Quantifying information leakage in process calculi*

Michele Boreale
Dipartimento di Sistemi e Informatica
Università di Firenze

Abstract. We study two quantitative models of information leakage in the pi-calculus. The first model presupposes an attacker with an essentially unlimited computational power. The resulting notion of *absolute leakage*, measured in bits, is in agreement with secrecy as defined by Abadi and Gordon: a process has an absolute leakage of zero precisely when it satisfies secrecy. The second model assumes a restricted observation scenario, inspired by the testing equivalence framework, where the attacker can only conduct repeated success-or-failure experiments on processes. Moreover, each experiment has a cost in terms of communication actions. The resulting notion of leakage *rate*, measured in bits per action, is in agreement with the first model: the maximum information that can be extracted by repeated experiments coincides with the absolute leakage A of the process. Moreover, the overall extraction cost is at least A/R, where R is the rate of the process. Strategies to effectively estimate both absolute leakage and rate are also discussed.

Keywords: process calculi, secrecy, information leakage, information theory.

1 Introduction

Research in language-based security has traditionally focused on qualitative aspects. Recently, a few models have been proposed that allow forms of quantitative reasoning on security properties. For a sequential program, it is natural to quantify leakage by measuring the information flow between secret ("high") and public ("low") variables induced by the computed function. Along these lines, an elegant theory of quantitative non-interference has been recently proposed by Clark et al. [12] (other proposals in the literature are examined in the concluding section.)

In this paper, we study quantitative models of information leakage in process calculi. Processes come with no natural notion of computed function. Rather, one is interested in quantifying the leakage induced by their *observable behaviour*. The difference in intent can be illustrated by the following concrete example. A smart-card implements a function that takes documents as input and releases documents signed with a secret key as output. However, typical attacks targeting the secret key do not focus on the function itself, but rather on the behaviour of the card, in terms e.g. of observed time variance of basic operations [9], or observed power consumption [10].

Our starting point is the notion of *secrecy* as formalized by Abadi and Gordon, originally in the setting of the spi-calculus [1]. In the sequel, we will refer to this notion as AG-secrecy. Informally, AG-secrecy holds for a process P and a parameter x representing a sensible information, if the the observable behaviour of P does not depend on the actual values x takes on. In other words, an attacker cannot infer anything about x

* Author's address: Dipartimento di Sistemi e Informatica, Viale Morgagni 65, I–50134 Firenze, Italy. Email: boreale@dsi.unifi.it. Work partially supported by the EU within the FET-GC2 initiative, project SENSORIA.

by interacting with P. The notion of "observable behaviour" is formalized in terms of behavioural equivalence, such as may testing [4, 2].

Although elegant and intuitive, AG-secrecy is in practice too rigid. The behaviour of a typical security application depends nontrivially on the sensible information it protects. Nevertheless, many such applications are considered secure, on the ground that the *amount* of leaked information is, on the average, negligible. Consider a PIN-checking process $P(x)$ that receives a code from a user and checks it against a 5-digits secret code x, in order to authorize or deny a certain operation. Clearly, an attacker may acquire negative information about x by interacting with $P(x)$. However, if $P(x)$ is intended to model, say, an off-line device like a card reader, such small leaks should be of no concern. More generally, one would like to first measure the information leakage of a given system and then decide if it is acceptable or not.

In the present paper, we propose two quantitative models of leakage for processes: one for measuring *absolute* leakage, and one for measuring the *rate* at which information is leaked. As explained below, the two models correspond to different assumptions on the control an attacker may exercise over processes. The connections between these two models will also be clarified.

After quickly reviewing a few notions from information theory that will be used in the paper (Section 2), we introduce our reference language, a pi-calculus with data values (Section 3). In the first model (absolute leakage, Section 4), we presuppose an attacker with full control over the process. Using the language of unconditional security, the model can be phrased as follows. A sensible information is modeled as a random variable, say X. The a priori *uncertainty* of an adversary about X is measured by the Shannon *entropy* $H(X)$, expressed in bits. For full generality, it is assumed that some "side-information" Y, possibly related to X, is publicly available: the conditional entropy $H(X|Y)$ measures the uncertainty about X given that Y is known. The process P, depending in general on both X and Y, induces a random variable $Z = P(X,Y)$: following the discussion above, it is reasonable to stipulate that Z takes as values "observable behaviours", that is, equivalence classes of a fixed behavioral semantics. Now, the conditional entropy $H(X|Y,Z)$ quantifies the uncertainty on X left after observing both Y and Z. Hence the difference $I = H(X|Y) - H(X|Y,Z)$ is the amount of uncertainty about X removed by P, that we take as its absolute leakage. We prove that this notion is in full agreement with the qualitative notion of AG-secrecy. In the special case when there is no side-information, this means that $P(x)$ respects AG-secrecy if and only if $P(X)$ has an absolute leakage of 0 for every random variable X. We also offer two alternative characterizations of zero-leakage, hopefully more amenable to automatic checking.

The second model we consider (rate of leakage, Section 5), refines the previous scenario by introducing a notion of *cost*. Adapting the testing equivalence framework [4], we stipulate that an attacker can only conduct upon P repeated experiments E_1, E_2,... each yielding a binary answer, success or failure. The attacker has "full control" – in the sense of the first model – over the compound systems $P||E$, but not over P itself. The security measure we are interested in is the overall number of *communications* required to extract one bit of information in this scenario. Thus, we define the rate at which P leaks information in terms of the maximal number of bits of information per visible action conveyed by an experiment on P. We then give evidence that this is indeed

a reasonable notion. First, we establish a relationship with the first model, showing that absolute leakage A coincides with the maximum information that can be extracted by repeated experiments, and that this costs at least A/R, where R is the rate of P. Second, we establish that, under certain conditions, process iteration ($*P$) leaves the rate of P unchanged, which is what one would expect from a good definition of rate. Finally, in the vein of testing equivalence, we give an experiment-independent characterization of rate in terms of execution traces.

Strategies to effectively estimate rate of leakage (Section 5) and absolute leakage (Section 6) are also discussed. These strategies depend on the use of symbolic semantics in the vein of [7, 3]. Some remarks on further and related work conclude the paper (Section 7). Proofs have been omitted due to lack of space.

2 Preliminary notions

We quickly recall a few concepts from elementary information theory; see e.g. [15] for full definitions and underlying motivations. We shall consider discrete random variables (r.v.) $X, Y, \ldots$ defined over a common probabilty space Ω. We say that a r.v. X is of *type* U, and write $X : U$, if $X(\Omega) \subseteq U$. We shall always assume U to be finite. Elements $u \in U$ are called *samples* of X, and $|X|$ is $|\{u \in U | \Pr[X = u] > 0\}|$. The concepts of independent and uniformly distributed (u.d.) random variables, and of expectation of X ($E[X]$, for X real-valued) are defined as usual. As a function, every random variable induces a partition into events of its domain Ω, $\{X^{-1}(u) \,|\, u \in X(\Omega)\}$: we say that two random variables X and Y are *equivalent* if they induce on Ω the same partition. A vector of random variables $\tilde{X} = (X_1, \ldots, X_n)$, with $n \geq 0$ and $X_i : U_i$, is just a random variable of type $U_1 \times \cdots \times U_n$.

Given $X : U$, the *entropy* X of and *conditional entropy of* X *given* $Y : V$ are defined by:

$$H(X) \stackrel{\text{def}}{=} -\sum_{u \in U} \Pr[X = u] \cdot \log(\Pr[X = u])$$
$$H(X|Y) \stackrel{\text{def}}{=} \sum_{v \in V} H(X|Y = v) \cdot \Pr[Y = v]$$

where $H(X|Y = v) = -\sum_{u \in U} \Pr[X = u|Y = v] \cdot \log(\Pr[X = u|Y = v])$, all logarithms are taken to the base of 2 and by convention $0 \cdot \log 0 = 0$. Two equivalent random variables have the same entropy and conditional entropy. The following (in)equalities hold:

$$0 \leq H(X) \leq \log |X| \tag{1}$$
$$H(X,Y) = H(X|Y) + H(Y) \qquad \text{(chain rule)} \tag{2}$$
$$H(X_1, \ldots, X_n) \leq H(X_1) + \cdots + H(X_n) \tag{3}$$

where: in (1), equality on the left holds iff X is a constant, and equality on the right holds iff X is u.d.; in (3), equality holds iff the X_i's are pairwise independent. Note that by (2) and (3), $H(X|Y) = H(X)$ iff X and Y are independent. If $Y = F(X)$ for some function F then $H(Y|X) = 0$. *Information on* X *conveyed by* Y (aka, *mutual information*) is defined as:

$$I(X;Y) \stackrel{\text{def}}{=} H(X) - H(X|Y).$$

By the chain rule, $I(X;Y) = I(Y;X)$, and $I(X;Y) = 0$ iff X and Y are independent. Mutual information can be generalized by conditioning on another r.v. Z: $I(X;Y|Z) \stackrel{\text{def}}{=} H(X|Z) - H(X|Z,Y)$. Conditioning on Z may in general either increase or decrease mutual information between X and Y. Note that entropy of a r.v. only depends on the underlying probability distribution; thus any probability vector $\tilde{p} = (p_1, \ldots, p_n)$ ($p_i \geq 0$, $\sum_i p_i = 1$) determines a unique entropy value denoted $H(\tilde{p})$; we shall often abbreviate $H(p, 1-p)$ as $H(p)$.

3 The model

We assume a countable set of *variables* $\mathcal{V} = \{x, y, ...\}$, a family of non-empty, finite *value-sets* $\mathfrak{U} \stackrel{\text{def}}{=} \{U, V, ...\}$, and a countable set of *names* $\mathcal{N} = \{a, b, ...\}$, partitioned into a family of *sorts* $\mathcal{S}, \mathcal{S}', ...$ We assume a function that maps each x to some $T \in \mathfrak{U} \cup \{\mathcal{S}, \mathcal{S}', ...\}$, written $x : T$, and say that x has *type* T. The inverse image of each T is infinite. These notations are extended to tuples as expected, e.g. for $\tilde{x} = (x_1, ..., x_n)$ and $\tilde{T} = (T_1, ..., T_n)$, $\tilde{x} : \tilde{T}$ means $x_1 : T_1, ..., x_n : T_n$. We let u, v be generic elements of a finite value-set. By slight abuse of notation, we sometimes denote by $\tilde{U}$ the cartesian product $U_1 \times \cdots \times U_n$.

An *evaluation* σ is a map from $\mathcal{V}$ to $\bigcup_{U \in \mathfrak{U}} U \cup \mathcal{N}$ that respects typing, that is, for each $x \in \text{dom}(\sigma)$, $x : T$ implies $\sigma(x) \in T$. We denote by $[\tilde{d}/\tilde{x}]$ the evaluation mapping $\tilde{x}$ to $\tilde{d}$ component-wise. By $t\sigma$, where t is a term over an arbitrary signature with free variables $\text{fv}(t) \subseteq \mathcal{V}$, we denote the result of replacing each free variable $x \in \text{dom}(\sigma) \cap \text{fv}(t)$ with $\sigma(x)$.

We assume a language of logical *formulae* $\phi, \psi, ...$ We leave the language unspecified, but assume it includes a first order calculus with variables $\mathcal{V}$, that function symbols include all values in $\mathfrak{U}$ and names as constants, and that the set of predicates includes equality $[x = y]$. We write $\mathfrak{U}, \mathcal{N} \models \phi$, or simply $\models \phi$, if for all evaluations σ s.t. $\text{dom}(\sigma) \supseteq \text{fv}(\phi)$, $\phi\sigma$ is valid (i.e. a tautology). We will often write $\phi(\tilde{x})$ to indicate that the free variables of ϕ are included in $\tilde{x}$, and in this case, abbreviate $\phi[\tilde{u}/\tilde{x}]$ as $\phi(\tilde{u})$.

The process language is a standard pi-calculus with variables and data values. We assume a countable set of *identifiers* $A, B, ...$ and use $e, e' ...$ to range over an unspecified set of *expressions*, that can be formed starting from variables, values and names. The syntax of processes $P, Q, ...$ is given below.

$$m ::= x \mid a$$

$$P, Q ::= \mathbf{0} \mid \tau.P \mid m(\tilde{x}).P \mid \overline{m}\tilde{e}.P \mid \phi P \mid P + P \mid (\nu b)P \mid P|P \mid A(\tilde{e})\ .$$

Each identifier A has an associated defining equation of the form $A(\tilde{x}) \stackrel{\text{def}}{=} P$. Input prefix $m(\tilde{x}).$ and restriction (νb) are binders for $\tilde{x}$ and b, respectively, thus, notions of free variables (fv) and free names (fn) arise as expected. We identify processes up to alpha-equivalence. We assume a few constraints on the syntax above: $\tilde{x}$ is a tuple of distinct elements in input prefix and in $A(\tilde{x}) \stackrel{\text{def}}{=} P$, and in the latter $\text{fv}(P) \subseteq \tilde{x}$; ϕ is quantifier-free. We assume a fixed sorting system *à la* Milner. In particular, each sort $\mathcal{S}$ has an associated *sort object* $ob(\mathcal{S}) = (T_1, ..., T_k)$ ($k \geq 0$). Here, each T_i is either a sort or a value-set from the universe $\mathfrak{U}$. Informally, a process obeys this sorting system if in every input and output prefix, a name/variable m of sort $\mathcal{S}$ carries a tuple of objects of the sort specified by $ob(\mathcal{S})$; we omit the details that are standard. We let Π^{o} the set of processes (possibly containing free variables) obeying these conditions and Π^{c} the subset of *closed* processes. Notationally, we shall often omit trailing $\mathbf{0}$'s, writing e.g. $a.b.$ instead of $a.b.\mathbf{0}$, we shall write $\sum_{i=1}^{n} P_i$ for nondeterministic choice $P_1 + \cdots + P_n$, and let replication $!P$ denote the process defined by the equation: $!P \stackrel{\text{def}}{=} P|!P$.

We assume over Π^{c} the standard *early* operational semantics of pi-calculus – see e.g. [14]. Let us just remind that in this form of semantics transitions are the form $P \xrightarrow{\mu} P'$, where μ is one of τ (invisible action), $a\tilde{d}$ (input action) or $(\nu\tilde{c})\overline{a}\tilde{d}$ with $\tilde{c} \subseteq \tilde{d}$

(output action) and $d ::= a \mid u$ (name or value). A few standard notations will also be used. In particular, for each *visible* (different from τ) action α, $P \stackrel{\alpha}{\Longrightarrow} P'$ means $P(\stackrel{\tau}{\longrightarrow})^* \stackrel{\alpha}{\longrightarrow} (\stackrel{\tau}{\longrightarrow})^* P'$. This notation is extended to any sequence of visible actions s (i.e. a *trace*), $P \stackrel{s}{\Longrightarrow} P'$, as expected. Finally, $P \stackrel{s}{\Longrightarrow}$ means that there is P' s.t. $P \stackrel{s}{\Longrightarrow} P'$.

We let $\sim$ be a fixed equivalence relation over Π^c. We denote by $[Q]$ the equivalence class of a process Q. We assume $\sim$ is included in *trace equivalence* [2], includes *strong bisimulation* [14] and preserves all operators of the calculus, except possibly input prefix and unguarded nondeterministic choice. We introduce now the main concept of this section. An *open process* is a pair $(P,\tilde{x})$, written $P(\tilde{x})$, with $\tilde{x}$ a tuple of distinct variables of type $\tilde{U} \subseteq \mathfrak{U}$ and $P \in \Pi^o$ such that $\mathrm{fv}(P) \subseteq \tilde{x}$; when no confusion arises, we shall abbreviate $P[\tilde{u}/\tilde{x}]$ as $P(\tilde{u})$ and $(P[\tilde{y}/\tilde{x}])(\tilde{y})$ as $P(\tilde{y})$ ($\tilde{y}$ a tuple of distinct variables.)

Definition 1 (open processes as random variables). *Let $P(\tilde{x})$ be an open process and $\tilde{X}$ a vector of random variables, with $\tilde{x} : \tilde{U}$ and $\tilde{X} : \tilde{U}$, for one and the same $\tilde{U}$. We denote by $P(\tilde{X})$ the random variable $F \circ \tilde{X}$, where $F = \lambda \tilde{u} \in \tilde{U}.[P[\tilde{u}/\tilde{x}]]$.*

Note that a sample of $P(\tilde{X})$ is an equivalence class of $\sim$.

Example 1. A PIN*-checking process can be defined as follows. Here, $x,z : 1..k$ for some integer k and x represents the secret code. The situation is modeled where an observer can freely interact with the checking process.*

$$Check(x) \stackrel{\text{def}}{=} a(z).([z = x]\overline{ok}.Check(x) + [z \neq x]\overline{no}.Check(x)). \tag{4}$$

The range of the function $F : u \mapsto [Check(u)]$ has k distinct elements, as $u \neq u'$ implies $Check(u) \not\sim Check(u')$. As a consequence, if $X : 1..k$ is a random variable, the distribution of $P(X)$ mirrors exactly that of X. E.g., if X is uniformly distributed, then so is $P(X)$, i.e. the probability of each sample is $1/k$.

Note that, if $P(\tilde{u}) \sim Q(\tilde{u})$ for each $\tilde{u}$, then, for any $\tilde{X}$, $P(\tilde{X})$ and $Q(\tilde{X})$ are the same random variable. Another concept we shall rely upon is that of *most general boolean*, borrowed from [7, 3], that is, the most general condition under which two given open processes are equivalent.

Definition 2 (mgb**).** *Let $P(\tilde{x})$ and $Q(\tilde{y})$ be two open processes, with $\tilde{x} : \tilde{U}$ and $\tilde{y} : \tilde{V}$. We denote by* $\mathrm{mgb}(P(\tilde{x}), Q(\tilde{y}))$ *a chosen formula $\phi(\tilde{x},\tilde{y})$ s.t. for each $\tilde{u} \in \tilde{U}$ and $\tilde{v} \in \tilde{V}$: $P(\tilde{u}) \sim Q(\tilde{v})$ if and only if $\phi(\tilde{u},\tilde{v})$ is true.*

It is worthwhile to notice that in many cases mgb's for pairs of open pi-processes can be automatically computed relying on *symbolic* transition semantics. Let us recall from [7, 3] that a symbolic transition also carries a logical formula: $P \stackrel{\mu,\phi}{\longrightarrow} P'$. In [7], an algorithm is described to compute mgb's for pair of processes both having *finite* symbolic transition systems. Here, we will just assume that the logical language guarantees existence of mgb for any given pair of open processes.

4 Absolute leakage

Throughout the section and unless otherwise stated, we let $P(\tilde{x},\tilde{y})$ be an arbitrary open process, with $\tilde{x} : \tilde{U}$ and $\tilde{y} : \tilde{V}$, while $\tilde{X} : \tilde{U}$ and $\tilde{Y} : \tilde{V}$ are two arbitrary vectors of random variables, and $Z \stackrel{\text{def}}{=} P(\tilde{X},\tilde{Y})$.

Definition 3 (absolute leakage). *The (absolute)* information leakage from $\tilde{X}$ to P given $\tilde{Y}$ *is* $\mathcal{A}(P;\tilde{X} \mid \tilde{Y}) \stackrel{\text{def}}{=} I(\tilde{X};Z|\tilde{Y}) = H(\tilde{X}|\tilde{Y}) - H(\tilde{X}|\tilde{Y},Z)$.

When $\tilde{Y}$ is empty, we simply write leakage as $\mathcal{A}(P;\tilde{X})$. A first useful fact says that leakage is nothing but the uncertainty about Z after observing $\tilde{Y}$. The proof is a simple application of the chain rule (2).

Lemma 1. $\mathcal{A}(P;\tilde{X} \mid \tilde{Y}) = H(Z|\tilde{Y})$. *In particular, if* $\tilde{y}$ *is empty,* $\mathcal{A}(P;\tilde{X}) = H(Z)$.

Example 2. The process $Check(x)$ *defined in (4) leaks* all *information about x. For example, if X is u.d on* $1..k$ *then* $Z = P(X)$ *is u.d. over a set of k samples. Hence* $\mathcal{A}(Check;X) = H(Z) = \log k = H(X)$.

Suppose now the adversary cannot interact freely with Check, but rather he observes the result of a user interacting with Check:

$$OneTry(x,y) \stackrel{\text{def}}{=} (\nu a)(Check(x)|\overline{a}y)\,. \tag{5}$$

Clearly, for any $X,Y : 1..k$, *the range of the random variable* $Z = OneTry(X,Y)$ *has only two elements, that is* $[\tau.\overline{ok}]$ *and* $[\tau.\overline{no}]$, *that have probabilities* $\Pr[X = Y]$ *and* $\Pr[X \neq Y]$, *respectively. In the case where X and Y are uniformly distributed and independent, these probabilities are* $1/k$ *and* $1 - 1/k$, *respectively. We are interested in* $\mathcal{A}(OneTry;X \mid Y)$. *Easy calculations show that Z and Y are in fact independent. For the sake of concreteness, let us assume* $k = 10$*; then we can compute absolute leakage as*

$$\mathcal{A}(OneTry;X \mid Y) = H(Z|Y) = H(Z) = H(\frac{1}{10}) \approx 0.469\,.$$

In this case, knowledge of Y brings no advantage to the adversary.

The next result is about composing leakage. Let us say that a n-holes context $C[\cdot,...,\cdot]$ preserves $\sim$ if whenever $P_i \sim P'_i$ for $1 \leq i \leq n$ then $C[P_1,...,P_n] \sim C[P'_1,...,P'_n]$. The following proposition states that leakage of a compound system cannot be greater than the sum of leakage of individual systems. The (simple) proof is based on inequality (3) plus the so called "data processing" inequality, saying that for any r.v. W and any function F of appropriate type, $H(F(W)) \leq H(W)$.

Proposition 1 (compositionality). *Let* $C[\cdot,...,\cdot]$ *be a n-holes context that preserves* $\sim$, *and let* $Q_i(\tilde{x},\tilde{y})$ *be open processes,* $1 \leq i \leq n$. *Let* $P(\tilde{x},\tilde{y}) = C[Q_1(\tilde{x},\tilde{y}),...,Q_n(\tilde{x},\tilde{y})]$. *Then*

$$\mathcal{A}(P;\tilde{X} \mid \tilde{Y}) \leq \sum_{i=1}^{n} \mathcal{A}(Q_i;\tilde{X} \mid \tilde{Y})\,. \tag{6}$$

For example, in the case of parallel composition, inequality (6) specializes to $\mathcal{A}(P|Q;\tilde{X} \mid \tilde{Y}) \leq \mathcal{A}(P;\tilde{X} \mid \tilde{Y}) + \mathcal{A}(Q;\tilde{X} \mid \tilde{Y})$. The inequality implies that leakage is never increased by unary operators. In the case of replication !, this leads to the somewhat unexpected conclusion $\mathcal{A}(!P;\tilde{X} \mid \tilde{Y}) \leq \mathcal{A}(P;\tilde{X} \mid \tilde{Y})$. Inequalities provided by (6) may hold strict or not, as shown below.

Example 3. Consider $P(x) = ([x = 0]a)|a$, *where* $x : \{0,1\}$, *and X u.d. on the same set. Then* $1 = \mathcal{A}(P;X) > \mathcal{A}(!P;X) = 0$. *The reason for the latter equality is that for* $v \in \{0,1\}$, $!P(v) \sim !a$, *that is, the behaviour of* $!P(x)$ *does not depend on x, so* $H(P(X)) = 0$.

On the other hand, consider $P_1(x) = [x = 2]a + [x = 4]a$ *and* $P_2(x) = [x = 1]b + [x = 2]b$, *where this time* $x : 1..4$, *and X is u.d. on the same set. Then* $\mathcal{A}(P_1|P_2;X) = \mathcal{A}(P_1;X) + \mathcal{A}(P_2;X) = H(\frac{1}{2}) + H(\frac{1}{2}) = 2$.

Our next task is to investigate the situation of zero leakage. We start from Abadi and Gordon' definition of Secrecy, originally formulated in the setting of the spi-calculus [1]. According to the latter, a process $P(\tilde{x})$ keeps $\tilde{x}$ secret if the observable behaviour of $P(\tilde{x})$ does not depend on the actual values $\tilde{x}$ may take on. Partly motivated by the non-interference scenario [5, 16], where variables are partitioned into "low" and "high", we find it natural to generalize the definition of [1] to the case where the behaviour of P may also depend on further parameters $\tilde{y}$ known to the adversary.

Definition 4 (generalized secrecy). *We say that $P(\tilde{x},\tilde{y})$* keeps $\tilde{x}$ secret given $\tilde{y}$ *if, for each $\tilde{v} \in \tilde{V}$, and for each $\tilde{u} \in \tilde{U}$ and $\tilde{u}' \in \tilde{U}$, it holds $P(\tilde{u},\tilde{v}) \sim P(\tilde{u}',\tilde{v})$.*

The main result of the section states agreement of diverse notions of secrecy: functional (described above), quantitative (zero leakage) and logical (independence of mgb's from $\tilde{x}$). The latter appears to be more amenable to automatic checking, at least in those cases where the mgb can be computed. We also offer an "optimized" version of the quantitative notion, by which it is sufficient to check zero-leakage relatively to uniformly distributed and independent $\tilde{X}$ and $\tilde{Y}$.

Theorem 1 (secrecy). *Let $P(\tilde{x},\tilde{y})$ be an open process. The following assertions are equivalent:*

1. *$P(\tilde{x},\tilde{y})$ keeps $\tilde{x}$ secret given $\tilde{y}$.*
2. *$\mathcal{A}(P;\tilde{X}^* \,|\, \tilde{Y}^*) = 0$, for some $\tilde{X}^* : \tilde{U}$ and $\tilde{Y}^* : \tilde{V}$ uniformly distributed and independent.*
3. *$\max_{\tilde{X}:\tilde{U},\tilde{Y}:\tilde{V}} \mathcal{A}(P;\tilde{X} \,|\, \tilde{Y}) = 0$.*
4. *$\phi \Leftrightarrow \exists \tilde{x}\tilde{x}'.\phi$, where $\phi = \mathrm{mgb}\big(P(\tilde{x},\tilde{y}),\, P(\tilde{x}',\tilde{y}')\big)$, for $\tilde{x}'$ and $\tilde{y}'$ tuples of distinct variables disjoint from $\tilde{x}$ and $\tilde{y}$, but of the same type.*

Example 4. Consider the following process, where $x,y : 1..4$:

$$Q(x,y) \stackrel{\text{def}}{=} (\nu c)\big(\overline{c} \,|\, [y=1]c.\overline{a}\big) + [x=2]\tau.\overline{a}\,.$$

It is immediate to see that Q does not keep x secret, given y. E.g., if the adversary knows that $y \neq 1$ and observes the behaviour $[\tau.\overline{a}]$ then he can infer that $x = 2$. In fact, the mgb *given by the theorem above is $\phi = \big([y=1] \to ([y'=1] \vee [x'=2])\big) \wedge \big([y'=1] \to ([y=1] \vee [x=2])\big)$, and clearly, $\phi \not\Leftrightarrow \exists xx'.\phi$. As an example, for X,Y independent and u.d on $1..4$, the leakage from X to Q given Y can be computed as $H(Z|Y) \approx 0.608$. The process $Q'(x,y) = Q(x,y) + [y \neq 1]\tau.\overline{a}$ keeps x secret given y.*

5 Rate of leakage

We assume now an attacker can only conduct upon P repeated experiments, each yielding a binary[1] answer, say success or failure. We are interested in the number of communications that are *necessary* for the adversary to extract one bit of information about $\tilde{X}$ in this way.

In the rest of the section, we fix $\sim$ to be *weak trace* equivalence (aka *may testing* equivalence [4, 2]) written $\simeq$, and defined as: $P \simeq Q$ iff for each trace s, $P \stackrel{s}{\Longrightarrow}$ iff $Q \stackrel{s}{\Longrightarrow}$.

[1] We expect no significant change in the theory if k-ary answers, with $k > 2$ fixed, were instead considered.

For the sake of simplicity, we shall only consider processes where channels transport tuples of values, i.e. we ban name-passing. For the same reason, we shall assume that no side-information is available to the attacker, i.e. $\tilde{y}$ is empty. We plan to present the treatment of the most general case in a full version of the present paper. Throughout the section and unless otherwise stated, $P(\tilde{x})$, where $\tilde{x}:\tilde{U}$, denotes an arbitrary open pi-process, $\tilde{X}$ an arbitrary vector of random variables of type $\tilde{U}$ and Z is $P(\tilde{X})$. Recall that $\mathcal{A}(P;\tilde{X}) = H(Z)$.

Definitions and basic properties Consistently with the testing equivalence framework [4, 2], we view an experiment E as a processes that, when composed in parallel with P, may succeed or not. Input on a distinct name ω, carrying no objects, is used to signal *success* to the adversary. Here, it is convenient to adjust the notion of composition ($\|$ below) to ensure that, in case of success, exactly one success action is reported to the adversary.

Definition 5 (experiments). *An* experiment *E is a closed process formed without using recursive definitions and possibly using the distinct success action ω.*

We say that a nonempty trace of visible actions s is successful for *E if ω does not occur in s and $E \stackrel{s\cdot\omega}{\Longrightarrow}$.*

For each E and process Q, let us define $Q\|E \stackrel{\text{def}}{=} (\nu\tilde{c},\omega')(P|E[\omega'/\omega]|\overline{\omega'}.\omega)$, where $\tilde{c} = \text{fn}(Q,E)\setminus\{\omega\}$ and $\omega' \notin \text{fn}(P,Q,\omega)$.

Note that for each Q it must be either $Q\|E \simeq \mathbf{0}$ – meaning that E fails – or $Q\|E \simeq \omega.\mathbf{0}$ – meaning that E succeeds. Hence, for each E, we can define a binary random variable thus[2]

$$E^* \stackrel{\text{def}}{=} P(\tilde{X})\|E\,.$$

Information on $\tilde{X}$ conveyed by E^* is $I(\tilde{X};E^*) = H(E^*) - H(E^*|\tilde{X}) = H(E^*)$. This information is at most one bit. The rate notion of rate we are after should involve a ratio between this quantity of information and the *cost* of E. The following example shows the role played by non-determinism in extracting information, and provides some indications as to what we should intend by cost.

Example 5. Consider again $Check(X)$, where this time X is u.d. over $1..k$, for some fixed even integer $k \geq 2$. An experiment E that extracts one bit out of $E \stackrel{\text{def}}{=} \sum_{d=1}^{k/2} \overline{a}d.ok.\omega$. An attacker can only observe the outcome of the interaction between Check and E, i.e. a sample of the r.v. $E^* = Check(X)\|E$. If action ω is observed, then it must be $X \leq k/2$; if action ω is not observed, then it must be $X > k/2$. Note that $I(X;E^*) = H(E^*) = H(\frac{1}{2}) = 1$.

The above example suggests that different successful traces of an experiment should be counted as different "trials" attempted by the attacker. The cost of each trial can be assumed to be proportional to its length as a trace. These considerations motivate the definition below.

Definition 6 (rate). *For each experiment E, define its* cost *as $|E| \stackrel{\text{def}}{=} \sum\{|s| : s \text{ is succesful for } E\}$. The* rate of P relative to $\tilde{X}$ *is*

$$\mathcal{R}(P;\tilde{X}) \stackrel{\text{def}}{=} \sup_{|E|>0} \frac{H(E^*)}{|E|}\,. \tag{7}$$

[2] We would write $E^*(P)$ should any confusion about P arise.

Our first result is an experiment-independent characterization of rate. In accordance with the may testing approach, this characterization is obtained in terms of observations of single traces. In what follows, given a trace of visible actions s, we consider the r.v. $P(\tilde{X}) \stackrel{s}{\Longrightarrow}$, which may yield $true$ or $false$, and denote by p_s the probability[3] $\Pr[(P(\tilde{X}) \stackrel{s}{\Longrightarrow}) = true]$. Recall that for $0 \leq p \leq 1$, we denote by $H(p)$ the entropy of the distribution $(p, 1-p)$.

Proposition 2. *It holds that* $\mathcal{R}(P;\tilde{X}) = \sup_{|s|>0} \frac{H(p_s)}{|s|}$.

Example 6. Consider the process $CheckOnce(x) \stackrel{\text{def}}{=} a(z).([z=x]\overline{ok} + [z \neq x]\overline{no})$, *where* $x, z : 1..10$, *and* X *u.d. on the same interval. It is immediate to verify that the ratio in the proposition above is maximized by any of* $s = ad \cdot \overline{ok}$ *or* $s = ad \cdot \overline{no}$, *for* $d \in 1..10$. *This yields* $\mathcal{R}(CheckOnce; X) = H(\frac{1}{10})/2 \approx 0.234$.

The proposition above allows one, at least in principle, to compute the rate of any process having a finite symbolic transition system. In fact, relying on P's symbolic transition system, it is possible to compute, for any given trace s, a logical formula $\phi_s(\tilde{x})$ expressing the exact condition on $\tilde{x}$ under which $P(\tilde{x})$ can perform s (see [7, 3]). From these formulae it is easy to compute, or at least estimate with any degree of precision, the rate of P – we omit the details.

The next result explains the relationship between the notion of rate and absolute leakage. In particular, (a) establishes that $H(Z)$ is the maximal information that can be extracted by *repeated* binary experiments; and (b) provides a lower bound on the cost necessary to extract this information, in terms of the rate of P – thus providing a justification for the name "rate". For $\tilde{E} = (E_1, E_2, ..., E_n)$ a vector of experiments, write $|\tilde{E}| = |E_1| + \cdots + |E_n|$ for its cost, and $\tilde{E}^*$ for the vector of r.v. $(E_1^*, E_2^*, ..., E_n^*)$.

Proposition 3. *It holds that*

$$\begin{array}{ll} (a) & \mathcal{A}(P;\tilde{X}) = H(Z) = \max_{\tilde{E}} I(\tilde{X};\tilde{E}^*) \\ (b) & \text{for each } \tilde{E},\ I(\tilde{X};\tilde{E}^*) \leq |\tilde{E}| \cdot \mathcal{R}(P;\tilde{X}) \ . \end{array}$$

Note in particular, that the cost of extracting *all* the available information $H(Z)$ cannot be less than $\frac{H(Z)}{\mathcal{R}(P;\tilde{X})}$. It is important to remark that processes with the same absolute leakage may well exhibit different rates. Here is a small example to illustrate this point.

Example 7. Let $P(x)$ *and* $Q(x)$, *where* $x : 0..3$, *be defined as follows:*

$$\begin{array}{l} P(x) = [x=0](a+b) + [x=1](b+c) + [x=2](c+d) + [x=3](d+a) \\ Q(x) = [x=0]a \quad\ + [x=1]b \quad\ + [x=2]c \quad\ + [x=3]d\,. \end{array}$$

Assume X *is u.d. over* $0..3$. *Both* $P(X)$ *and* $Q(X)$ *are u.d. on a domain of four elements (the four distinct equivalence classes* $[P(i)]$, *resp.* $[Q(i)]$, *for* $i \in 0..3$*). Hence leakage is* $H(P(X)) = H(Q(X)) = H(X) = 2$ *bits. On the other hand, each nonempty trace of* P *occurs with probability* $1/2$, *while each nonempty trace of* Q *occurs with probability* $1/4$. *Thus, by Proposition 2,* $\mathcal{R}(P;X) = H(\frac{1}{2}) = 1$ *and* $\mathcal{R}(Q;X) = H(\frac{1}{4}) \approx 0.811$. *Proposition 3(b) implies that gaining all information about* X *costs the attacker no less than* 2 *in the case of* P, *and no less than* 3 *in the case of* Q. *Indeed, a sequence of two (resp. three) one-action experiments is sufficient (*$\overline{a}.\omega, \overline{b}.\omega$ *for* P *and* $\overline{a}.\omega, \overline{b}.\omega, \overline{c}.\omega$ *for* Q*) to determine* X.

[3] It is important to note that this definition does *not* induce a probability distribution on the set of traces; rather, it assigns each trace s a binary distribution $(p_s, 1-p_s)$.

Compositionality It is possible to give upper bounds for the rate of a compound process in terms of the component expressions, in the vein of Proposition 1. Some of these upper bounds are rather crude (e.g. in the case of restriction), others are more sophisticated (e.g. $\mathcal{R}(\overline{a}e.P;\tilde{X}) \leq \max\{H([e(\tilde{X}) = v])),\ \mathcal{R}(P;\tilde{X})\}$) – we leave the details for the full version of the paper. Here, we concentrate on the rate of iterated processes. In order to define iteration, we have to first define sequential composition. Output on a distinct name *stop*, not carrying objects, is used to signal termination of a thread. Hence we define sequential composition as $P;Q \stackrel{\text{def}}{=} (\nu\ stop')(P[^{stop'}/_{stop}] \mid stop'.Q)$ (with $stop'$ fresh). This is not sequential composition in the usual sense, but it is equivalent in the context we are going to consider – see definition below. For any P, let iteration $*P$ be the process recursively defined by $*P \stackrel{\text{def}}{=} P;*P$. We show that, under suitable conditions, the rate of $*P$ is the same as P's. The condition below requires essentially that termination of a single thread in a process is equivalent to termination of the whole process.

Definition 7 (determinate processes). *Let Q be a closed process. We say that a trace s is* terminating for Q *if* $Q \stackrel{s\cdot\overline{stop}}{\Longrightarrow}$. *We say that Q is* determinate *if for every terminating trace s, whenever $Q \stackrel{s}{\Longrightarrow} Q'$ then $Q' \simeq \overline{stop}$. Finally, an open process $P(\tilde{x})$ is determinate if $\sum_{\tilde{u}\in\tilde{U}} P(\tilde{u})$ is determinate.*

We need another technical condition: let us say that Q is *stable* if whenever $Q \stackrel{\varepsilon}{\Longrightarrow} Q'$ (ε = empty trace) then $Q' \simeq Q$.

Theorem 2 (iteration rate). *Suppose that $P(\tilde{x})$ is determinate, and that for each $\tilde{u}$, $P(\tilde{u})$ is stable. Then $\mathcal{R}(*P;\tilde{X}) = \mathcal{R}(P;\tilde{X})$.*

Example 8. It is easy to check that $CheckOnceStop(x) \stackrel{\text{def}}{=} a(z).([z = x]\overline{ok}.\overline{stop} + [z \neq x]\overline{no}.\overline{stop})$ is determinate. ($x : 1..10$). Hence, being $Check(d) \simeq *CheckOnceStop(d)$, for every d, by Theorem 2 and Example 6 we have: $\mathcal{R}(Check;X) = \mathcal{R}(CheckOnceStop;X) = H(\frac{1}{10}) \approx 0.234$.

6 Computing bounds on absolute leakage

In this section, we analyze the problem of bounding absolute leakage, from the position of someone – e.g. a developer – who has access to the process' code P, and for whom it is inexpensive to draw independent samples of the data $\tilde{X}$. For simplicity, we shall limit our discussion to the case where the side-information $\tilde{Y}$ is empty, so that absolute leakage reduces to $H(Z)$, where $Z = P(\tilde{X})$. The problem is nontrivial, because even for moderately complex P, the distribution of Z may be extremely difficult to compute or approximate. Methods commonly employed to estimate entropy in absence of an explicit description of distribution involve generation of sample sequences, long enough to let the underlying source's redundancy become appreciable. These methods are not applicable to our case, as operating on samples of Z is extremely expensive. Generation of even a *single* sample of Z – that is, an equivalence class, represented in some form or another – generally takes exponential time and space in the size of P.

We suggest a strategy that may work in practice in a number of cases, but we will not dwell on complexity-theoretical issues. For any discrete random variable W, its *index*

of coincidence $IC(W)$ is defined as the probability that two independent experiments yield the same result, that is, denoting by U the type of W:

$$IC(W) \stackrel{\text{def}}{=} \sum_{u \in U} \left(\Pr[X = u]\right)^2 .$$

Relationship of *IC* with Shannon entropy is seen by applying a well-known inequality of convex functions (Jensen's inequality, see e.g. [15]), which yields: $-\log IC(W) \leq H(W)$ (the quantity on the LHS is known as *Renyi's entropy of order 2*.) This inequality has been vastly generalized by Harremoës and Topsøe [8], who provide whole families of lower- and *upper*-bounds of Shannon entropy in terms of *IC*. These bounds are, in a certain technical sense, the "best possible" and provide fairly good estimates of $H(W)$ in many cases[4]. It remains to be seen how $IC(W)$ can be efficiently estimated in our case ($W = Z$). We show that this can be achieved via mgb's. Let $\phi(\tilde{x}, \tilde{x}') \stackrel{\text{def}}{=} \text{mgb}\big(P(\tilde{x}), P(\tilde{x}')\big)$, where $\tilde{x}'$ is a tuple of distinct variables disjoint from $\tilde{x}$. By interpreting the boolean values true and false as 1 and 0, $\phi(\tilde{x}, \tilde{x}')$ can be interpreted as a function $\tilde{U} \times \tilde{U} \to \{0,1\}$. We then have the following proposition, based on elementary reasoning on probabilities.

Proposition 4. *Let $\tilde{X}'$ be independent from $\tilde{X}$, but with the same type and distribution as $\tilde{X}$. Then $IC(Z) = E[\phi(\tilde{X}, \tilde{X}')]$.*

The expectation $E[\phi(\tilde{X}, \tilde{X}')]$ can be estimated with any desired precision via the law of large numbers: in practice, one draws several independent samples of $\phi(\tilde{X}, \tilde{X}')$ and then takes the resulting arithmetical mean. The efficiency of this procedure depends on the distribution of $\tilde{X}$ and on the size of ϕ. Therefore, the problem of evaluating $IC(Z)$ can be reduced to the task of computing the formula ϕ, and possibly reducing its size by means of logical simplifications. Dedicated algorithms exist for that (see [7]) which are practical in many cases. Using this methodology, we have conducted some simple but very encouraging experiments on timing-dependent leakage in modular exponentiation algorithms (see e.g. [9]) that will be reported in the extended version of the paper.

7 Conclusions and related work

Results and proofs presented here carry over essentially unchanged to other calculi equipped with behavioral equivalences, such as the spi-calculus – except for those that depend on pi's symbolic semantics, like effective computation of leakage. The examples considered in the paper are admittedly a bit artificial. More realistic case-studies, possibly involving cryptography or probabilistic behaviour, are needed for assessing the model's scalability. In the leakage rate scenario, different notions of "cost" are also worthwhile to be investigated.

Early works on quantitative information flow are [13, 17, 6]. Volpano and Smith have later developed a quantified theory of non-interference for imperative programs, also giving a notion of rate [16], albeit not based on information theory. These approaches, like the one by Clark et al. [12], presuppose that computations produce some form or another of "result" , possibly with an associated probability distribution, in the sense already discussed in the introduction. A notable exception is represented by the recent work of Lowe [11]. There, quantitative non-interference for timed CSP is defined

[4] As an example, in the case of binary distributions $(p, 1-p)$, an upper bound U can be given s.t. the ratio H/U lies between 1 and 0.9 for all distributions with $p \in [0.03, 0.97]$.

as the number of different "low" behaviours that a "high" user can induce on the process. This definition is shown to be in agreement with a qualitative notion of lack of information flow due to Focardi and Gorrieri [5]. A notion of rate is also introduced by taking time explicitly into account. These notions are not easily comparable to ours, due to the different goals and settings (secrecy vs. non-interference, untimed vs. timed.)

Acknowledgements The reviewers comments have been very useful to improve on the presentation. Valentina Fedi has conducted several experiments with the metodology described in Section 6 as part of her MSC dissertation.

References

1. M. Abadi and A. Gordon. A calculus for cryptographic protocols: The Spi-calculus. *Information and Computation*, 148(1): 1-70, 1999.
2. M. Boreale and R. De Nicola. Testing equivalence for mobile processes. *Information and Computation*, 120(2): 279-303, 1995.
3. M. Boreale and R. De Nicola. A symbolic semantics for the pi-calculus. *Information and Computation*, 126(1): 34-52, 1996.
4. R. De Nicola and M.C.B. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.
5. R. Focardi and R. Gorrieri. A classification of security properties. *Journal of Computer Security*, 3(1): 5-34, 1995).
6. J.W. Gray, III. Towards a mathematical foundation for information flow security. In *Proc. of 1991 IEEE Symposium on Research in Computer Security and Privacy*, 1991.
7. M.C.B. Hennessy and H. Lin. Symbolic bisimulations. *Theoretical Computer Science*, 138(2): 353-389, 1995.
8. P. Harremoës and F. Topsøe. Inequalities between entropy and index of coincidence derived from information diagrams. *IEEE Transactions on Information Theory*, 47(7): 2944-2960, 2001.
9. P. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. *CRYPTO 1996*: 104-113, 1996.
10. P. Kocher, J. Jaffe and B. Jun. Differential Power Analysis. *CRYPTO 1999*: 388-397, 1999.
11. G. Lowe. Defining information flow quantity. *Journal of Computer Security*, 12(3-4): 619-653, 2004).
12. D. Clark, S. Hunt and P. Malacaria. Quantitative Analysis of the Leakage of Confidential Data. *Electr. Notes Theor. Comput. Sci.*, 59(3), 2001.
13. J. Millen. Covert channel capacity. In *Proc. of 1987 IEEE Symposium on Research in Computer Security and Privacy*, 1987.
14. D. Sangiorgi and D. Walker. *The pi-calculus: A Theory of Mobile Processes*. Cambridge University Press, 2001.
15. F. Topsøe. Basic concepts, identities and inequalities – the Toolkit of Information Theory. *Entropy*, 3:162–190, 2001. Also available at `http://www.math.ku.dk/~topsoe/toolkitfinal.pdf`.
16. D. Volpano and G. Smith. Verifying Secrets and Relative Secrecy. In *POPL 2000*, 268-276, 2000.
17. J.T. Wittbold and D. Johnson. Information flow in nondeterministic systems. In *Proc. of 1990 IEEE Symposium on Research in Computer Security and Privacy*, 1990.