}_k P'/A}, \quad \pi \in A \\
hiding2 & \dfrac{P \xrightarrow{\pi,p}_k P'}{P/A \xrightarrow{\pi,p}_k P'/A}, \quad \pi \notin A \\
recur & \dfrac{P \xrightarrow{\pi,p}_k P'}{rec\ X.P \xrightarrow{\pi,p}_k P'[rec\ X.P/X]}
\end{array}
$$

FIGURE 2. Semantics

3.2. **Formal semantics.** The formal semantics of PPA-Lite is shown in Figure 2, where $P \xrightarrow{\pi,p}_k Q$ is a derivation meaning that in P, action π will be executed with probability p, and its result is as Q. In addition, k is used to record the derived object. For example, $([0.5]a.\underline{0}+[0.2]a.\underline{0}+[0.3]b.\underline{0}) \xrightarrow{a,0.5}_{1.1} 0$ and $([0.5]a.\underline{0}+[0.2]a.\underline{0}+[0.3]b.\underline{0}) \xrightarrow{a,0.2}_{2.1} 0$. In the rule *Erest*, the normalization factor $v(P,A)$ is used to compute the accumulative probability of executing actions in A from P, which is defined by $v(P,A) = \sum\{|p_i | P \xrightarrow{\pi,p_i}_k P_i, \pi \in A|\}$, where $\{|\ldots|\}$ is a multi-set. For example, if $P = [0.1]a.\underline{0}+[0.35]a.\underline{0}+[0.25]b.\underline{0}+[0.18]b.\underline{0}+[0.12]c.\underline{0}$, then $v(P,\{a\}) = 0.45$, $v(P,\{b\}) = 0.43$, $v(P,\{c\}) = 0.12$ and $v(P,\{a,c\}) = 0.57$.

The rule *Prefix* shows that π will be executed with probability 1 in $\pi.P$ and then $\pi.P$ is reduced to P. The rule *Choice* tells us that if P_i can be reduced to $P_i^{'}$ after π is executed with probability p, then $\sum_{i\in I}[q_i]P_i$ will become $P_i{}'$ with probability $p.q_i$. The other rules have a similar meaning.

4. **Probabilistic Information Leakage.** There are two different approaches which can be used to identify information leakage: one is based on weakly probabilistic bi-simulation, while the other is based on probabilistic trace equivalence.

Before discussing how to identify probabilistic information leakage, we present some notions. Let PRO be a set of probabilistic processes and $P \in PRO$, a sequence $P \xrightarrow{\pi_1,p_1}_{j_1} P_1 \xrightarrow{\pi_2,p_2}_{j_2} P_2 \ldots \xrightarrow{\pi_i,p_i}_{j_i} P_i$ is called an *execution* (Ex) if $P \xrightarrow{\pi_1,p_1}_{j_1} P_1$ and $P_i \xrightarrow{\pi_{i+1},p_{i+1}}_{j_{i+1}} P_{i+1}$ for all $i \geq 1$ can be derived by Figure 2. Given an $Ex = P \xrightarrow{\pi_1,p_1}_{j_1} P_1 \xrightarrow{\pi_2,p_2}_{j_2} P_2 \ldots \xrightarrow{\pi_i,p_i}_{j_i} Q$, the trace tr of Ex is a sequence of actions of Ex in the original order, i.e., $tr = \pi_1\pi_2\ldots\pi_i$, and the *execution probability* $PR_{j_1 j_2 \ldots j_i}(tr) = p_1 \times p_2 \ldots \times p_i$. In addition, $lst(Ex)$ represents the last term of Ex, i.e., $lst(Ex) = Q$. An Ex is called a

complete execution if $lst(Ex) = \underline{0}$. For simplicity, $P \xrightarrow{\pi_1,p_1}_{j_1} P_1 \xrightarrow{\pi_2,p_2}_{j_2} P_2 \ldots \xrightarrow{\pi_i,p_i}_{j_i} Q$ is written $P =\xrightarrow{tr,PR_{j_1j_2\ldots j_i}(tr)}_{j_1j_2\ldots j_i} Q$. Let P_{tr} be a set of derived processes from P with trace tr and $TR(P)$ be a set of traces from P, i.e., $P_{tr} = \left\{Q | P \xrightarrow{tr,PR_{j_1j_2\ldots j_i}(tr)}_{j_1j_2\ldots j_i} Q\right\}$ and $TR(P) = \left\{tr | P \xrightarrow{tr,PR_{j_1j_2\ldots j_i}(tr)}_{j_1j_2\ldots j_i} Q \ is \ an \ execution\right\}$.

Given $tr \in TR(P)$, let $PR_{P,Q}(tr)$ be the sum of *execution probability* of tr from P to Q and $PR_P(tr)$ be the accumulative probability from P by executing tr, that is, $PR_{P,Q}(tr) = \sum \left\{|PR_{j_1j_2\ldots j_i}(tr)| P \xrightarrow{tr,PR_{j_1j_2\ldots j_i}(tr)}_{j_1j_2\ldots j_i} Q \ is \ an \ execution|\right\}$ and $PR_P(tr) = \sum_{Q\in P_{tr}} \left\{|PR_{j_1j_2\ldots j_i}(tr)| P \xrightarrow{tr,PR_{j_1j_2\ldots j_i}(tr)}_{j_1j_2\ldots j_i} Q \ is \ an \ execution|\right\}$. If $S \subseteq PRO$, then $PR_{P,S}(tr) = \sum_{Q\in S} \left\{|PR_{j_1j_2\ldots j_i}(tr)| P \xrightarrow{t} r, PR_{j_1j_2\ldots j_i}(tr)_{j_1j_2\ldots j_i} Q \ is \ an \ execution|\right\}$, that is, $PR_{P,S}(tr)$ is the accumulative probability from P into any process in S by executing tr.

In order to further explain these notions, we give an example as follows.

Example 4.1. *Let* $P \stackrel{def}{=} [0.3]a.P_1 + [0.6]b.P_2 + [0.1]a.P_2$, $P_1 \stackrel{def}{=} [0.8]c.P_2 + [0.2]b.P_3$, $P_2 \stackrel{def}{=} [1]a.P_3$ *and* $P_3 \stackrel{def}{=} [1]c.\underline{0}$. *Then both* $P \xrightarrow{a,0.3}_1 P_1 \xrightarrow{a,0.2}_2 P_3 \xrightarrow{c,1}_1 \underline{0}$ *and* $P \xrightarrow{a,0.1}_3 P_2 \xrightarrow{a,1}_1 P_3 \xrightarrow{c,1}_1 \underline{0}$ *are executions of* P. *Let* $Ex_1 \equiv P \xrightarrow{a,0.3}_1 P_1 \xrightarrow{a,0.2}_2 P_3 \xrightarrow{c,1}_1 \underline{0}$ *and* $Ex_2 \equiv P \xrightarrow{a,0.1}_3 P_2 \xrightarrow{a,1}_1 P_3 \xrightarrow{c,1}_1 \underline{0}$. *Then the traces of* Ex_1 *and* Ex_2 *are aac. Because* $lst(Ex_1) = lst(Ex_2) = \underline{0}$, Ex_1 *and* Ex_2 *are complete. In* Ex_1 *and* Ex_2, *the* execution probabilities *of acc are respectively as follows:* $PR_{1,2,1}(aac) = 0.3 \times 0.2 \times 1 = 0.06$ *and* $PR_{3,1,1}(aac) = 0.1 \times 0.1 = 0.01$. *The derived set from* P *by executing* a *is* $\{P_1, P_3\}$, *i.e.,* $P_a = \{P_1, P_3\}$. *The set* $TR(P)$ *of traces from* P *is* $\{a, ac, aca, acac, aa, aac, b, bc, bca, bcac, bac\}$. *The accumulative probability* $PR_{P,\underline{0}}(aac)$ *from* P *to* $\underline{0}$ *by executing aac is equal to* $PR_{1,2,1}(aac) + PR_{3,1,1}(aac) = 0.07$ *and the accumulative probability* $PR_P(a)$ *from* P *by executing* a *is equal to* $0.3 + 0.1 = 0.4$.

Now we can define the probabilistic bi-simulation and probabilistic trace equivalence. Given an equivalence relation $R \subseteq PRO \times PRO$, let PRO/R be the equivalence class induced by R. Then the *weak probabilistic bi-simulation* [34] is defined as follows.

Definition 4.1. [1] *An equivalence relation* $R \subseteq PRO \times PRO$ *is called a weak probabilistic bi-simulation, if each* $(P, Q) \in R$ *implies that* $\forall \pi \in Act$, $\forall S \in PRO/R$, $PR_{P,S}(\tau^*\pi) = PR_{Q,S}(\tau^*\pi)$. *If there exists a weakly probabilistic bi-simulation* R *such that* $(P, Q) \in R$, *then* P *and* Q *are weakly probabilistic bi-simulation equivalent, written as* $P \sim Q$.

In the above definition, τ^* is a sequence of unobservable actions (including zero unobservable action).

Definition 4.2. *A process* P *can implicitly leak information via weakly probabilistic bi-simulation iff* $P\backslash\mathcal{H} \not\sim P/\mathcal{H}$.

Next, we discuss the information leakage based on probabilistic trace equivalence.

Definition 4.3. *Given the two processes* P, $Q \in PRO$, P *and* Q *are strongly probabilistically trace equivalent, denoted* $P \approx Q$, *if* $\forall x \in TR(P) \cup TR(Q)$ *and* $lst(x) \equiv \underline{0}.PR_P(x) = PR_Q(x)$.

[1]The definition is slightly different from [34], because we concentrate on fully probabilistic systems.

Definition 4.3 requires that if $P \approx Q$, then for any trace, its *execution probability* in P is equal to its execution probability of in Q. This definition is very strict, because a trace may include unobservable actions. In many situations, only observable actions need to be analyzed. This leads to another definition, that is, *weakly probabilistically trace equivalent* (WPTE).

Definition 4.4. *Given a trace tr and a set $\mathcal{A}$ of observable actions, the projection of tr on $\mathcal{A}$ is the sequence obtained by deleting the unobservable actions, formally defined by*

$$tr \uparrow \mathcal{A} = \begin{cases} \underline{0} & \text{if } tr = \underline{0} \\ t \uparrow \mathcal{A} & \text{if } tr = at \ \ and \ \ a \notin \mathcal{A} \\ a(t \uparrow \mathcal{A}) & \text{if } tr = at \ \ and \ \ a \in \mathcal{A} \end{cases}$$

Given a process P and a set $\mathcal{A}$ of observable actions, the set $ProjTR_{\mathcal{A}}(P)$ of observable traces is defined as $ProjTR_{\mathcal{A}}(P) = \{tr \uparrow \mathcal{A} | tr \in TR(P)\}$. Because unobservable actions are deleted, the execution probability of an element in $ProjTR_{\mathcal{A}}(P)$ may be different from that of the element in $TR(P)$.

Now we define the *execution probability* of an element in $ProjTR_{\mathcal{A}}(P)$. Let $x \in ProjTr_{\mathcal{A}}(P)$, then the *execution probability* of $PR_{P|\mathcal{A}}(x)$ can be defined by $PR_{P|\mathcal{A}}(x) = \sum_{tr \in TR(P)} \{|PR_P(tr) | tr \uparrow \mathcal{A} = x|\}$. For example, if the traces of the process P are $\{a.\tau.b.\underline{0}, \tau.a.b.\underline{0}, a.\tau.c.\underline{0}\}$, and their execution probabilities are 0.2, 0.5 and 0.3, respectively, then its projection on $\{a, b, c\}$ is $\{a.b.\underline{0}, a.c.\underline{0}\}$ and their execution probabilities are 0.7 and 0.3, respectively.

Definition 4.5. *Given the two processes P, $Q \in PRO$, and a set $\mathcal{A}$ of observable actions, P and Q are weakly probabilistic trace equivalent, denoted $P \cong Q$, iff $\forall x \in ProjTr_{\mathcal{A}}(P) \cup ProjTr_{\mathcal{A}}(Q)$, $lst(x) = \underline{0}.PR_{P|\mathcal{A}}(x) = PR_{Q|\mathcal{A}}(x)$.*

Definition 4.6. *A process P can implicitly leak information via weakly probabilistic trace equivalence, iff $P\backslash\mathcal{H} \ncong P/\mathcal{H}$.*

Next, we give an example of implicit information leakage via weakly probabilistic trace equivalence. The implicit information leakage via weakly probabilistic bi-simulation is similar.

Example 4.2. *Let $P = [0.3]([0.6]([0.4]l_0.\underline{0}+[0.6]l_0.l_1.\underline{0})+[0.4]l_0.l_1.l_2.\underline{0})+[0.7]([0.8]l_0.h.l_1.\underline{0}$ $+[0.2]l_0.l_1.h.l_2.\underline{0})$, and two kinds of users: LU (the set of his actions is $\{l_0, l_1, l_2\}$) and HU (his action is h).*

The hiding operator can be used to model the process that LU cannot observe HU's action h, i.e., $P/\{h\} = [0.3]([0.6([0.4]l_0.\underline{0}+[0.6]l_0.l_1.\underline{0}])+[0.4]l_0.l_1.l_2.\underline{0})+[0.7]([0.8]l_0.\tau.l_1.\underline{0}+$ $[0.2]l_0.l_1.\tau.l_2.\underline{0})$. That is, if HU executes its action h, then LU will observe the traces $l_0.\underline{0}$, $l_0.l_1.\underline{0}$ and $l_0.l_1.l_2.\underline{0}$ with probability 0.072, 0.668 and 0.26, respectively.

The restriction operator can be used to model the process that HU does not execute its action h, i.e., $P\backslash\{h\} = [0.3]([0.6]([0.4]l_0.\underline{0} + [0.6]l_0.l_1.\underline{0}) + [0.4]l_0.l_1.l_2.\underline{0}) + [0.7]([0.8]l_0.\underline{0} +$ $[0.2]l_0.l_1.\underline{0})$. That is, if HU does not execute its action h, then LU will observe the traces $l_0.\underline{0}$, $l_0.l_1.\underline{0}$ and $l_0.l_1.l_2.\underline{0}$ with probability 0.632, 0.248 and 0.26, respectively.

In this example, there exists a trace (for example, $l_0.\underline{0}$ and $l_0.l_1.\underline{0}$) such that its execution probability in $P/\{h\}$ is not equal to its probability $P\backslash\{h\}$, so that $P/\{h\} \ncong P\backslash\{h\}$. We assume that HU and LU have a prior agreement: if HU want to transmit bit 0, then he will not execute his actions h, otherwise, he will execute his actions h. In this way, LU can infer the transmitted bit by observing these traces. This means that P can implicitly leak information.

5. **Estimation of Probabilistic Information Leakage.** As mentioned in Section 1, many different methods have been proposed to measure the information leakage, but their accuracies vary. Even for a given method, its accuracy depends on its metric. In this section, we discuss these methods and their metrics.

5.1. **Method based on pure probability.** In [23], the authors propose a metric, we call it pure probability method, as follows.

Definition 5.1. [2] *Given a process P and a set PRO of its derived processes, and $R \subseteq PRO \times PRO$ is equivalence relation such that $(P\backslash\mathcal{H}, P/\mathcal{H}) \in R$, then the amount of information leakage is defined by*

$$\xi_A = \inf_R \sup_{(P',P'')\in R,\pi\in Act,S\in PRO/R} |PR_{P',S}(\tau^*\pi) - PR_{P'',S}(\tau^*\pi)|$$

where, sup and inf are the supremum and infimum of a set, respectively, and PRO/R is the equivalence class introduced by equivalence relation R.

5.2. **Methods based on relative entropy.** Let $Rest = \{tr | tr \in ProjTr_{\mathcal{A}}(P\backslash\mathcal{H})$ *and* $lst(tr) = \underline{0}\}$ and $Hiding = \{tr | tr \in \Pr ojTr_{\mathcal{A}}(P/\mathcal{H})$ *and* $lst(tr) = \underline{0}\}$, and $PR_{(P\backslash\mathcal{H})|\mathcal{A}}$ and $PR_{(P/\mathcal{H})|\mathcal{A}}$ be probability mass functions of execution probability on sets $Rest$ and $Hiding$, respectively. Because it is possible that $Rest \neq Hiding$, in order to use the relative entropy to measure the information leakage, we need extend the two probability mass functions from on $Rest$ and $Hiding$ to $Rect \cup Hiding$, as follows.

$$PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}(tr) = \begin{cases} PR_{(P\backslash\mathcal{H})|\mathcal{A}}(tr) & \text{if } tr \in Rest \\ 0 & \text{otherwise} \end{cases}$$

$$PR^E_{(P/\mathcal{H})|\mathcal{A}}(tr) = \begin{cases} PR_{(P/\mathcal{H})|\mathcal{A}}(tr) & \text{if } tr \in Hiding \\ 0 & \text{otherwise} \end{cases}$$

Given $PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}$ and $PR^E_{(P/\mathcal{H})|\mathcal{A}}$, the amount of information leakage is calculated as follows:

$$\xi_{KL} = D_{KL}\left(PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}, PR^E_{(P/\mathcal{H})|\mathcal{A}}\right)$$

5.3. **Methods based on (α-) mutual information.** In order to use mutual information to estimate information leakage, the probability of the events *hiding* and *restriction* is needed. For simplicity, let $P//\mathcal{H}$ be a set of events of process P obtained by hiding all actions in $\mathcal{H}$, let $P\backslash\backslash\mathcal{H}$ be a set of events of process P obtained by restricting actions in $\mathcal{H}$, and $\mathcal{IN} = \{P//\mathcal{H}, P\backslash\backslash\mathcal{H}\}$. the probability of the event *hiding* and *restriction* depends on the probability of the transmitted information. For example, consider a binary file where bit 0 and bit 1 account for 45% and 55% respectively. If the event *restriction* and *hiding* are used to transmit bit 0 and bit 1, then the probability of executing $P\backslash\backslash\mathcal{H}$ and $P//\mathcal{H}$ will be 0.45 and 0.55, respectively. Let $\mathcal{OUT} = \{tr | tr \in ProjTr_{\mathcal{A}}(P/\mathcal{H}) \cup ProjTr_{\mathcal{A}}(P\backslash\mathcal{H})$ *and* $lst(tr) = \underline{0}\}$, which represents a set of traces which can be observed by LU. Let IN and OUT be the random variables over $\mathcal{IN}$ and $\mathcal{OUT}$, respectively, then the information leakage based on mutual information is measured as:

$$\begin{aligned}\xi_{MI} = D_{MI}(IN;OUT) &= \sum_{in\in\mathcal{IN},tr\in\mathcal{OUT}} p(IN=in, OUT=tr)\log\frac{p(IN=in,OUT=tr)}{p(IN=in)p(OUT=tr)} \\ &= \sum_{in\in\mathcal{IN},tr\in\mathcal{OUT}} p(IN=in)p(OUT=tr|IN=in)\log\frac{p(OUT=tr|IN=in)}{\sum_{in\in\mathcal{IN}} p(OUT=tr|IN=in)p(IN=in)}\end{aligned}$$

where $p(OUT = tr|IN = P\backslash\backslash\mathcal{H}) = PR_{(P\backslash\mathcal{H})|\mathcal{A}}(tr)$ and $p(OUT = tr|IN = P//\mathcal{H}) = PR_{(P/\mathcal{H})|\mathcal{A}}(tr)$.

[2] The definition is slightly different from [34], because we only focus on fully probabilistic systems.

Similarly, the measure based on α-mutual information is as follows.

$$\xi_{\alpha MI} = D_{\alpha MI}(IN;OUT) = \frac{1}{a-1}\sum_{in\in\mathcal{IN}}\sum_{tr\in\mathcal{OUT}}\log\frac{p(IN=in,OUT=tr)^a}{(p(IN=in)p(OUT=tr))^{a-1}}$$

where $p(OUT = tr|IN = P\backslash\backslash\mathcal{H}) = PR_{P\backslash\mathcal{H}|\mathcal{A}}(tr)$ and $p(OUT = tr|IN = P//\mathcal{H}) = PR_{P/\mathcal{H}|\mathcal{A}}(tr)$.

5.4. **Methods based on simple distance, Euclidean distance and contra-cosine distance.** Similarly to Section 5.2, the probability mass functions $PR_{(P\backslash\mathcal{H})|\mathcal{A}}$ over *Rest* and $PR_{(P/\mathcal{H})|\mathcal{A}}$ over *Hiding* should be respectively expanded to $PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}$ and $PR^E_{(P/\mathcal{H})|\mathcal{A}}$ over $Rest \cup Hiding$ in order to measure the information leakage by using simple distance, Euclidean distance or Contra-cosine distance.

A). *Methods based on Simple Distance.* In this method, the difference between one of traces in *hiding* and that in *restriction* is used to estimate information leakage, as follows.

$$\xi_S = D_S\left(PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}, PR^E_{(P/\mathcal{H})|\mathcal{A}}\right) = PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}(tr) - PR^E_{(P/\mathcal{H})|\mathcal{A}}(tr)$$

where $tr \in Rest \cup Hiding$.

B). *Methods based on Euclidean Distance.* In this way, Euclidean distance is used to estimate information leakage, defined as:

$$\xi_{Eu} = D_{Eu}\left(PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}, PR^E_{(P/\mathcal{H})|\mathcal{A}}\right) = \sqrt{\sum_{tr\in Rest\cup Hiding}(PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}(tr) - PR^E_{(P/\mathcal{H})|\mathcal{A}}(tr))^2}$$

C). *Methods based on Contra-Cosine Distance.* In this way, contra-cosine distance is used to estimate information leakage, defined as:

$$\begin{aligned}\xi_{CC} &= D_{CC}(PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}, PR^E_{(P/\mathcal{H})|\mathcal{A}})\\ &= 1 - \frac{\sum_{tr\in Rect\cup Hiding} PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}(tr)PR^E_{(P/\mathcal{H})|\mathcal{A}}(tr)}{\sqrt{\sum_{tr\in Rect\cup Hiding} PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}(tr)^2}\sqrt{\sum_{tr\in Rect\cup Hiding} PR^E_{(P/\mathcal{H})|\mathcal{A}}(tr)^2}}\end{aligned}$$

6. **Comparison of Metrics for Information Leakage.** We qualitatively and quantitatively compare these measures, respectively.

6.1. **Qualitative analysis.** Now we discuss the qualitative relationship among these measures.

Proposition 6.1. $\xi_A = 0 \Rightarrow \xi_{KL} = 0$.

Proof: According to [23], $\xi_A = 0$ if and only if $P\backslash\mathcal{H}$ is weakly probabilistic bisimulation equivalent to $P/\mathcal{H}$, that is, $\xi_A = 0$ iff $P\backslash\mathcal{H} \sim P/\mathcal{H}$. Similar to the proof of [35], we can prove that if $P\backslash\mathcal{H} \sim P/\mathcal{H}$, then $P\backslash\mathcal{H} \cong P/\mathcal{H}$. If $P\backslash\mathcal{H} \cong P/\mathcal{H}$, then $\forall x \in ProjTr_{\mathcal{A}}(P\backslash\mathcal{H}) \cup ProjTr_{\mathcal{A}}(P/\mathcal{H})$ *and* $lst(x) \equiv \underline{0}.PR_{P\backslash\mathcal{H}|\mathcal{A}}(x) = PR_{P/\mathcal{H}|\mathcal{A}}(x)$. According to the definitions of $PR^E{}_{P\backslash\mathcal{H}|\mathcal{A}}$ and $PR^E{}_{P/\mathcal{H}|\mathcal{A}}$, we have that if $P\backslash\mathcal{H} \cong P/\mathcal{H}$, then $\forall x \in ProjTr_{\mathcal{A}}(P\backslash\mathcal{H}) \cup ProjTr_{\mathcal{A}}(P/\mathcal{H})$ *and* $lst(x) \equiv \underline{0}.PR^E{}_{P\backslash\mathcal{H}|\mathcal{A}}(x) = PR^E{}_{P/\mathcal{H}|\mathcal{A}}(x)$. So, if $P\backslash\mathcal{H} \cong P/\mathcal{H}$, then $\xi_{KL} = D_{KL}(PR^E{}_{(P\backslash\mathcal{H})|\mathcal{A}}, PR^E{}_{(P/\mathcal{H})|\mathcal{A}}) = 0$ (because $D_{KL}(p,q) = 0$ iff $p = q$, where p and q are probability mass function), that is, $\xi_A = 0 \Rightarrow \xi_{KL} = 0$.

Similarly, we have the following propositions.

Proposition 6.2. $\xi_{KL} = 0 \Leftrightarrow \xi_{Eu} = 0 \Leftrightarrow \xi_{CC} = 0 \Leftrightarrow \xi_{MI} = 0 \Leftrightarrow \xi_{\alpha MI} = 0$.

Proposition 6.3. $\xi_{KL} = 0 \Rightarrow \xi_S = 0$ *and* $\xi_S = 0 \not\Rightarrow \xi_{KL} = 0$.

Propositions 6.1 – 6.3 show that most of measures are consistent when being used to determine whether there exists information leakage.

6.2. **Simulation.** The qualitative analysis of these methods has been given in the Section 6.1, next we design a simulation scheme to discuss their accuacy [3]. In this scheme, we assume that HU transmits a binary file F to LU by scheduling a process P which can implicitly leak information. Let ξ be P's actual leakage amount of leakage each scheduling, N be scheduling times, and S be the size of file F. Then TER (Transmission Error Rate) is equal to $1 - \frac{\xi \times N}{S}$ if $\frac{\xi \times N}{S} \leq 1$, otherwise $TER = 0$. In the following experiment, we always make $1 \geq TER > 0$ by selecting appropriate N and S. In this case, because $TER = 1 - \frac{\xi \times N}{S}$, the leakage amount should be inversely proportional to the actual TER.

6.2.1. *Simulation scheme.* In order to simulate the TER, we need a probabilistic process P. Let $P = [p_3]([p_2]([p_1]l_0.\underline{0} + [1-p_1]l_0.l_1\underline{0}) + [1-p_2]l_0.l_1.l_2.\underline{0}) + [1-p_3]([p_4]l_0.h.l_1.\underline{0} + [1-p_4]l_0.l_1.h.l_2.\underline{0})$. HU transmits a binary file to LU by scheduling P. They agree that (1) HU schedules the process N ($N \geq 1$) times for transmitting every bit; (2) HU executes $P\backslash\{h\}$ to transmit bit 0 and $P/\{h\}$ to transmit bit 1. LU receives bit x ($x = 0$ or 1) by the following approach (Here, we take relative entropy as an example).

$$x = \begin{cases} 0 & \text{if } D_{KL}(PR^E_{(P\backslash\mathcal{H})|\mathcal{A}}, O) < D_{KL}(PR^E_{(P/\mathcal{H})|\mathcal{A}}, O) \\ 1 & \text{otherwise} \end{cases}$$

where O is the probability mass of the traces observed by LU. Because TER depends on many factors, the most important of which is distance function, we simulate the different distance functions and select the minimal TER as an actual TER. The other simulation environment is as follows. CPU: Pentium 4 3.4GHz, RAM: 512MB, Operating system: Window XP Home. A binary file Source.bin with size 232 Bytes is transmitted, of which bit 1 accounts for 49.95% and bit 0 accounts for 50.05%. In this simulation, we fix $p_1 = 0.1$, $p_2 = 0.2$, $p_3 = 0.7$ and $N = 20$. In order to have better accuracy, each experiment is independently repeated three times and its average is taken.

6.2.2. *Simulation analysis.* For the sake of clarity, two figures are used to compare the TERs. Figures 3 and 4 show the TER with the change of p_4 (Parts of the data are from our previous work [29]).

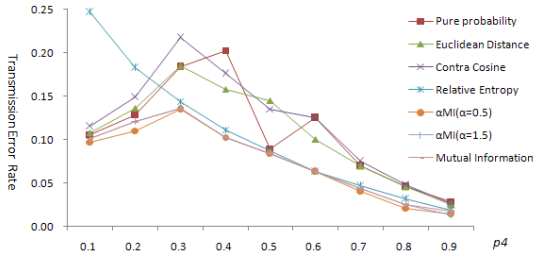

FIGURE 3. TER of different methods (except the simple method)

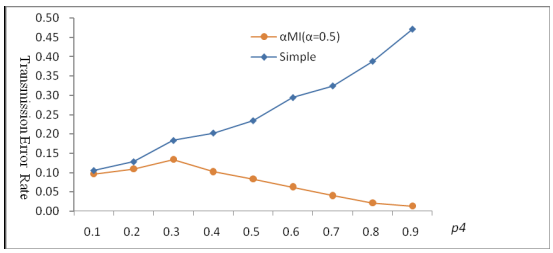

FIGURE 4. Comparison of TER based on αMI and simple distance

As shown in Figures 3 and 4, we can know that: (1) TER based on Euclidean (Contra-Cosine, simple distance, relative entropy and Aldini) is high, (2) TER based on the (α-) MI is the lowest. Thus, we have shown that the (α-) MI as a distance function is more

[3]In fact, this scheme is indirect. It is very difficult to design an experiment to directly demonstrate which metrics are more accurate, because a direct demonstration is closely correlated to optimal coding algorithms, however, none of the known algorithms come close to achieving this goal [30].

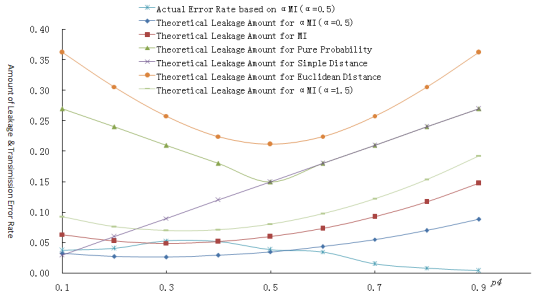

FIGURE 5. Comparison between the least TER and theoretical leakage amount of different methods (except the relative entropy and contra-cosine)

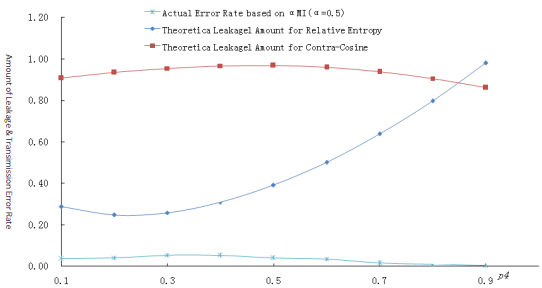

FIGURE 6. Comparison between the least TER and theoretical leakage amount based on the relative entropy and contra-cosine

accurate than the others. We therefore select the TER based on αMI ($\alpha = 0.5$) as an actual TER.

Figures 3 and 4 show the relationship between the actual TER and the computed leakage amount. As shown before, when $TER = 1 - \frac{\xi \times N}{S}$, the leakage amount should be inversely proportional to the actual TER. From Figures 3 and 4, we know that the methods based on (α-) MI accurately reflect this trend, so the methods based on (α-) MI are more accurate.

As shown in Section 5.3, however, a necessary condition to adopt the (α-) mutual information is to obtain the probability mass of the transmitted information. In many situations, this condition cannot easily be satisfied. In these cases, the method based on the relative entropy should be adopted, because this method closely reflects this trend (i.e., for the measure based on relative entropy, the computed leakage amount is almost inversely proportional to the actual TER).

Now we give some intuitive explanations for the accuracy of these methods. For the simple distance, only a single trace (l_0 of P in simulation) rather than all traces (for example, $l_0.l_1$ and $l_0.l_1.l_2$ of P) is considered. In fact, the other traces (for example, $l_0.l_1$ and $l_0.l_1.l_2$) may cause information leakage, so the method based on the simple distance is less accurate. The (α-) mutual information is used to measure the amount of information that one random variable contains another, which is the amount of shared information by the two variables, so the methods based on (α-) mutual information are more accurate.

7. **Conclusions and Future Work.** Quantifying the implicit information leakage is of great importance, especially for an FPS. Although many quantitative methods have been proposed, there has been little work to analyze their consistency and accuracy. Compared with the related work, the major novelty of our work is that: (1) some existing measure methods, which can be used to estimate the implicit information leakage, are uniformly and formally modeled using PPA-Lite; (2) we analyze the consistency and accuracy of these methods. The results show that (1) most of measures are consistent when being used to determine whether there exists information leakage, (2) methods based on (α-) mutual information are the most accurate if the probability of information has been known; otherwise, the methods based on relative entropy becomes more accurate.

With respect to the future work, our analysis only focused on a fully probabilistic system, while a timed probabilistic system may implicitly leak information. Therefore,

we will try to quantify the information leakage of a timed probabilistic system. In addition, because, as shown in Figures 3 and 4, the TER based on α-Mutual information ($\alpha = 0.5$) is less than that based on the others, it is worthy of further study that TER reaches a minimum when α varies. Moreover, we do not study the attacking ability. In fact, Jonsson [36] shows that different adversaries may have different attacking abilities, and even for a given adversary, its attacking ability changes with the phases of the attacking process. Therefore quantifying the attacking ability for a probabilistic/timed probabilistic system is planned in our future work.

Acknowledgments. This work is partially supported by National Basic Research Program of China (973 Program) (2007CB311100), the National High Technology Research and Development Program of China (863 Program) (2000AA01Z438), the National Natural Science Foundation of China (61070186, 61063002 and 60903079).

REFERENCES

[1] S.-H. Chen, P.-C. Chang, Q. Zhang and C.-B. Wang, A guided memetic algorithm with probabilistic models, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(B), pp.4753-4764, 2009.

[2] J. Park, W. Liang, J. Choi, A. A. El-Keib and J. Watada, Probabilistic production cost credit evaluation of wind turbine generators, *International Journal of Innovative Computing, Information and Control*, vol.5, no.11(A), pp.3637-3646, 2009.

[3] J. Park, S. Jeong, J. Choi, J. Cha and A. El-Keib, A probabilistic reliability evaluation of Korea power system, *ICIC Express Letters*, vol.2, no.2, pp.137-142, 2008.

[4] Y. Jiang and S. Wang, Measurement and quantitative analysis of human visual interpolation ability for partially erased objects, *ICIC Express Letters*, vol.2, no.1, pp.7-13, 2008.

[5] B. Fang, Y. Guo and Y. Zhou, Information content security on the Internet: The control model and its evaluation, *Science in China Series F: Information Sciences*, vol.53, no.1, pp.30-49, 2010.

[6] J. Rutkowska, The implementation of passive covert channels in the Linux kernel, *Chaos Communication Congress*, pp.1-9, 2004.

[7] S. Qin, Covert channel analysis in secure operating systems with high security level, *Journal of Software*, vol.15, no.12, pp.1837-1849, 2004.

[8] S. H. Son, R. Mukkamala and R. David, Integrating security and real-time requirements using covert channel capacity, *IEEE Transactions on Knowledge and Data Engineering*, vol.12, no.6, pp.865-879, 2000.

[9] K. G. Lee, J. H. Choi, K. S. Lim and S. Lee, Novel methodologies to detect covert databases, *International Journal of Innovative Computing, Information and Control*, vol.6, no.3(B), pp.1313-1324, 2010.

[10] S. Cabuk, C. E. Brodley and C. Shields, IP covert timing channels: Design and detection, *Proc. of the 11th ACM Conference on Computer and Communications Security*, pp.178-187, 2004.

[11] S. Cabuk, C. E. Brodley and C. Shields, IP covert channel detection, *ACM Transactions on Information and System Security*, vol.12, no.4, pp.1-29, 2009.

[12] S. Zander, G. Armitage and F. Branch, A survey of covert channels and countermeasures in computer network protocols, *IEEE Communications Surveys and Tutorials*, vol.9, no.3, pp.44-57, 2007.

[13] G. Shah, A. Molina and M. Blaze, Keyboards and covert channels, *Proc. of the 15th USENIX Security Symposium*, pp.59-75, 2006.

[14] D. E. R. Denning, *Cryptography and Data Security*, Addison-Wesley Longman Publishing Co., Inc., Boston, 1982.

[15] D. Clark, S. Hunt and P. Malacaria, A static analysis for quantifying information flow in a simple imperative language, *Journal of Computer Security*, vol.15, no.3, pp.321-371, 2007.

[16] J. K. Millen, Covert channel capacity, *Proc. of IEEE Symposium on Security and Privacy*, pp.60-66, 1987.

[17] J. W. Gray III, Toward a mathematical foundation for information flow security, *Journal of Computer Security*, vol.1, no.3, pp.255-294, 1992.

[18] S. McCamant and M. D. Ernst, Quantitative information flow as network flow capacity, *ACM SIGPLAN Notices*, vol.43, no.6, pp.193-205, 2008.

[19] J. Newsome, S. McCamant and D. Song, Measuring channel capacity to distinguish undue influence, *Proc. of the ACM SIGPLAN on Programming Languages and Analysis for Security*, pp.73-85, 2009.
[20] M. Boreale, Quantifying information leakage in process calculi, *Automata, Languages and Programming*, vol.4052, pp.119-131, 2006.
[21] M. R. Clarkson, A. C. Myers and F. B. Schneider, Belief in information flow, *Proc. of the 18th IEEE Workshop Computer Security Foundations*, pp.31-45, 2005.
[22] A. D. Pierro, C. Hankinb and H. Wiklickyb, Approximate non-interference, *Journal of Computer Security*, vol.12, no.1, pp.37-81, 2004.
[23] A. Aldini and A. D. Pierro, A quantitative approach to noninterference for probabilistic systems, *Electronic Notes in Theoretical Computer Science*, vol.99, pp.183-203, 2004.
[24] A. Aldini and A. D. Pierro, Estimating the maximum information leakage, *International Journal of Information Security*, vol.7, no.3, pp.219-242, 2008.
[25] G. Lowe, Quantifying information flow, *Proc. of IEEE Computer Security Foundations Workshop*, pp.18-31, 2002.
[26] P. Malacaria and H. Chen, Lagrange multipliers and maximum information leakage in different observational models, *Proc. of the 3rd ACM SIGPLAN Workshop on Programming Languages and Analysis for Security*, pp.135-146, 2008.
[27] G. Smith, On the foundations of quantitative information flow, *Lecture Notes in Computer Science*, vol.5504, pp.288-302, 2009.
[28] I. S. Moskowitz and M. H. Kang, Covert channels-here to stay? Computer assurance, *Proc. of the 9th Annual Conference on COMPASS*, pp.235-243, 1994.
[29] Y. Guo, L. Yin, Y. Zhou and B. Fang, Quantifying information leakage for fully probabilistic systems, *Proc. of the 10th IEEE International Conference on Computer and Information Technology*, pp.589-595, 2010.
[30] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley-Interscience, 2006.
[31] E. Oubel, H. Neemuchwala, A. Hero, L. Boisrobert, M. Laclaustra and A. Frangi, Assessment of artery dilation by using image registration based on spatial features, *Proc. of SPIE Medical Imaging*, pp.1283-1291, 2005.
[32] N. López and M. Nunez, An overview of probabilistic process algebras and their equivalences, *Validation of Stochastic Systems*, pp.89-123, 2004.
[33] A. Giacalone, C. C. Jou and S. A. Smolka, Algebraic reasoning for probabilistic concurrent systems, *Proc. of IFIP TC2 Working Conference on Programming Concepts and Methods*, pp.1-14, 1990.
[34] A. Aldini, M. Bravetti and R. Gorrieri, A process-algebraic approach for the analysis of probabilistic noninterference, *Journal of Computer Security*, vol.12, no.2, pp.191-245, 2004.
[35] I. Christoff, Testing equivalences and fully abstract models for probabilistic processes, *CONCUR'90 Theories of Concurrency: Unification and Extension*, pp.26-138, 1990.
[36] E. Jonsson and T. Olovsson, A quantitative model of the security intrusion process based on attacker behavior, *IEEE Transactions on Software Engineering*, vol.23, no.4, pp.235-245, 1997.