A Formal Framework for Confidentiality-Preserving Refinement

Thomas Santen

Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin, Germany
santen@acm.org

Abstract. Based on a system model consisting of processes describing the machine, the honest users and the adversary, this paper introduces an abstract framework of refinement relations preserving existential confidentiality properties for nondeterministic, probabilistic systems. It allows a refinement step to trade functionality between the machine and its environment, thus shifting the conceptual boundary between machine and environment. A refinement also permits the realization to extend the observational means of an adversary. A confidentiality-preserving refinement relation is defined in terms of another, more basic relation that considers deterministic probabilistic processes. An instantiation with an entropy-based confidentiality property illustrates the use of this framework. The relationship to other concepts of secure refinement, in particular to reactive simulatability, is discussed.

1 Introduction

The paradigm of system and software development by stepwise refinement is a long standing one. Although rarely practiced rigorously, it provides the formal semantical justification for modern techniques such as behavioral subtyping for object-oriented inheritance [15], design by contract [21], and model-driven development. In its rigorous form, it has been applied, among others, in safety-critical applications [4].

The general idea of stepwise refinement is to first capture the essential requirements on a system in a concise model, the initial specification, that abstracts from all unnecessary detail and leaves room for subsequent design decisions. In a refinement step, two models, the *(abstract) specification* and the *(concrete) realization* are related by a preorder on models, the *refinement relation*. Compared to the specification, the refinement relation may admit to reduce nondeterminism, change the types of data, or replace atomic actions by sequences of "more primitive" actions. This process terminates with a completely refined model, the *implementation*, which is supposed to be easily transformable to a conventional program. A refinement relation should preserve the essential properties of the specification, be a preorder, and be compositional, which means that replacing a sub-specification by a realization in the context of a model yields a realization of that model, i.e., contexts are monotonic functions with respect to the refinement preorder.

Starting with [11], there is a vast body of research (e.g., [1, 7, 13]) investigating refinement relations that preserve *functional* properties such as deadlock freedom or the observational behavior of an abstract data type. The preservation theorems for those refinement relations are *universal* in the following sense: they guarantee that *any* realization refining a given specification has the same (suitably rephrased) properties as the specification.

Considering the development of secure systems, confidentiality poses particular problems. It is well-known that refinement relations which allow one to reduce nondeterminism [12] do not preserve confidentiality properties, such as possibilistic information flow properties, of nondeterministic specifications. Roscoe [24] called this the *refinement paradox*. Several approaches to deal with this deficiency of classical refinement relations have been proposed. Roscoe et al. [26] avoid the problem by requiring the adversary's view of the system to be deterministic. Similarly, Jürjens [14] distinguishes specification nondeterminism from implementation nondeterminism, and disallows specification nondeterminism whenever it influences the validity of a security property. Mantel [18] shows how refinement operators tailored for specific information flow properties can modify an intended realization such that the resulting realization preserves the given flow property. Ryan and Schneider [27] discuss the effects of nondeterminism on information flow properties in depth. They conceptually distinguish High nondeterminism and system nondeterminism to show where nondeterminism may influence information flow. That distinction somehow reflects the distinction between nondeterminism for specification purposes and the kind of nondeterminism induced by probabilistic choices at run-time.

Contributions. In the present paper, we take a different view to the problem. Accepting that the particular way of resolving nondeterminism influences the confidentiality properties of the resulting realization, we investigate refinement relations that preserve the *existence* if a secure implementation: they keep invariant the existence of a deterministic realization of the given model that satisfies a given confidentiality property.

We consider a system model including the machine[1] to be built as well as its environment. The environment consists of a model of the honest users and a model of the adversary. This allows one to model systems whose security depends on assumptions of user and adversary behavior, i.e., to make these assumptions explicit. This system model is an extension of the one we proposed before [10, 29]. It has some similarity with the one proposed by Backes et. al [3], which we discuss in Section 7.

Our models may use three kinds of choice: External choice, nondeterministic choice, and probabilistic choice. Jürjens' implementation nondeterminism basically is an abstraction of probabilistic choice. We consider it important to explicitly model the probabilistic behavior of systems because confidentiality is inherently probabilistic: a system keeps confidential which one of alternative system behaviors that produce the same observations for an adversary not by just producing the same observations, but their relative probabilities also must be approximately equal. Otherwise the adversary's risk of making a false guess would be (unacceptably) low. The underlying assumption of possibilistic models is that the (unknown) stochastic behavior of the system is such that it produces a sufficiently high risk for the adversary of guessing wrong. To include

[1] Following Jackson [32], we call the "system to be implemented" *machine* in order to distinguish it from the *system* comprising the machine *and* its environment.

probabilities in the analysis, however, does not mean that one needs to exactly know the stochastic behavior of the system: Realistic assumptions together with a robustness of the confidentiality properties that allows for a range of probabilities without destroying security are sufficient.

(Specification) nondeterminism, on the other hand, is important for abstract models, in which specifiers cannot or do not want to prematurely choose between alternative behavior. Furthermore, certain modeling operators in general introduce nondeterminism.

We propose an abstract notion of confidentiality preserving refinement. It builds on behavior refinement, which allows a realization to reduce nondeterminism and to change the data types of input / output data. The realization may also provide additional means for the adversary to observe the machine. The refinement relation is parameterized by another preorder, the *information flow refinement order*, which ensures the existential preservation of a given confidentiality property.

Since the data space and the means of observation may change, the preservation property refers to a *point of reference* describing the model to which the confidentiality property directly refers. In a sequence of refinement steps, the initial specification usually is the point of reference.

The system model and the refinement framework allow for different environments in the specification and the realization. Thus, *trading* of functionality between the environment and the machine can be accomplished in a refinement step.

The initial specification will usually be concise and the machine model may even be deterministic. Data refinements or trading can then produce nondeterminism in more detailed models, which will be resolved by implementation choices in subsequent refinement steps.

Finally, we instantiate the framework with a confidentiality property based on the entropy of classes of indistinguishable behavior, and with an information flow refinement order defined in terms of mutual information.

Overview. Section 2 sketches probabilistic CSP, the process calculus on which our framework is based. Section 3 introduces our general system model, consisting of processes describing the machine and its environment, as well as the adversary capabilities of observing the system. The structure of confidentiality properties is the topic of Section 4, and Section 5 presents the main result of this paper: the abstract definition of confidentiality-preserving refinement and the corresponding preservation theorem. Section 6 instantiates that framework for a specific confidentiality property. The reader may wish to browse this section first to aid understanding the abstract discussions in Sections 3 through 5. Section 7 discusses related work, before the conclusion gives some pointers to ongoing and future research.

2 Probabilistic Communicating Sequential Processes

To formally model the systems we reason about, we use the probabilistic extension PCSP of the process algebra CSP [25] which Morgan et al. [22] have proposed. We use PCSP because it integrates probabilistic choice with nondeterministic and external choice, and its semantics, in particular the semantics of probabilistic choice, is centered around the concept of refinement. This supports well our investigation of the relationship of refinement and confidentiality.

CSP. A *process* P produces sequences of *events*, called *traces*. An event $c.d$ consists of a channel name c and a data item d. Two processes can *synchronize* on a channel c by transmitting the same data d over c. If one process generates an event $c.d$ and the other generates an event $c.x$, where x is a variable, both processes exchange data when synchronizing on channel c: the value of x becomes d. The set of traces of P is $traces(P)$. The length of a trace t is $\#t$.

The process $e \rightarrow P$ first generates event e, and behaves like P afterwards. The process $P \,|[\, X \,]|\, Q$ is a parallel composition of P and Q synchronized on the channels in X: if P or Q generate events on channels not in X, then those events appear in an arbitrary order; if a process generates an event on a channel in X, it waits until the other process also generates an event on the same channel; if the data transmitted by both processes are equal (or can be made equal because an event contains a variable), then the parallel composition generates that event, otherwise the parallel composition deadlocks. The composition $P \Downarrow_X Q$ asynchronously transmits data from P to Q and synchronizes the processes on the remaining channels, such that the behavior of Q on X cannot influence P. The definition of $P \Downarrow_X Q$ involves a buffering process that collects events from P on X and forwards them to Q while blocking any flow of events from Q to P through X.

Although not uniformly definable in terms of the standard CSP operators, $P \Downarrow_X Q$ can be constructed for any given P, Q, and X.

In the notion of refinement we use, we are interested in changing data representations of the communicated data (*I/O refinement*), because many effects compromising confidentiality can be described by distinguishing data representations in an implementation that represent the same abstract data item (e.g., different representations of the same natural number). For a relation R on data, the process[2] $P[\![R]\!]_D$ is the process P where each data item a in events of P is replaced by a data item b that is in relation with a, i.e., $a\,R\,b$ holds.

The process $P \setminus X$ is distinguished from P by *hiding* the channels in $X \subseteq \alpha P$, where αP is the set of channels used by P. The traces of $P \setminus X$ are the traces of P where all events over channels in X are removed. The external choice $P \,\Box\, Q$ is the process that behaves like either P or Q, depending on the event that the environment offers. For a family of processes $P(x)$, the process $\bigsqcap P(x)$ nondeterministically behaves like one of the $P(x)$. The process $P \sqcap Q$ nondeterministically behaves like P or like Q.

There are several refinement relations for standard CSP. Most commonly, one uses the failures/divergences refinement. Informally, the process Q refines the process P, written $P \sqsubseteq Q$, if Q is more deterministic and less diverging than P. For details, see [25].

For $n \in \mathbb{N}$, the *finite approximation* $P \downarrow n$ of P behaves like P for the first n events and diverges afterwards. Any process P is characterized by its finite approximations. It

[2] The subscript D indicates that this variant of relational renaming does not change the channel names but only the communicated data.

is their least upper bound with respect to the refinement order: $P = \bigsqcup n : \mathbb{N} \bullet P \downarrow n$. A process F that diverges after n events is called a *finite* process.

The *cone* $P\uparrow$ of a process P is the set of all refinements of P, $P\uparrow = \{Q : CSP \mid P \sqsubseteq Q\}$.

Probabilistic CSP. Morgan et al. [22] extend standard CSP by a probabilistic choice operator: The process $P\,{}_p\!\oplus Q$ behaves like P with a probability of p, and it behaves like Q with a probability of $1-p$. This view of probabilistic processes does not appeal to the intuition that a process chooses particular behavior (a trace or a failure) probabilistically. It rather emphasizes that a probabilistic process behaves like a standard process with a certain probability. Although it may seem unfamiliar at first sight, this view leads to a smooth integration of probabilistic choice with the other operators of CSP, in particular with nondeterministic choice.

The semantics of PCSP relies on continuous evaluations over a Scott topology of the inductive partial ordering $(CSP, \sqsubseteq)$. We can only present the essential concepts relevant to our work here. See [22] for further detail.

The set of *probabilistic processes PCSP* is the set of continuous evaluations mapping "Scott-open" sets of standard processes to $[0,1]$ over the failure-divergences model of CSP under the refinement order, $(CSP, \sqsubseteq)$. Let P and Q be probabilistic processes in $PCSP$. The process Q *refines* P iff for all Scott-open $Y \subseteq CSP$: $P(Y) \leq Q(Y)$. Since standard processes can be embedded in PCSP and the refinement orders coincide, we write $P \sqsubseteq Q$ for PCSP refinement, too.

For a finite standard process F and a probabilistic process P, we write $F \precsim P$ for the probability that P is a member of $F\uparrow$. If $P \sqsubseteq Q$ then it also holds for all finite $F \in CSP$ that $F \precsim P \leq F \precsim Q$.

For processes P, Q in $PCSP$, and $p \in [0,1]$, the *probabilistic choice* of P and Q is defined for all Scott-open subsets Y of CSP:

$$(P\,{}_p\!\oplus Q)(Y) = p \cdot P(Y) + (1-p) \cdot Q(Y)$$

Because the cone of a finite process is Scott-open, the following relationship between the probability of refining a finite process and probabilistic choice holds. For P, Q in $PCSP$, finite F in CSP and probability p,

$$F \precsim P\,{}_p\!\oplus Q = p \cdot (F \precsim P) + (1-p) \cdot (F \precsim Q)$$

Furthermore, any non-recursive probabilistic process can be expressed as a probabilistic choice of finitely many standard processes, because probabilistic choice distributes through all (embedded) operators of CSP.

Finally, we remark that nondeterministic choice generalizes probabilistic choice (for any probability p) and external choice in PCSP, whereas probabilistic choice and external choice are not related by refinement.

$$P \sqcap Q \sqsubseteq \begin{cases} P\,{}_p\!\oplus Q \\ P \,\square\, Q \end{cases}$$

The indexed probabilistic choice $\bigoplus_{i:I}^{\mathcal{P}} P_i$ canonically generalizes the binary operator for finite index sets I: this process chooses i – and thus P_i – with probability $\mathcal{P}(i)$.

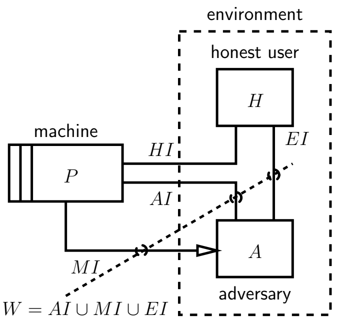

Fig. 1. A system consists of a machine and its environment

Probabilistic Linear Processes. We consider probabilistic confidentiality properties that refer to the probability of a process QE performing a trace t, i.e., the process QE is considered a random variable on traces. This is possible only if QE is deterministic, does not admit external choice, and if the length of the considered traces of QE is bounded by some natural number k. The latter is necessary to distinguish a trace t from a prefix s of t if QE may block after s. Then s and t refer to different probabilistic events.

To resolve nondeterministic and external choices, we consider the set $P^{\top}$ of all *maximal* refinements of P. The members Q of $P^{\top}$ are probabilistic deterministic, i.e., they are free of nondeterminism, but they still admit external choices. The latter are resolved by means of an environment process E that probabilistically resolves external choices of Q and thus serves as a scheduler [30, 6]. We call a process E achieving this in the k-approximation of the composition $Q \parallel_{\to W} E$ an *admissible* environment. Admissibility is characterized by the fact that $(Q \parallel_{\to W} E) \downarrow k$ is *probabilistic linear*, i.e., there is a probability function $\mathcal{P}_E$ such that

$$(Q \parallel_{\to W} E) \downarrow k = \bigoplus\nolimits^{\mathcal{P}_E}_{t \in traces(Q) \downarrow k} \mathrm{Fin}_k(t)$$

where $\mathrm{Fin}_k(t)$ is the process producing the first k events of t and diverging afterwards. If the length of t is less than k then $\mathrm{Fin}_k(t)$ deadlocks after t.

The environment process E is admissible for an arbitrary process P if it is admissible for all $Q \in P^{\top}$. For $QE = (Q \parallel_{\to W} E) \downarrow k$ the probability $\mathcal{P}_E(t)$ is $\mathrm{Fin}_k(t) \sqsubseteq QE$. Therefore, we define

$$\mathrm{Pr}^k_{QE}(t) = (\mathrm{Fin}_k(t) \sqsubseteq QE)$$

This is the probability of QE producing exactly the first $\min\{k, \#t\}$ events of t.

3 System Model

A system consists of three PCSP processes, as shown in Figure 1: the machine P, the (honest) user environment H, and the adversary environment A. The machine synchronizes with the adversary via the channels in the (functional) *adversary interface* AI.

Additionally, the adversary can observe the machine on the channels in the *monitoring interface MI*, and it can interact with the honest users on the *environment interface EI*. The sets of channels AI, EI, and MI partition the channels of A. An *adversary model* $(P, A, H, HI, AI, MI, EI, k)$ additionally determines a bound k on the length of system traces that are to be considered. The union of the adversary interfaces $W = AI \cup EI \cup MI$ is the *adversary window*. By convention, we use W (with suitable indexes) to denote adversary windows.

The set $\mathcal{E}_{P,k}^{EI,MI}(H,A)$ comprises all deterministic admissible environment processes.

$$\mathcal{E}_{P,k}^{EI,MI}(H,A) = \{H_d \,|[EI]|\, A_d \mid H_d \in H^\top \wedge A_d \in A^\top \\ \wedge\, H_d \,|[EI]|\, A_d \text{ admissible for } P, W, k\}$$

To make assertions about the probabilistic behavior of an adversary model means to consider the probabilistic linear processes $(Q \Downarrow_{MI} E) \downarrow k$ where $Q \in P^\top$ and $E \in \mathcal{E}_{P,k}^{EI,MI}(H,A)$. Those processes refine the system in its environment, and we call them the *variants* of the adversary model:

$$P \Downarrow_{MI} (H \,|[EI]|\, A) \sqsubseteq Q \Downarrow_{MI} E$$

An adversary model captures a model of the machine to be built together with assumptions on the behavior of the honest users and an adversary. The interfaces AI and EI allow the adversary to actively influence the machine and the honest users (permitting active attacks on the users). The user environment can also allow an adversary to compromise users during a system run.

The bound k not only reflects a technical necessity but also a realistic assumption: Associating time units with system events, it restricts the time an adversary can spend on attacks. It is a parameter of the concepts we introduce in the following.

It is possible to strengthen these concepts, requiring them to hold for all k and using an inductive argument to establish the required properties. However, we will restrict the presentation and consider a fixed k only.

4 Confidentiality Properties

This section discusses a common abstraction of the confidentiality properties of adversary models. In particular, it motivates the existential nature of those properties.

Basic Confidentiality Properties. The concept of indistinguishable traces is the foundation for defining confidentiality properties of adversary models. Given a set of channels W, two traces $s, t \in \mathit{traces}(P)$ of a process P are *indistinguishable by* W (denoted $s \equiv_W t$) if their projections to W are equal:

$$s \equiv_W t \Leftrightarrow s \upharpoonright W = t \upharpoonright W$$

where $s \upharpoonright W$ is the projection of s to the sequence of events on W.

Indistinguishability induces a partition on the trace set of a process. We are particularly interested in the traces up to the length of k. The set $J_W^{P,k}(o)$ contains the traces of P with a length of at most k that produce the observation o on W. The set $\mathrm{Obs}_W^k(P)$ comprises all observations P produces with traces that are no longer than k.

$$\begin{aligned} J_W^{P,k}(o) &= \{t : traces(P) \mid t \upharpoonright W = o \wedge \#t \leqslant k\} \\ \mathrm{Obs}_W^k(P) &= \{t \upharpoonright W \mid t \in traces(P) \wedge \#t \leqslant k\} \end{aligned}$$

The traces that are indistinguishable by an adversary window W are the ones that an adversary cannot obviously distinguish. Given an observation o, the adversary does not know which member of $J_W^{P,k}(o)$ caused the observation (unless that set is a singleton).

In earlier work [28], we have discussed several confidentiality properties based on indistinguishability. *Possibilistic* confidentiality properties, such as the various information flow properties that Mantel [19] analyzes, basically require at least one alternative indistinguishable behavior to exist for any given one, according to the system design. They neither distinguish systems with respect to the number of alternative behaviors, nor with respect to the degree of evidence (in any suitable measure) an adversary might assign to the alternative behaviors in question. We are primarily interested in *probabilistic* confidentiality properties. These define the "degree of evidence" of alternative behaviors based on the probabilistic behavior of the system in a given environment. Therefore, we focus on predicates $\mathcal{CP}(QE, W, k)$ depending on a probabilistic linear process QE (a realization of the machine process in an admissible environment), an adversary window W and the length bound k. We call such a property a *basic confidentiality property*.

We do not further characterize basic confidentiality properties here. Section 6 discusses an example. In the following discussion of the structure of confidentiality properties, a predicate $\mathcal{CP}(QE, W, k)$ serves as a placeholder.

Structure of Confidentiality Properties. It is not obvious for an adversary model $(P, A, H, HI, AI, MI, EI, k)$ which refinements QE of a given machine P in an admissible environment $E \in \mathcal{E}_{P,k}^{EI,MI}(H, A)$ must satisfy $\mathcal{CP}(QE, W, k)$ for the adversary model to satisfy a confidentiality property based on $\mathcal{CP}(QE, W, k)$.

As already indicated in Section 1, the refinement paradox motivates the *existential* nature of confidentiality properties.

Since it has become known that possibilistic information flow properties are closure properties [19, 20], the observation that refinement does not preserve confidentiality in general is not so surprising anymore: refinement reduces nondeterminism and thus diminishes the set of traces, which is the definition of trace refinement. A closure property requires that, given a member of a set, certain other items are also members of that set. Therefore, a trace refinement, in general, does not preserve closure properties.

These considerations show that we cannot expect *all* refinements QE of a system in its environment to satisfy a given basic confidentiality property $\mathcal{CP}(QE, W, k)$ with respect to the adversary window W and the trace bound k, unless we can exclude "specification nondeterminism" in P. However, this is hardly possible in the current theory of probabilistic (and standard) CSP for two reasons. A technical reason is that hiding and data renaming almost inevitably introduce nondeterminism. Methodologically, the nondeterministic choice of CSP has an interpretation as "execution time nondeterminism", because it is *demonic* and must be considered to be resolved "after" all probabilistic and external choices. On the other hand, it is refined by probabilistic and external choice, as well. Thus, the definition of CSP refinement clearly considers nondeterminism as a means of postponing implementation decisions. From a methodological point of view, it is also necessary to allow P to contain "specification nondeterminism", because P actually *is* a specification and, as such, must provide ways of abstracting from design decisions including decisions on how the system chooses alternative behavior.

The environment, in contrast to the machine process, must be considered with all variations that the adversary model permits. Analyzing a system for security with respect to a *single* admissible realization $H_0 \,|[EI]|\, A_0$ of the user and adversary environment would yield a quite weak result. Instead, all $E \in \mathcal{E}_{P,k}^{EI,MI}(H,A)$ need to be considered for evaluating the security of a system.

This argument shows that, although we inevitably need to consider an environment process E describing user and adversary behavior to obtain a probabilistic analysis of security properties of a system, we must not restrict the analysis to one particular such process but we must carry out that analysis for all members of $\mathcal{E}_{P,k}^{EI,MI}(H,A)$. In particular, this allows an analysis to consider arbitrary adversary. Taking the chaotic process *Chaos* as the adversary environment models the most liberal assumption about the adversary behavior, because *Chaos* is refined by any other process.

We conclude that an adversary model satisfies a confidentiality property that is defined in terms of a basic confidentiality property $\mathcal{CP}$ if *there is* a probabilistic linear realization of the machine process that satisfies $\mathcal{CP}$ in all admissible environments. Therefore, a confidentiality property has the general form:

$$\exists\, Q : P^{\top} \bullet \forall\, E : \mathcal{E}_{P,k}^{EI,MI}(H,A) \bullet \textbf{let } QE = (Q \Vert_{MI} E) \downarrow k \bullet \mathcal{CP}(QE, W, k)$$

This definition avoids the refinement paradox, because it explicitly states that not necessarily all functionally correct realizations are supposed to be secure but that at least one realization needs to exist that is. It also avoids the misconception that a system will be secure in any working environment but makes the admissible working conditions and the constraints on the behavior of adversaries explicit.

Remark 1. Other "non-functional" requirements have a similar "existential" nature: To be adequate for a system with real-time performance requirements, for example, a model must admit a performing implementation, but not all functionally correct implementations of the model necessarily satisfy the real-time constraints.

5 Confidentiality Preserving Refinement

This section discusses a definition of refinement of adversary models that preserves a given confidentiality property. The refinement relation allows the realization to be more deterministic, to change the communicated data, to shift the responsibility to realize behavior from the environment to the machine, and to extend the adversary window, thus providing the adversary with new means of observation. The motivation[3] to consider

[3] Refer to [10] for a more detailed motivation.

these variation points lies in the fact that moving from an initial specification to an implementation, the concepts the adversary models need to consider necessarily include more detailed descriptions of the processed data and also more intricate ways of the adversary to observe the system. Furthermore, the interpretation of an adversary model differs depending on its role in a refinement: as a specification, an adversary model reflects what an adversary *is allowed* to observe (and to do); as a realization, an adversary model describes what an adversary *can* observe (or do). Therefore, the refinement relation needs to ensure that an adversary's abilities do not exceed his permissions: if the specification satisfies a confidentiality property then the realization must satisfy a similar confidentiality property that is "re-abstracted" to the data model of the first confidentiality property.

In the following, we first introduce behavior refinement. Then, we define a re-abstraction preorder on the variants of the specification and the realization adversary models. The definition of confidentiality preserving refinement (CPR) refers to another preorder, the information flow refinement order: CPR holds if the re-abstraction and the information flow refinement preorders coincide on the adversary models in question.

Behavior Refinement. Allowing the refining process to communicate different data than the refined process, behavior refinement generalizes PCSP refinement. Of course, the change of data must not be completely arbitrary but there must be a relation between the concrete and the abstract data that is compatible to PCSP refinement. A *retrieve relation R* maps the data of the concrete process Q to the data of the abstract process P, i.e., it is total on the data of Q and its range is in the data of P.

A retrieve relation R abstracts away the additional detail of the concrete data to "retrieve" the abstract data that the concrete data implements. The following definition of behavior refinement uses a retrieve relation to abstract the data of the refining process before comparing that "data abstracted" process to the refined process with PCSP refinement. With data renaming, we have a CSP operator at hand to perform the data abstraction.

Definition 1 (Behavior Refinement). *Let P and Q be probabilistic processes. Let R be a retrieve relation from Q to P. Then Q* behaviorally refines *P* via *R (written $P \sqsubseteq_R Q$), if $P \sqsubseteq Q[\![R]\!]_D$.*

Behavior refinement allows Q to resolve nondeterminism in P (as usual either by external or by probabilistic choice). Additionally, it offers *new* implementation freedom for Q if R maps several data items c_{i,k_i} of Q to the same abstract data item a_i of P. In particular, if P offers a probabilistic choice between several $e_1.a_i$ and $e_2.a_j$, then the refinement condition $P \sqsubseteq Q[\![R]\!]_D$ requires Q to produce $e_1.c_{i,k_i}$ and $e_2.c_{j,k_j}$ for *some* k_i and k_j with the same distribution as P, but it does not prescribe the choice of the k_i and k_j, which Q may choose nondeterministically.

Extending behavior refinement to adversary models, there are two points to clarify: first, how can the relationship between system process and environment change in a refinement; and second, how do the adversary windows relate?

The following Definition 2 allows a refinement to change the "responsibility" of the machine and its environment to establish certain behavior. It relates the abstract and concrete machines *in their respective environments*. It does not require that the concrete machine as such refines the abstract one (and the environment processes relate similarly).

A central objective of our investigation on confidentiality preserving refinement is to clarify the conditions under which the adversary's observational power may change securely under refinement. Definition 2 allows the refining adversary model to extend the adversary window, i.e., it requires $W_a \subseteq W_c$. The additional channels in W_c give the adversary means of observing the system that are not present in the abstract adversary model. The behavior refinement does not relate those means of observation to the abstract model, which the definition reflects by hiding $W_c - W_a$. Thus it allows the adversary to make arbitrary additional observations. In the rest of this section, we addresses the question whether those additional observations affect the security of the system.

Definition 2 (Behavior Refinement of Adversary Models). *Let two adversary models with identical bound k be given:* $\mathcal{A} = (P_a, A_a, H_a, HI_a, AI_a, MI_a, EI_a, k)$ *and* $\mathcal{C} = (P_c, A_c, H_c, HI_c, AI_c, MI_c, EI_c, k)$. *The realization* $\mathcal{C}$ behaviorally refines *the specification* $\mathcal{A}$ *via the retrieve relation* R_{ca} *(written* $\mathcal{A} \sqsubseteq_{R_{ca}} \mathcal{C}$*) if* $W_a \subseteq W_c$ *and*

$$P_a \Lbag\!\!\!\rightarrow_{MI_a} (H_a \,|[EI_a]|\, A_a) \sqsubseteq_{R_{ca}} (P_c \Lbag\!\!\!\rightarrow_{MI_c} (H_c \,|[EI_c]|\, A_c)) \setminus (W_c - W_a)$$

To refine a specification to an implementation in a stepwise fashion, any refinement relation must be a preorder, i.e., be reflexive and transitive for an appropriate choice of retrieve relations. Behavior refinement inherits these properties from PCSP refinement, i.e., $\mathcal{A} \sqsubseteq_{\mathrm{id}} \mathcal{A}$ and $\mathcal{A} \sqsubseteq_{R_{ba}} \mathcal{B} \wedge \mathcal{B} \sqsubseteq_{R_{cb}} \mathcal{C} \Rightarrow \mathcal{A} \sqsubseteq_{R_{cb} \circ R_{ba}} \mathcal{C}$ hold.

Re-Abstraction. Basic confidentiality properties refer to the variants of adversary models, and CPR must place conditions on the "matching" variants of the specification and the realization in order to ensure preservation of the property. Re-abstraction relates the variants of the specification to the "data abstracted" variants of the realization. By definition, variants are probabilistic linear (before diverging after k events). This means that a variant of the specification cannot be refined further (up to k). Data renaming a variant of the realization, however, may introduce nondeterminism. Therefore, there may be several "matching" variants of the specification for a given variant of the realization. These are exactly the ones that the re-abstraction selects.

Definition 3 (Re-Abstracted Refinement). *Let* R_{ca} *be a retrieve relation from the data of* QE_c *to the data of* QE_a. *Let* W_a *and* W_c *be sets of channels of* QE_a *and* QE_c, *respectively, such that* $W_a \subseteq W_c$. *Then the re-abstracted refinement of* (QE_c, W_c) *by* (QE_a, W_a), *denoted* $(QE_c, W_c) \overset{\nwarrow}{\sqsupseteq}_{R_{ca}} (QE_a, W_a)$, *is defined by*

$$(QE_a, W_a) \overset{\nwarrow}{\sqsupseteq}_{R_{ca}} (QE_c, W_c) \Leftrightarrow QE_c \setminus (W_c - W_a)[\![R_{ca}]\!]_D \sqsubseteq QE_a$$

Similar to behavior refinement, re-abstraction is a preorder.

Information Flow Refinement. A behavioral refinement possibly refines the data which the processes communicate, and it may also change the adversary window. A basic confidentiality property $\mathcal{CP}$ refers to the data and the adversary window of the abstract model. To determine whether a refined adversary model satisfies the same confidentiality property, it is in general necessary to relate the concrete data and adversary window back to the abstract ones, to which $\mathcal{CP}$ originally refers. In a sequence of refinement steps, one usually wishes to relate back to the confidentiality property of the initial specification.

To capture this formally, we say that a *refined basic confidentiality property* $\mathcal{CP}_r(QE, W, W_r, R_r, k)$ refers to a *point of reference* consisting of an adversary window W_r and a retrieve relation R_r. A refined basic confidentiality property induces a simple one by the following equivalence:

$$\mathcal{CP}(QE, W, k) \Leftrightarrow \mathcal{CP}_r(QE, W, W, \mathrm{id}, k)$$

Re-abstraction relates the "matching" variants of the specification and the realization adversary models. The following concept of information flow refinement serves as an abstraction of the relationship that the matching variants must satisfy in order to preserve a given confidentiality property.

Definition 4 (Information Flow Refinement). *Let $\mathcal{CP}_r$ be a refined confidentiality property. A preorder $(QE_a, W_a) \preccurlyeq^k_{R_{ca}} (QE_c, W_c)$ on pairs of probabilistic linear processes and adversary windows is called an* information flow refinement *relation for $\mathcal{CP}_r$ with the point of reference (W_r, R_r) if it strengthens the re-abstraction preorder and it is sufficient to preserve $\mathcal{CP}_r$, i.e., for all adversary models $\mathcal{A} = (P_a, AI_a, W_a, k, H_a, A_a)$ and $\mathcal{C} = (P_c, AI_c, W_c, k, H_c, A_c)$ such that the domain of R_r comprises the data space of $\mathcal{A}$, and $\mathcal{A} \sqsubseteq_{R_{ca}} \mathcal{C}$ holds, the following is satisfied:*

$$\begin{aligned}
&\forall\, Q_a : P_a^{\top};\, Q_c : P_c^{\top};\; E_a : \mathcal{E}^{EI_a, MI_a}_{P_a, k}(H_a, A_a);\; E_c : \mathcal{E}^{EI_c, MI_c}_{P_c, k}(H_c, A_c) \bullet \\
&\quad \mathbf{let}\ QE_a = Q_a \Downarrow_{MI_a} E_a;\; QE_c = Q_c \Downarrow_{MI_c} E_c \bullet \\
&\qquad\quad (\,(QE_a, W_a) \preccurlyeq^k_{R_{ca}} (QE_c, W_c) \Rightarrow (QE_a, W_a) \overset{\nwarrow}{\sqsupseteq}_{R_{ca}} (QE_c, W_c)\,) \\
&\qquad \wedge (\,(QE_a, W_a) \preccurlyeq^k_{R_{ca}} (QE_c, W_c) \wedge \mathcal{CP}_r(QE_a, W_a, W_r, R_r, k) \\
&\qquad\qquad \Rightarrow \mathcal{CP}_r(QE_c, W_c, W_r, R_{ca} \fatsemi R_r, k)\,)
\end{aligned}$$

By definition, an information flow refinement relation is a subset of the re-abstraction relation. Usually, it makes sense only for variants that are related by re-abstraction. In the following, we will see that the crucial condition for confidentiality-preserving refinement requires that the reverse implication is true and the two preorders coincide on the variants of the adversary models in question.

CPR. Under which condition is a behavioral refinement of adversary models a confidentiality-preserving one? Due to the existential nature of confidentiality properties, it is not necessarily the case that a behavioral refinement admits a secure refinement at all. The realization might exclude all possible secure refinements even though the specification satisfies the confidentiality property, i.e., there is a variant of the specification satisfying the desired basic confidentiality property. The behavior refinement can only preserve confidentiality if there is a secure variant $\widehat{QE_a}$ of the specification that (PCSP-)

refines the re-abstracted realization. If this is the case, then we need to know that the re-abstracted variants of the realization matching $\widehat{QE_a}$ are secure, too. Confidentiality-preserving refinement guarantees the latter.

Definition 5 (Confidentiality-Preserving Refinement, CPR). *Let $\preccurlyeq$ be an information flow refinement relation. The adversary model $\mathcal{C}$ is a* confidentiality-preserving refinement (CPR) *of the adversary model $\mathcal{A}$ for $\preccurlyeq$ via the retrieve relation R_{ca} (written $\mathcal{A} \sqsubseteq^{\preccurlyeq}_{R_{ca}} \mathcal{C}$) if $\mathcal{A} \sqsubseteq_{R_{ca}} \mathcal{C}$ and the re-abstracted refinement of variants of $\mathcal{A}$ and $\mathcal{C}$ is sufficient for their information flow refinement:*

$$
\begin{aligned}
&\forall\, Q_a : P_a^{\top};\ Q_c : P_c^{\top};\ E_a : \mathcal{E}^{EI_a,MI_a}_{P_a,k}(H_a,A_a);\ E_c : \mathcal{E}^{EI_c,MI_c}_{P_c,k}(H_c,A_c) \bullet \\
&\quad \textbf{let } QE_a = Q_a \Downarrow\!\!\rightarrow_{MI_a} E_a;\ QE_c = Q_c \Downarrow\!\!\rightarrow_{MI_c} E_c \bullet \\
&\qquad\qquad (QE_a, W_a) \sqsupseteq^{\nwarrow}_{R_{ca}} (QE_c, W_c) \Rightarrow (QE_a, W_a) \preccurlyeq^{k}_{R_{ca}} (QE_c, W_c)
\end{aligned}
$$

In conjunction with the first implication in Definition 4, the definition of CPR implies that (given $\mathcal{A} \sqsubseteq_{R_{ca}} \mathcal{C}$) $\mathcal{A} \sqsubseteq^{\preccurlyeq}_{R_{ca}} \mathcal{C}$ is equivalent to the identity of information flow refinement and re-abstraction on the variants of $\mathcal{A}$ and $\mathcal{C}$.

The following lemma establishes that CPR is a well-behaved refinement relation.

Lemma 1 (CPR is a Preorder). *For all adversary models $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, CPR satisfies $\mathcal{A} \sqsubseteq^{\preccurlyeq}_{\mathrm{id}} \mathcal{A}$ and $\mathcal{A} \sqsubseteq^{\preccurlyeq}_{R_{ba}} \mathcal{B} \wedge \mathcal{B} \sqsubseteq^{\preccurlyeq}_{R_{cb}} \mathcal{C} \Rightarrow \mathcal{A} \sqsubseteq^{\preccurlyeq}_{R_{cb} \fatsemi R_{ba}} \mathcal{C}$.*

Proposition 1 states the most important property of CPR, namely that it does indeed preserve confidentiality properties (with an appropriately adjusted point of reference). As indicated before, CPR cannot be expected to allow "secure" refinements only but it can establish that a behavior refinement whose re-abstraction admits an "abstractly secure" PCSP refinement preserves that security over data refinement and extension of adversary windows.

Proposition 1 (CPR preserves $\mathcal{CP}_r$). *Let $\mathcal{CP}_r$ be a refined basic confidentiality property with point of reference (W_r, R_{ar}). Let $\preccurlyeq$ be an information flow refinement property for $\mathcal{CP}_r$. If $\mathcal{A} \sqsubseteq^{\preccurlyeq}_{R_{ca}} \mathcal{C}$ then the following implication holds:*

$$
\begin{aligned}
&\big(\exists\, Q_a : P_a^{\top} \bullet \forall\, E_a : \mathcal{E}^{EI_a,MI_a}_{P_a,k}(H_a,A_a) \bullet \\
&\qquad ((P_c \Downarrow\!\!\rightarrow_{MI_c} (H_c \,|[EI_c]|\, A_c)) \setminus (W_c - W_a))[\![R_{ca}]\!]_D \sqsubseteq Q_a \Downarrow\!\!\rightarrow_{MI_a} E_a \\
&\qquad\qquad \wedge\, \mathcal{CP}_r(Q_a \Downarrow\!\!\rightarrow_{MI_a} E_a, W_a, W_r, R_{ar}, k)\big) \\
&\qquad\qquad\qquad \Rightarrow \\
&\big(\exists\, Q_c : P_c^{\top} \bullet \forall\, E_c : \mathcal{E}^{EI_c,MI_c}_{P_c,k}(H_c,A_c) \bullet \mathcal{CP}_r(Q_c \Downarrow\!\!\rightarrow_{MI_c} E_c, W_c, W_r, R_{ca} \fatsemi R_{ar}, k)\big)
\end{aligned}
$$

Remark 2. Carefully analyzing the constellation of the quantifiers in Proposition 1 suggests that the definition of information flow refinement might be too strong. For confidentiality preservation, it suffices indeed to require alternating universal and existential quantifiers like $\forall\, Q_a\, \exists\, Q_c\, \forall\, E_c\, \exists\, E_a \bullet \ldots$ in Definition 4. Unfortunately, the resulting definition of CPR is not transitive, because the required witnesses for the variant of the intermediate adversary model need not match.

6 Probabilistic Confidentiality: Ensured Entropy

This section serves to illustrate an instantiation of the framework discussed in the preceeding sections. It presents the probabilistic confidentiality property of *ensured entropy* [28]. Space limitations[4] do not permit to show the definitions in full detail or discuss the use of this property by way of an application example.

As mentioned in Section 4, classes $J_W^{P;k}(o)$ of indistinguishable traces are our starting point for defining confidentiality properties. Each variant of an adversary model induces a system of indistinguishability classes. Turning this fact into a requirement, one can propose a *mask* $\mathcal{M}$ that is a system of sets of traces such that the traces in each $M \in \mathcal{M}$ are indistinguishable. A variant *QE* (possibilistically) secures a mask $\mathcal{M}$ if its set $\mathcal{I}$ of indistinguishability classes *covers* $\mathcal{M}$, written $\mathcal{I} \Supset \mathcal{M}$:

$$\mathcal{I} \Supset \mathcal{M} \Leftrightarrow \forall M : \mathcal{M};\ I : \mathcal{I} \bullet M \cap I = \emptyset \vee M \subseteq I$$

If $\mathcal{I} \Supset \mathcal{M}$, the variant *QE* can produce all members of each $M \in \mathcal{M}$. Upon observing o at W, an adversary cannot conclude which member of $M \subseteq J_W^{QE,k}(o)$ caused that observation.

Ensured entropy extends this idea and requires that the indistinguishability classes of *QE* not only cover $\mathcal{M}$ but also that their entropy[5] (given the respective observation) exeeds a lower bound associated to the members of $\mathcal{M}$. As the entropy is a measure of uncertainty, this requirement puts a lower bound on the adversary's effort to infer the trace that caused an observation from that observation. Ensured entropy is weaker than, e.g., probabilistic noninterference [9] because it does not strictly prevent information flow from the machine to the adversary.

Given a mask $\mathcal{M}$ and a mapping $\mathcal{H} : \mathcal{M} \to \mathbb{R}^+$, the corresponding basic confidentiality property $\mathcal{CP}_{\mathcal{M}}(QE, W, k)$ is defined by

$$\begin{aligned}\mathcal{CP}_{\mathcal{M}}(QE, W, k) \Leftrightarrow{}& \{o : \mathrm{Obs}_W(QE) \bullet J_W^{QE,k}(o)\} \Supset (\mathcal{M} \downarrow k)\\ &\wedge \forall M : \mathcal{M} \bullet \forall o : \mathrm{Obs}_W(QE) \mid M \subseteq J_W^{QE,k}(o) \bullet \mathcal{H}(M) \le H_W^k(QE|o)\end{aligned}$$

where $H_W^k(QE|o)$ is the entropy of the process *QE* given the observation o, i.e., the entropy of $J_W^{QE,k}(o)$ given o. $\mathcal{CP}_{\mathcal{M}}(QE, W, k)$ is well-defined because *QE* is a probabilistic linear process and, therefore, the probabilities of traces of *QE* can be determined. (We spare the reader the technical definition of the refined version of $\mathcal{CP}_{\mathcal{M}}(QE, W, k)$.)

Information Flow Refinement. The information flow refinement relation that preserves $\mathcal{CP}_{\mathcal{M}}$ is defined in terms of the conditional mutual information between the behavior of the specification variant QE_a and the observations of the realization variant QE_c given an observation o_a of the specification. Using the identifiers of processes and adversary windows to denote random variables for the respective processes and

[4] A companion paper presenting the details of what is sketched here is in preparation.

[5] For an explanation of entropy, mutual information and the other concepts of information theory, refer to any book on coding theory, such as [17].

their observations, the information flow refinement relation $\preccurlyeq^k_{R_{ca}}$ for $\mathcal{CP}_{\mathcal{M}}(QE, W, k)$ is defined by:

$$\begin{aligned}(QE_a, W_a) \preccurlyeq^k_{R_{ca}} (QE_c, W_c) \Leftrightarrow \big(& QE_c \setminus (W_c - W_a)[\![R_{ca}]\!]_D = QE_a \\ & \wedge I(QE_a;\ (QE_c, R_{ca}, W_c) \mid W_a = o_a) = 0\big)\end{aligned}$$

The mutual information[6] $I(QE_a;\ (QE_c, R_{ca}, W_c) \mid W_a = o_a)$ describes the difference of the entropy of the traces of QE_a producing the observation o_a on W_a, and of the entropy of the observations on W_c produced by traces of QE_c whose re-abstractions produce the abstract observation o_a.

It is relatively straightforward to show that this information flow refinement relation preserves $\mathcal{CP}_{\mathcal{M}}(QE, W, k)$, i.e., the entropy of re-abstracted indistinguishability classes is equal to the one of the corresponding specification classes.

The transitivity proof of the relation, however, is quite involved. It needs to use several independence relationships between observations and process behaviors at different levels of refinement.

Having established those lemmas, however, the framework of CPR delivers a theory of "refinement preserving the entropy of indistinguishable system behavior".

7 Related Work

The work presented here extends previous work [10, 29] on CPR. The general idea of an adversary window to model possible observations is already present there. To consider indistinguishability classes as the basis for definitions of confidentiality properties also is not new. Zakinthinos and Lee [31] call indistinguishability classes *low level equivalence sets* (LLES). They give a definition of a (possibilistic) security property as one that can be recast as a property holding for each indistinguishability class and show that several information flow properties can be defined as properties of those classes.

The definition of CPR in [10, 29] is a quite restrictive variant of ensured entropy: It basically requires the entropy of all indistinguishability classes of the realization to be maximal, but it does not relate the probabilistic properties of the specification and the realization. Furthermore, that early definition of CPR assumed that scheduling takes place at the "meta-level" and did not make the task of the environment as a scheduler explicit. Finally, it assumed a probabilistic extension of CSP but did not explicitly base on PCSP.

The relationship to other propositions of secure refinement [14, 18, 24] has been discussed in Section 1. To the best of our knowledge, Graham-Cumming [8] still is one of the few to address security issues in data refinement. But he does not consider I/O refinement as we do.

Lowe [16] also uses the idea of quantifying over the possible refinements of a specification similar to our variants of an adversary model. Furthermore, he quantifies information flow discretely, without referring to probabilities, and thus his work mediates between a possibilistic yes/no concept of information flow and one based on probabilistic information theory. In contrast to our view, he uses a pessimistic approximation and

[6] The exact definition is quite technical and cannot be presented here.

considers the worst case, i.e., maximal, flow of information produced by all variants. Our view is pessimistic on the environment but considers an optimistic view on the machine, because the implementors control the way machine nondeterminism is resolved.

The framework as it stands now has some strong similarities with the system model underlying reactive simulatability [23, 3], which addresses the cryptographically secure implementation of "ideal" cryptographic protocols by "real" ones using cryptographic algorithms [2]. Both system models explicitly distinguish the machine, the honest users, and the adversary, all of which can interact through designated communication channels. Like our adversary window, "forbidden" channels model means of the adversary to which honest users do not have access. The differences between the two approaches stem from the different purposes they are designed to serve: We aim at a stepwise development of an "ideal" system starting from a very abstract initial specification and ending at an implementation model that still abstracts from issues of computational complexity. The model of Backes, Pfitzmann and Waidner, in contrast, is designed to support the last transition from such an "ideal" implementation model to one that uses "real" algorithms. Therefore, it is asynchronous and deterministic. It has a very detailed step semantics that allows one to analyze computation and communication acts in a very detailed manner (including their computational complexity). The concept of reactive simulatability is used to compare an ideal with a real model. It essentially is a strong (probabilistic) bisimulation [30] that enforces cryptographic indistinguishability (not to be confused with our notion of indistinguishability) of the honest users' view of the system, while the adversary can change in the transition from "ideal" to "real". Thus, reactive simulatability is very well suited to analyze the question whether an implementation of a cryptographic protocol is correct. However, it is too restrictive to support stepwise refinement from a very abstract to a much more detailed system model. In particular, it insists that the user model is the same for both, ideal and real system. This also implies that trading functionality between the machine and its environment does not establish a valid simulation.

8 Conclusion

Our framework for CPR captures general conditions sufficient to preserve probabilistic confidentiality properties in behavior refinements of nondeterministic probabilistic systems. It takes the refinement paradox into account by considering existential confidentiality properties. Definitions 4 and 5 provide an abstraction that separates the issues of preserving a probabilistic (basic) property from the ones of preserving an existential one. Thus, it allows to investigate the relationship of different properties within the same framework.

The central definitions only rely on the fact that PCSP refinement is a compositional preorder, and that hiding may introduce nondeterminism. Any formalism coming with such a refinement order could replace PCSP in the framework.

The framework establishes the essential property of a refinement relation, namely that it is a preorder. Research on conditions of compositionality of CPR is still going on. However, because CPR is more liberal than, e.g., simulatability, a result as strong as the one for that relation [5] cannot be expected without additional side conditions.

To identify conditions of compositionality is one task of ongoing research, as is the representation of standard confidentiality properties such as probabilistic [9] noninterference. Furthermore, tool support is a very important issue. Here, one can build on established tools for standard CSP and combine those with verifiers for probabilistic calculi.

Finally, confidentiality is not the only property that standard behavior refinement does not preserve. Many properties such as real-time constraints and quality of service behave similarly under refinement. Therefore, there is hope to apply the present results to other application areas as well.

Acknowledgments. Thanks go to anonymous reviewers for constructive and informed comments.

References

[1] J.-R. Abrial. *The B-Book: Assigning programs to meanings*. Cambridge University Press, 1996.

[2] M. Backes, B. Pfitzmann, and M. Waidner. A composable cryptographic library with nested operations (extended abstract). In *Proc. 10th ACM Conference on Computer and Communications Security*, pages 220–230, 2003.

[3] M. Backes, B. Pfitzmann, and M. Waidner. Secure asynchronous reactive systems. IACR ePrint Archive, March 2004. Online available at `http://eprint.iacr.org/2004/082.ps`.

[4] P. Behm, P. Benoit, A. Faivre, and J.-M. Meynadier. Météor: A successful application of B in a large project. In J. Wing, J. Woodcock, and J. Davies, editors, *FM'99 – Formal Methods*, volume I of *LNCS 1708*, pages 369–387. Springer-Verlag, 1999.

[5] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. 42nd IEEE Symposium on Foundations of Computer Science*, pages 136–145, 2001.

[6] F. Ciesinski and M. Größer. On probabilistic computation tree logic. In C. Baier, B. R. Haverkort, H. Hermanns, J.-P. Katoen, and M. Siegle, editors, *Validation of Stochastic Systems: A Guide to Current Research*, LNCS 2925, pages 147 – 188. Springer-Verlag, 2004.

[7] J. Derrick and E. Boiten. *Refinement in Z and Object-Z*. Springer-Verlag, London, 2001.

[8] J. Graham-Cumming and J. W. Sanders. On the refinement of non-interference. In *9th IEEE Computer Security Foundations Workshop*, pages 35–42. IEEE Computer Society Press, 1991.

[9] J. W. Gray, III. Toward a mathematical foundation for information flow security. *Journal of Computer Security*, pages 255–294, 1992.

[10] M. Heisel, A. Pfitzmann, and T. Santen. Confidentiality-preserving refinement. In *14th IEEE Computer Security Foundations Workshop*, pages 295–305. IEEE Computer Society Press, 2001.

[11] C. A. R. Hoare. Proof of correctness of data representations. *Acta Informatica*, 1:271–281, 1972.

[12] J. Jacob. On the derivation of secure components. In *IEEE Symposium on Security and Privacy*, pages 242–247. IEEE Press, 1989.

[13] C. B. Jones. *Systematic Software Development using VDM*. Prentice Hall, 2nd edition, 1990.

[14] J. Jürjens. Secrecy-preserving refinement. In J. N. Oliveira and P. Zave, editors, *FME 2001: Formal Methods for Increasing Software Productivity*, LNCS 2021, pages 135–152. Springer-Verlag, 2001.

[15] B. Liskov and J. Wing. A behavioral notion of subtyping. *ACM Transactions on Programming Languages and Systems*, 16(6):1811–1841, 1994.

[16] G. Lowe. Quantifying information flow. In *15th IEEE Computer Security Foundations Workshop*, pages 18–31. IEEE Computer Society, 2002.

[17] D. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[18] H. Mantel. Preserving information flow properties under refinement. In *IEEE Symposium on Security and Privacy*, pages 78–91. IEEE Computer Society Press, 2001.

[19] H. Mantel. *A Uniform Framework for the Formal Specification and Verification of Information Flow Security*. PhD thesis, Universität des Saarlandes, 2003.

[20] J. McLean. A general theory of composition for a class of "possibilistic" properties. *IEEE Transactions on Software Engineering*, 22(1):53–67, 1996.

[21] B. Meyer. Applying "design by contract". *IEEE Computer*, pages 40–51, October 1992.

[22] C. Morgan, A. McIver, K. Seidel, and J. W. Sanders. Refinement-oriented probability for CSP. *Formal Aspects of Computing*, 8(6):617–647, 1996.

[23] B. Pfitzmann and M. Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *IEEE Symposium on Security and Privacy*, pages 184–201. IEEE Computer Society, 2001.

[24] A. W. Roscoe. CSP and determinism in security modelling. In *Proc. IEEE Symposium on Security and Privacy*, pages 114–127. IEEE Computer Society Press, 1995.

[25] A. W. Roscoe. *The Theory and Practice of Concurrency*. Prentice Hall, 1998.

[26] A. W. Roscoe, J. C. P. Woodcock, and L. Wulf. Non-interference through determinism. In D. Gollmann, editor, *European Symposiom on Research in Computer Security (ESORICS)*, LNCS 875, pages 33–53. Springer-Verlag, 1994.

[27] P. Y. A. Ryan and S. A. Schneider. Process algebra and non-interference. In *12th IEEE Computer Security Foundations Workshop*, pages 214–227. IEEE Computer Society, 1999.

[28] T. Santen. Probabilistic confidentiality properties based on indistinguishability. In H. Federrath, editor, *Proc. Sicherheit 2005 – Schutz und Zuverlässigkeit*, Lecture Notes in Informatics, pages 113–124. Gesellschaft für Informatik, 2005.

[29] T. Santen, M. Heisel, and A. Pfitzmann. Confidentiality-preserving refinement is compositional – sometimes. In D. Gollmann, G. Karjoth, and M. Waidner, editors, *Computer Security – ESORICS 2002*, LNCS 2502, pages 194–211. Springer-Verlag, 2002.

[30] R. Segala and N. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

[31] A. Zakinthinos and E. S. Lee. A general theory of security properties. In *Proc. IEEE Symposium on Security and Privacy*, pages 94–102, 1997.

[32] P. Zave and M. Jackson. Four dark corners of requirements engineering. *ACM Transactions on Software Engineering and Methodology*, 6(1):1–30, 1997.