Confidentiality-Preserving Refinement is Compositional — Sometimes

Thomas Santen[1], Maritta Heisel[2], and Andreas Pfitzmann[3]

[1] Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin, Germany
email: santen@acm.org

[2] Institut für Praktische Informatik und Medieninformatik
Technische Universität Ilmenau, Germany
email: maritta.heisel@prakinf.tu-ilmenau.de

[3] Fakultät Informatik, Technische Universität Dresden, Germany
email: pfitza@inf.tu-dresden.de

Abstract. Confidentiality-preserving refinement describes a relation between a specification and an implementation that ensures that all confidentiality properties required in the specification are preserved by the implementation in a probabilistic setting. The present paper investigates the condition under which that notion of refinement is *compositional*, i.e. the condition under which refining a subsystem of a larger system yields a confidentiality-preserving refinement of the larger system. It turns out that the refinement relation is not composition in general, but the condition for compositionality can be stated in a way that builds on the analysis of subsystems thus aiding system designers in analyzing a composition.

1 Introduction

In systems and software engineering, the consent is growing that secure systems cannot be built by adding security features *ex post* to an existing implementation but that "security-aware" engineering of systems and software must take security concerns into account, starting from requirements engineering through architectural and detailed design to coding, testing, and deployment.

It is obvious that only some kind of divide-and-conquer approach makes building non-trivial systems feasible. Such an approach must support decomposing a system into subsystems, implementing those subsystems largely independently of each other, and finally composing the implementations of those subsystems to make up an implementation of the entire system. More specifically, such an approach decomposes a system *specification* into specifications of subsystems, and it composes (correct) *implementations* of subsystem specifications to yield a (correct) implementation of the system specification.

In this setting, the question arises whether correctness of the subsystem implementations with respect to their specifications is sufficient to guarantee that composing the subsystem implementations yields a correct implementation of the original specification. If this is true, then the implementation relation relating a specification to the set of its correct implementations is called *compositional*. Adapting this definition to security, the question arises whether security-property-preserving implementations of the subsystems with respect to their specifications are sufficient to guarantee that composing the subsystem implementations yields a security-property-preserving implementation of the original specification of the whole system.

When formal techniques are used for system development, both the specification and the implementation are often described in the same formalism. Then, the former is called the *abstract* specification, and the latter is called the *concrete* specification. The relation describing the correct implementations of an abstract specification is called a *refinement* relation, and a concrete specification implementing an abstract one is called a refinement of the abstract specification. For a notion of refinement, the properties of transitivity and compositionality are very important. Without these properties, the practical application of refinement is hardly possible.

In earlier work [4], we have motivated a probabilistic notion of a confidentiality-preserving refinement and sketched its formalization using an extension of CSP with probabilistic choice. Classical formal techniques, which are possibilistic, either impose sufficient conditions that are too strong or impose only necessary conditions that are too weak to realize required confidentiality properties [16].

In the present paper, we further investigate properties of confidentiality-preserving refinement, with the goal of enhancing its potential for practical applicability. For these investigations, we represent our systems using a probabilistic variant of CSP, and slightly rephrase our definition to better capture the intuition motivated in [4]. We prove that the resulting refinement relation is *transitive*. By way of a counterexample, we show that confidentiality-preserving refinement, in general, is *not compositional*. The main contribution of the paper is a necessary and sufficient condition for compositionality of confidentiality-preserving refinement, called *non-disclosure*.

A technical report [14] contains more explanatory prose and the complete proofs of all theorems and lemmas mentioned in the paper.

2 Probabilistic CSP and Behavioral Refinement

We use a probabilistic extension of the process algebra of "Communicating Sequential Processes" (CSP) to formally describe the systems we reason about. Roscoe [12] comprehensively treats classical CSP. In this section, we briefly introduce the notation and the notion of behavioral refinement on which we build confidentiality-preserving refinement in Section 3.

2.1 CSP Notation

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.

In the following, we describe the CSP notation used in this paper. In the following, P and Q are processes, $e \in \Sigma$ is an event, $X \subseteq \Sigma$ is a set of events, $S \in \Sigma \leftrightarrow \Sigma$ is a relation on events, and $R \in D \leftrightarrow D$ is a relation on data.

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: 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.

In the notion of refinement we use, we are interested in changing data representations (*data 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 D, the process $P[\![R]\!]_D$ 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 $\alpha\, P$ 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 as 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 $P(x)$. As an extension to classical CSP, we also need a *probabilistic* choice $\bigoplus_x^{\mathcal{P}} P(x)$: this process chooses x – and thus $P(x)$ – according to a probability distribution $\mathcal{P}$.

For behavioral refinements, we disregard distributions on choices and treat all probabilistic choices as nondeterministic ones: the possibilistic version $\widehat{P}$ of a process P is defined by replacing each occurrence of the probabilistic choice $\bigoplus$ by a corresponding nondeterministic choice $\bigsqcap$.

2.2 Refinement of Behavior and Data

There are several notions of refinement for CSP: trace refinement, failure refinement, and failure-divergence refinement. The latter two imply trace refinement. If P is refined by Q, denoted $P \sqsubseteq Q$, then – regardless of the refinement relation used – $\mathrm{traces}(Q) \subseteq \mathrm{traces}(P)$.

We wish to cover changes of data representations in our refinement relation. Therefore, we extend the usual CSP refinement with a *retrieve relation* mapping concrete to abstract data, and define *behavioral refinement* as a combination of CSP refinement and data renaming according to the retrieve relation.

Definition 1 (Retrieve Relation). *Let P and Q be processes over Σ. A relation $R \in D \leftrightarrow D$ between concrete and abstract data is called a* retrieve relation *from Q to P, if $\mathrm{dom}\, R \subseteq \mathrm{data}\, Q$ and $\mathrm{ran}\, R \subseteq \mathrm{data}\, P$, where $\mathrm{dom}\, R$ and $\mathrm{ran}\, R$ denote the domain and range, respectively, of relation R, and $\mathrm{data}\, P$ is the set of data occuring in events of P.*

At some places we need the set $R^{-1}(r)$ of all possible data refined versions of a trace r. Applying R^{-1} to a trace r means applying the inverse of R to the data in each event of the trace, and $R^{-1}(r)$ denotes the set of all such traces:

$R^{-1}(r) = \{t \mid \mathrm{dom}\, t = \mathrm{dom}\, r \,\land$

$$\forall\, i \in \operatorname{dom} r;\ c \in Ch;\ d \in D \bullet \exists\, d' \in R(d) \bullet t(i) = c.d \Rightarrow r(i) = c.d'\}$$

Definition 2 (Behavioral Refinement). *Let P and Q be processes over Σ. Let R be a retrieve relation from Q to P. Then Q* refines *P via R (written $P \sqsubseteq_R Q$), if $\widehat{P} \sqsubseteq \widehat{Q}[\![R]\!]_D$, where $\sqsubseteq$ is the usual refinement of CSP.*

Behavioral refinement is transitive and monotonic [14].

We will need to consider a restriction of a "concrete" process Q to a behavior implementing a given "abstract" trace r.

Definition 3. *Let Q be a process, R be a retrieve relation abstracting the data in Q, and let r be a trace over the range of R. Then $Q|_r^R$ is a process that chooses a behavior of Q whose starting sequence is compatible with r.*

$$Q|_r^R := \mathrm{Pr}(r)[\![R^{-1}]\!]_D \,|[\alpha\, Q]|\, Q$$

The process $\mathrm{Pr}(r)$ produces the trace r and behaves arbitrarily afterwards:

$$\mathrm{Pr}(\langle\,\rangle) = \mathrm{Pr}(\langle \checkmark \rangle) = RUN \qquad \mathrm{Pr}(\langle e \rangle \frown s) = e \rightarrow \mathrm{Pr}(s)$$

The event $\checkmark$ at the end of a trace signifies termination of the process. The process RUN engages in any communication the environment proposes.

Behavioral refinement is defined in terms of the possibilistic versions of the involved processes. Morgan et.al. [11] define a refinement relation for probabilistic processes, which turns out to be a very delicate task when allowing both, nondeterministic and probabilistic choices.[1] Because confidentiality-preserving refinement as defined in Section 3 imposes a condition on processes that, in particular, does not require the probabilistic behavior of a process to be preserved in a refinement, we do not use Morgan, et.al.'s definition[2] of refinement.

2.3 Probability Distributions on Processes

We conclude the brief discourse on probabilistic CSP with some properties of probability distributions, which we will need later. Given a process P, which may contain external, nondeterministic, and probabilistic choices, the set of processes $\mathrm{Prob}(P)$ contains all processes that are obtained by replacing each external choice and each nondeterministic choice in P by a probabilistic choice for some (arbitrary) distribution.

When we will consider probabilistic properties of processes later, we will argue about all members $\mathrm{Prob}(P)$ for a given P, because thus we consider all possible probabilistic behavior of the environment (external choices) and all possible probabilistic behavior of an implemented system for which the process (as a specification) does not determine the distribution (nondeterministic choices).

For a process Q that contains only probabilistic choices, we define a family of probability distributions $\mathcal{P}_n(Q, t)$ that is indexed by the maximal length n of traces it considers: $\mathcal{P}_n(Q, t)$ is a distribution on the set of traces with length n or that terminate (the last

[1] It is not easy to avoid either one when defining processes.
[2] An investigation of the relation between the two is nevertheless theoretically interesting.

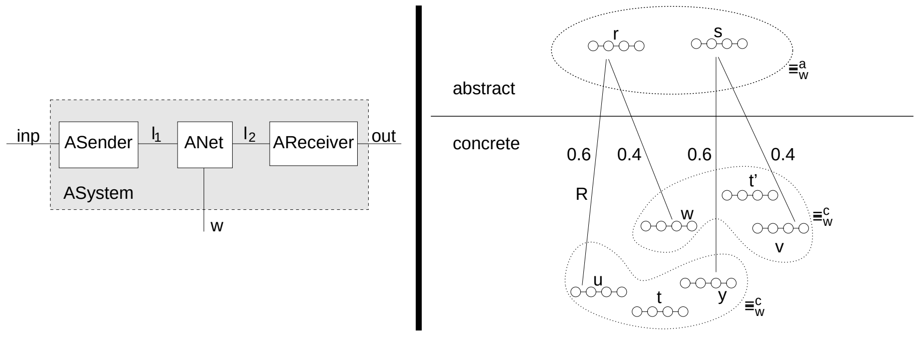

Fig. 1. System model (left), and concretization vs. indistinguishability (right)

event is ✓) and have a length less than *n*. For a given *t*, we write $\mathcal{P}_Q(t)$ for $\mathcal{P}_{\#t}(Q, t)$, where $\#t$ is the length of *t*.

The probability of *Q* producing a trace in a set *M*, which describes a certain property of *Q*, is given by

$$\mathcal{P}_Q(t \in M) = \sum_{t \in \mathrm{pfree}(M)} \mathcal{P}_Q(t) \tag{1}$$

The set $\mathrm{pfree}(M) \subseteq M$ is the maximal subset of *M* that does not contain any $t \in M$ for which there is a prefix t' of *t* in *M*.

Finally, we define $\mathcal{P}_Q(s) = 0$ for traces $s \notin \mathrm{traces}(Q)$.

3 Confidentiality-Preserving Refinement (CPR)

In this section, we present our definition of confidentiality-preserving refinement, which we have extensively motivated in an earlier publication [4]. To specify confidentiality properties we use a system model illustrated on the left-hand side of Fig. 1 for the example of a communication system between a sender and a receiver communicating over an untrusted network. We specify a system for which confidentiality is a relevant requirement by a pair of a process and a window channel.

Definition 4 (System, Window). *A* system specification $A = (Q, w)$ *is a pair of a process definition Q and a distinguished channel* $w \in \alpha\, Q$*, called the* window *of A.*

Specifying the system *ASystem* of Fig. 1, we define three processes *ASender*, *ANet*, and *AReceiver*, where *ASender* and *ANet* communicate via the internal channel l_1, and *ANet* and *AReceiver* communicate over the internal channel l_2. Then, the process *ASystem* is the parallel composition of those three processes. The window *w* is a distinguished channel of *ASystem*.

$$\begin{aligned} &\mathit{ASystem} = (Q_A, w) \\ &\qquad Q_A \mathrel{\widehat{=}} ((\mathit{ASender}\, |[l_1]|\, \mathit{ANet})\, |[l_2]|\, \mathit{AReceiver}) \setminus \{l_1, l_2\} \end{aligned}$$

$$\alpha\, Q_A = \{inp, out, w\}$$

The channel w models the flow of data from the system to an adversary. Observing the channel w, the adversary gains information about the system. Any distinction the adversary can make about the internal state of the system based on the observations on w is information that the system does not keep confidential. Conversely, the system keeps confidential any aspect of its behavior that an adversary cannot distinguish by observing w. We formally capture that confidentiality property by defining equivalences over system traces.

Definition 5 (Indistinguishability). *Let $A = (Q, w)$ be a system specification. Two traces $s, t \in \mathrm{traces}(Q)$ are* indistinguishable by *w (denoted $s \equiv_w t$) iff their projections to w are equal: $s \equiv_w t \;\Leftrightarrow\; s \upharpoonright \{w\} = t \upharpoonright \{w\}$*

In the transition from an abstract to a concrete system specification, the interpretation of a window changes. The window of an *abstract* system specifies what information is *allowed* to be visible to an adversary. The window of a *concrete* system specifies what information *cannot* be hidden from the adversary.

Here, a purely logical argument is insufficient because it is not enough to ask whether a distinction in the concrete system *definitely* allows an observer to distinguish confidential data, but we must describe whether such a distinction provides *more* information about the confidential data than the abstract window reveals. Therefore, we consider the respective probabilities of internal data that may cause a particular observable behavior on a window. The right-hand side of Fig. 1 illustrates our approach to formalizing that probabilistic argument:

Consider an abstract and a concrete system that behaviorally refines the abstract one with retrieve relation R. Let r and s be two abstract traces that are indistinguishable with respect to the window w, i.e. $r \equiv_w^a s$. According to the retrieve relation R, trace r can be represented by the concrete traces u and w, and trace s can be represented by the concrete traces v and y, where u and y as well as v and w are indistinguishable by observing the concrete window, i.e. $u \equiv_w^c y$ and $w \equiv_w^c v$. For keeping r and s indistinguishable in the concrete system, we must require that the probability that r is represented by u be the same as the probability that s is represented by y. If this were not the case, an adversary might be able to gain information whether r or s happened on the abstract layer: if the probability that r is represented by u is greater than the probability that s is represented by y, for an adversary, the observation of some element $t \equiv_w^c y$ increases the probability of r with respect to s.

Definition 6 reflects this argument: A confidentiality-preserving refinement is one that (1) is the behavioral refinement of the processes describing a system (c.f. Definition 2), and that (2) (probabilistically) preserves the indistinguishability of system traces. In the latter condition, we consider the behavior of the concrete system implementing an abstract trace r, i.e. the process $Q|_r^R$. This process may contain external choices stemming from the different implementation choices that R^{-1} assigns to the data in r. Because there is no way of knowing with what probability those implementation choices are resolved, we need to consider all possible distributions that make $Q|_r^R$ a probabilistic process, i.e. all members of $\mathrm{Prob}(Q|_r^R)$.

Remark 1. To keep the language simple, we will talk about a probabilistic property E of a process Q, when we mean that all members $Q_p \in \mathrm{Prob}(Q)$ satisfy E.

Definition 6 (Confidentiality-Preserving Refinement, CPR). *Let $A = (P, w)$ and $C = (Q, w)$ be two system specifications. Let $\equiv^a_w$ be the indistinguishability in A (wrt. w), and let $\equiv^c_w$ be the indistinguishability in C (wrt. w). The system C is a* confidentiality-preserving refinement (CPR) *of the system A via the retrieve relation R from Q to P ($A \sqsubseteq^{cpr}_R C$) iff:*

1. *$P \setminus \{w\} \sqsubseteq_R Q \setminus \{w\}$, and* — *behavioral refinement, BR*
2. *$\forall\, r, s \in \mathrm{traces}(P);\ t \in \mathrm{traces}(Q);$* — *indistinguishability preservation, IP*
   $Q_r \in \mathrm{Prob}(Q|^R_r);\ Q_s \in \mathrm{Prob}(Q|^R_s) \bullet$
   $r \equiv^a_w s \Rightarrow \mathcal{P}_{Q_r}(u \equiv^c_w t) = \mathcal{P}_{Q_s}(v \equiv^c_w t)$

We write $P \sqsubseteq^{cpr}_{R,w} Q$ for $(P, w) \sqsubseteq^{cpr}_R (Q, w)$, which is useful when analyzing systems with respect to different windows. Although the windows of the two systems have the same name w, they may carry different data because the processes of the systems determine the data that is transmitted on a channel.

Hiding w from P and Q in Condition *BR*, Definition 6 does not require Q to refine the window w: At the implementation level, an adversary may have means of observation that are in no way related to the means of observation given at the specification level. Requiring Q to refine the window w therefore would – inadequately – allow the specification to impose restrictions on the power of an adversary at the level of implementation. Condition *IP* captures the important restriction on confidentiality-preserving implementations that adversaries must not be able to infer more information by observing the implementation than the window at the specification level allows them to. It states that, given two indistinguishable abstract traces r and s, and a concrete trace t, the probability of choosing a concrete trace u which is indistinguishable of t as an implementation of r must be the same as choosing a concrete trace v which is indistinguishable of t as an implementation of s, see Fig. 1.

To determine the probability $\mathcal{P}_{Q_r}(u \equiv^c_w t)$, we consider a subset of the set $T = \{u \mid u \in \mathrm{traces}(Q_r) \wedge u \equiv^c_w t\}$ of all traces that are indistinguishable from t. Because T need not be prefix-free, we must consider the set $\mathrm{pfree}(T)$ for calculating probabilities. Then, it holds:

$$\mathcal{P}_{Q_r}(u \equiv^c_w t) = \sum_{u \in \mathrm{pfree}(T)} \mathcal{P}_{Q_r}(u)$$

To support stepwise refinement and the independent refinement of subsystems, a refinement relation must have two properties: it must be transitive and compositional. The following Theorem 1 establishes the transitivity of CPR. Section 4 extensively discusses compositionality.

Theorem 1 (Transitivity of CPR). *Let $A = (P_a, w)$, $B = (P_b, w)$, and $C = (P_c, w)$ be system specifications. Let R_{ba} and R_{cb} be retrieve relations from P_b to P_a, and from P_c to P_b, respectively. Then $A \sqsubseteq^{cpr}_{R_{ba}} B \wedge B \sqsubseteq^{cpr}_{R_{cb}} C \Rightarrow A \sqsubseteq^{cpr}_{R_{cb} \fatsemi R_{ba}} C$, where $\fatsemi$ is the forward composition of relations.*

4 Compositionality of CPR

Compositionality of security properties and compositionality of refinement are two different notions. Section 4.1 contrasts the two. Indistinguishability is not a compositional property, and CPR is, in general, not compositional. Section 4.2 illustrates this fact by way of a counterexample. For a composed system for which refinements of subsystems are known, we can find a condition that characterizes the circumstances under which a CPR is compositional. This condition is more intuitive than the Condition *IP* of Definition 6, and it allows one to build on the analyses made for verifying the CPR of subsystems. Section 4.3 establishes this result in Theorem 2.

4.1 Compositionality of Security Properties vs. Compositionality of Refinement

Compositionality of a security property, as it is often considered in the context of secure systems, means the preservation of that property under composition of systems: if certain systems satisfy a certain security property, some variant of non-interference, say, then the composition of those systems satisfies the same property. Mantel [10] investigates the relation between compositionality results for many known information flow properties.

Compositionality of a refinement relation, in which we are interested in this paper, addresses the interplay of decomposing a system specification, and composing the implementations of the subsystems to yield an implementation of the original system specification. Thus it is a preservation property that relates different levels of abstraction (specification – implementation), whereas the compositionality of security properties is concerned with one level of abstraction only. As we will see in the following section, the security property indistinguishability (c.f. Definition 5) is not compositional. Since this is true for the abstract as well as the concrete level of a refinement, the non-compositionality of indistinguishability, in principle, *weakens* the requirements for a refinement to be compositional. As a consequence, embedding a system into a context but refining it in isolation leads to two questions to be answered:

1. Does the composed system still fulfill the desired security requirements?
2. Does the replacement of the abstract by the concrete system in the given context compromise security?

Question 1 means that the *composition* has to be considered and possibly be rejected, independently of later refinements. The present paper does not address this problem. Since the usual notion of correctness of refinement does provide for preservation of integrity and hopefully availability, but not at all for confidentiality, we narrow Question 2 to: Does the replacement of the abstract by the concrete system in the given context compromise confidentiality? This amounts to showing that the refinement is compositional for the given context. In the rest of the paper, we will show how to answer this question.

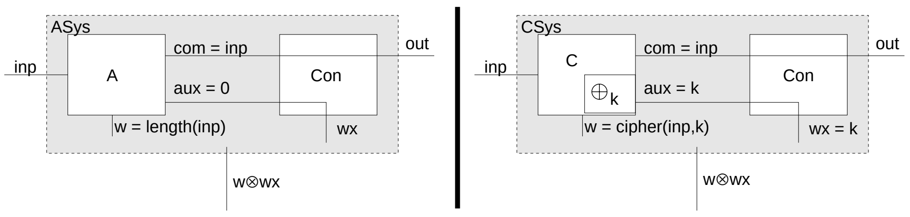

Fig. 2. Confidentiality-preserving refinement is not compositional

4.2 A Counterexample

The two systems shown in Fig. 2 illustrate that, in general, CPR is *not* compositional. The left-hand side of the figure shows an abstract system *ASys* with two communication channels *inp* and *out*, and a window $w \otimes wx$ that is a combination of the windows *w* and *wx* of the two subsystems *A* and *Con*: all data observable on *w* or *wx* are also observable on $w \otimes wx$. The subsystem *A* specifies a secure communication service. It allows its environment to observe the length of messages transmitted from channel *inp* to channel *com*, but no other information about the content of messages. The subsystem *Con* specifies the context in which *A* operates. The context *Con* copies the data that *A* produces on *com* to its output channel *out*. The systems *A* and *Con* also communicate via the channel *aux*: *Con* receives data from *A* that do not contain any relevant information (represented by a constant 0). The window *wx* of *Con* allows data received on *aux* to be observed by the environment.

The right-hand side of Fig. 2 shows an implementation *CSys* of *ASys*. The subsystem *C* is an implementation of the communication service that *A* specifies in the presence of an untrusted network. The implementation *C* probabilistically chooses keys *k* with equal probabilities and uses a (suitable) encryption function *cipher* to conceal the transmitted data from an observer who can intercept communication on the network: the window *w* of *C* allows an adversary to observe the ciphertext $cipher(inp, k)$.

Because observing that ciphertext will reveal the length of the message but nothing else about its content, the system *C* is a CPR of the system *A*, as shown in [4]. If CPR was unconditionally compositional, we would expect *CSys*, which is obtained from *ASys* by substituting *C* for *A*, to be a CPR of *ASys*. This, however, is not true: *C* not only implements *A* correctly[3], but it also transmits the selected key *k* over the channel *aux*. This does not compromise confidentiality if *C* is considered in isolation, because *w* does not make information about the data on *aux* available to the adversary. Composing *C* with *Con* as in *CSys*, however, allows the adversary to observe the key *k* on the *new* window *wx*! Thus, the combination of the information gained by observing *w* and *wx* reveals the original input message to an adversary, which is not revealed on the abstract level.

[3] The retrieve relation maps each key transmitted on channel *aux* to the constant 0, which is an admissible data refinement.

In this example, the non-compositionality of CPR is a direct consequence of the non-compositionality of indistinguishability: composing the windows w and wx in $CSys$ makes *more* observations possible (the key becomes observable) and thus an observer can distinguish more behavior of the subsystems than by observing their respective windows alone.

The same argument, however, is also true for the abstract level: the indistinguishability *requirement* on the implementation will, in general, become weaker by combining windows, thus strengthening the premise of Condition *IP* in Definition 6, and allowing for *more* confidentiality-preserving refinements. Additionally, much more subtle effects relating to the probabilistic nature of Definition 6 may compromise the compositionality of CPR.

4.3 A Condition for Compositionality

After the somewhat discouraging result of the previous section, we will now investigate the conditions under which CPR is nevertheless compositional. We will make precise the intuition gained from analyzing the counterexample, and come up with a condition for compositionality that reduces the question of compositionality to the question what *additional* distinctions a new window on the subsystem allows an adversary to make.

Formally, we consider two systems $A = (P, w)$ and $C = (Q, w)$ with $A \sqsubseteq_R^{cpr} C$ for some retrieve relation R from Q to P. The context $Con = (Cx, wx)$ is another system that can communicate with A and C via a set of channels K. The context window is different[4] from the window on A and C: $wx \neq w$.

Combining the system A with the context Con yields the system $(P \,|[K]|\, Cx, w \otimes wx)$. We assume here that the processes P and Cx work on the same set of abstract data. Therefore, to combine C with Con, we must "concretize" the data in Cx by a data renaming consistent with the retrieve relation from Q to P: $(Q \,|[K]|\, Cx[\![R^{-1}]\!]_D, w \otimes wx)$. Viewed abstractly, the process $Cx[\![R^{-1}]\!]_D$ has the same behavior as Cx, but for each data item a that Cx transmits, the process $Cx[\![R^{-1}]\!]_D$ transmits a data item b that implements a, i.e. for which $b \; R \; a$ holds. In this setting, compositionality means that C combined with Con refines A combined with Con:

$$(P \,|[K]|\, Cx, w \otimes wx) \sqsubseteq_R^{cpr} (Q \,|[K]|\, Cx[\![R^{-1}]\!]_D, w \otimes wx) \tag{2}$$

If Condition *IP* of Definition 6 does not hold for (2), then the additional observations of the concrete system that the window wx permits via the context must allow an adversary to distinguish more behavior of P than the window w permits. Composing the context with the processes P or Q, respectively, forces them to synchronize with the context and thus reduces their possible behavior. This may change the probabilities of traces of P and Q, which might also affect Condition *IP*.

This analysis motivates the three essential tasks to solve for stating our compositionality theorem:

1. to reduce the combination of a system with a context, which itself has an additional window, to adding a window to the system;

[4] Although the channel names are different, the same data can, of course, be transmitted over those channels.

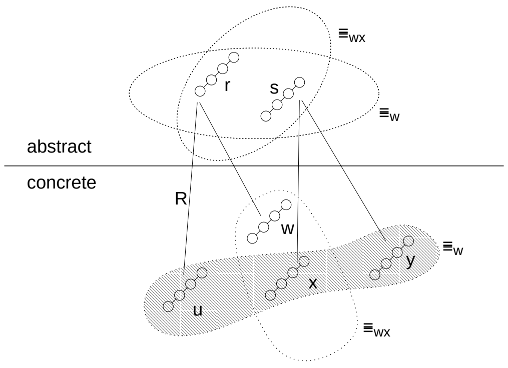

Fig. 3. Non-Disclosure

2. to come up with a condition describing the circumstances under which a CPR between two systems is preserved under addition of a window to both systems;
3. to show that reducing behavior by adding a context preserves CPR.

To solve the first task, we wish to consider the context as a means of an adversary to observe the behavior of P or Q at the channels K. Technically, we can achieve this by hiding all channels of Cx but wx and but the ones in K from the composed systems. This leads to the notion of a system in context:

Definition 7 (System in Context). *Let $A = (P, w)$ and $Con = (Cx, wx)$ be system specifications. Let $K \subseteq (\alpha P \cap \alpha Cx) - \{w, wx\}$ be a set of channels over which A and Con can communicate, and let $X := \alpha Cx - (\alpha P \cup \{wx\})$. Then A* in context *Con, written $A \overset{K}{\triangleleft} Con$, is the system $(P \,|[K]|\, (Cx \setminus X), w \otimes wx)$.*

The system $A \overset{K}{\triangleleft} Con$ is the system A with an additional window wx and the reduced behavior that is a consequence of synchronizing with Con. Similarly, at the concrete level, the system $C \overset{K}{\triangleleft} Con[\![R^{-1}]\!]_D$ is the system C with an additional window and reduced behavior. Lemma 1 shows that it is sufficient to prove a CPR between those systems in context to establish (2).

Lemma 1. *Let $A = (P, w)$, $C = (Q, w)$, and $Con = (Cx, wx)$ be system specifications. Let $w \in (\alpha P \cap \alpha Q) - \alpha Cx$, $wx \in \alpha Cx$, $K \subseteq (\alpha P \cap \alpha Q \cap \alpha Cx) - \{w, wx\}$, and $X := \alpha Cx - (\alpha P \cup \alpha Q \cup \{wx\})$. Let R be a retrieve relation from Q to P. Let $(\widetilde{P}, w \otimes wx) := A \overset{K}{\triangleleft} Con$ and $(\widetilde{Q}, w \otimes wx) := C \overset{K}{\triangleleft} Con[\![R^{-1}]\!]_D$. Then*

$$P \sqsubseteq^{cpr}_{R,w} Q \wedge \widetilde{P} \sqsubseteq^{cpr}_{R,w\otimes wx} \widetilde{Q} \;\Rightarrow\; P \,|[K]|\, Cx \sqsubseteq^{cpr}_{R,w\otimes wx} Q \,|[K]|\, Cx[\![R^{-1}]\!]_D$$

To solve the second task, consider the effect of adding a window wx to the abstract and concrete systems. By the following Equivalence (3), going from w to $w \otimes wx$ makes the equivalence classes of the indistinguishability finer.

$$s \equiv_{w\otimes wx} t \Leftrightarrow s \equiv_w t \wedge s \equiv_{wx} t \tag{3}$$

As Fig. 3 illustrates, the equivalence classes at both, the abstract and the concrete level, become finer. Thus, we need to consider *fewer* pairs (r, s) of abstract traces in Condition *IP* of Definition 6, which weakens the condition, but at the same time, we need to show a *stronger* property for the pairs (r, s) that we still must consider: For all traces t of the concrete system, the probability of the concrete system to choose a behavior u implementing r that is indistinguishable from t by both w and wx must be the same as the probability of the system to choose an implementation v of s that is indistinguishable from t by both w and wx. Formally, we need to show:

$$\sum\nolimits_{u \,\in\, \mathrm{pfree}\left([t]_{\equiv^c_w} \cap [t]_{\equiv^c_{wx}}\right)} \mathcal{P}_{Q_r}(u) = \sum\nolimits_{v \,\in\, \mathrm{pfree}\left([t]_{\equiv^c_w} \cap [t]_{\equiv^c_{wx}}\right)} \mathcal{P}_{Q_s}(v) \qquad (4)$$

We already know $[t]_{\equiv^c_w}$ from proving $A \sqsubseteq^{cpr}_R C$. Proving Equation (4) as it stands would mean to analyze which behavior of C that is indistinguishable by w remains indistinguishable when adding wx. From a practical point of view, however, it is more suitable to analyze which observations made using wx allow an adversary to *distinguish* behavior that is indistinguishable by w, and to compare probabilities for that behavior. This means to let the sums range over the set difference of the respective equivalence classes. If the resulting equation of probabilities holds, we call wx *non-disclosing* on the system with respect to r and s.

Definition 8 (Non-Disclosure). *Let $S = (Q, w)$ be a system specification, and let $wx \in \alpha\, Q - \{w\}$ be a channel of Q that is distinct from w. Let R be a retrieve relation from Q to the data in two traces r and s. We call wx* non-disclosing *on S wrt. R, r and s, written $Q|^R_{r,s} \vdash w \mathrel{\mathring{=}} wx$, iff the following condition holds:*

$$\begin{aligned}&\forall\, t : \mathrm{traces}(Q);\ Q_r \in \mathrm{Prob}(Q|^R_r);\ Q_s \in \mathrm{Prob}(Q|^R_s) \bullet \\ &\qquad\qquad \sum\nolimits_{u \in \left(T(Q_r,t)\ -\ [t]_{\equiv_{wx}}\right)} \mathcal{P}_{Q_r}(u) \ = \ \sum\nolimits_{v \in \left(T(Q_s,t)\ -\ [t]_{\equiv_{wx}}\right)} \mathcal{P}_{Q_s}(v)\end{aligned}$$

The set of traces $T(Q,t)$ is given by $T(Q,t) := T_0(Q,t) \cup \mathrm{pfree}(T_1(Q,t))$, where

$$\begin{aligned}T_0(Q,t) = \{u \in \mathrm{traces}(Q) \mid u \equiv_w t \wedge \#(u \upharpoonright \{wx\}) = \#(t \upharpoonright \{wx\}) \wedge \\ \neg\, (\exists\, u' \in \mathrm{traces}(Q) \bullet u' \text{ prefix } u \wedge \\ u' \equiv_w t \wedge \#(u' \upharpoonright \{wx\}) = \#(t \upharpoonright \{wx\}))\}\end{aligned} \qquad (5)$$

$$\begin{aligned}T_1(Q,t) = \{u \in \mathrm{traces}(Q) \mid u \equiv_w t \wedge u \notin T_0(Q,t) \wedge \\ \neg\, (\exists\, u' \in T_0(Q,t) \bullet u \text{ prefix } u')\}\end{aligned} \qquad (6)$$

The prefix-free set $T(Q,t)$ is an alternative for $\mathrm{pfree}([t]_{\equiv_w})$ when computing the probability that Q produces a trace u with $u \equiv_w t$. Additionally, the (rather technical) construction ensures that $T(Q,t)$ contains a maximal number of traces that have as many observations over the other window wx as t has.

Lemma 2 states that – given $P \sqsubseteq^{cpr}_{R,w} Q$ – non-disclosure characterizes the circumstances under which CPR is preserved when a new window is added.

Lemma 2. *Let P and Q be processes, $w, wx \in \alpha\, P \cap \alpha\, Q$ be channels common to P and Q, and let R be a retrieve relation from Q to P. If $P \sqsubseteq^{cpr}_{R,w} Q$ then*

$$P \sqsubseteq^{cpr}_{R,w\otimes wx} Q \Leftrightarrow \left(\forall\, r, s \in \mathrm{traces}(P) \bullet r \equiv^a_{w\otimes wx} s \Rightarrow Q|^R_{r,s} \vdash w \mathrel{\mathring{=}} wx\right)$$

To solve the third task, we need Lemma 3 that relates a given CPR to a CPR of the same systems in a context that does *not* add a new window. When we apply this lemma to prove compositionality, we will regard *wx* not as a window but as an ordinary channel of *Cx*.

Lemma 3. *Let $A = (P, w)$, $C = (Q, w)$, and $Con = (Cx, wx)$ be system specifications with $w \neq wx$. Let $K \subseteq (\alpha\, P \cap \alpha\, Q \cap \alpha\, Cx) - \{w, wx\}$ be a set of channels over which A and Con, and C and Con, respectively, can communicate. Let $(\widetilde{P}, w \otimes wx) := A \overset{K}{\lhd} Con$ and $(\widetilde{Q}, w \otimes wx) := C \overset{K}{\lhd} Con[\![R^{-1}]\!]_D$. Then $P \sqsubseteq^{cpr}_{R,w} Q \Rightarrow \widetilde{P} \sqsubseteq^{cpr}_{R,w} \widetilde{Q}$.*

Lemmas 1, 2, and 3 allow us to prove the main result of this paper: CPR is compositional if the context window *wx* is non-disclosing on the refined subsystem.

Theorem 2 (Compositionality of CPR). *Let $A = (P, w)$, $C = (Q, w)$, and $Con = (Cx, wx)$ be system specifications with $w \neq wx$. Let $K \subseteq (\alpha\, P \cap \alpha\, Q \cap \alpha\, Cx) - \{w, wx\}$ be a set of channels over which A and Con or C and Con, respectively, can communicate. Let R be a retrieve relation from Q to P. Let $\widetilde{P}$ be the process of $A \overset{K}{\lhd} Con$, and let $\widetilde{Q}$ be the process of $C \overset{K}{\lhd} Con[\![R^{-1}]\!]_D$. If*

1. *$A \sqsubseteq^{cpr}_{R} C$, and*
2. *$\forall \widetilde{r}, \widetilde{s} : \mathrm{traces}(\widetilde{P}) \bullet \widetilde{r} \equiv^{a}_{w \otimes wx} \widetilde{s} \Rightarrow \widetilde{Q}|^{R}_{\widetilde{r},\widetilde{s}} \vdash w \mathrel{\mathring{=}} wx$*

then $(P\,|[K]|\,Cx, w \otimes wx) \sqsubseteq^{cpr}_{R} (Q\,|[K]|\,Cx[\![R^{-1}]\!]_D, w \otimes wx)$.

Proof. Assume $P \sqsubseteq^{cpr}_{R,w} Q$. With Lemma 3, we get $\widetilde{P} \sqsubseteq^{cpr}_{R,w} \widetilde{Q}$, which implies $\widetilde{P} \sqsubseteq^{cpr}_{R,w\otimes wx} \widetilde{Q}$ by Assumption 2 and Lemma 2. From Assumption 1 and Lemma 1, we conclude $P\,|[K]|\,Cx \sqsubseteq^{cpr}_{R,w\otimes wx} Q\,|[K]|\,Cx[\![R^{-1}]\!]_D$. □

5 Related Work

Because indistinguishability and its preservation by refinement (CPR) is concerned with hiding certain information about events occurring in a system from an adversary, our work is related to research on non-interference.

Non-interference, first introduced by Goguen and Meseguer [1], is a security property that has extensively been studied. Much work on non-interference is possibilistic, i.e. it disregards probabilistic arguments. Ryan and Schneider [13] recast many known definitions of possibilistic non-interference in (classical) CSP and show that the different ways of defining non-interference are closely related to the different notions of process equivalence.

The windows in our setting can be viewed as a channel from the considered system to an outside adversary, i.e. from the "high" system to the "low" outside world. In contrast to non-interference, we do *not* require no information to flow through that channel. Our definition of CPR ensures that possible observations which adversaries may make of the implemented system do not offer them additional ways of inferring information about the system than the specification allows them to.

Ryan and Schneider [13] discuss an approach of generalizing non-interference that has a similar motivation: requiring total absence of information flow often is too strong in practice. Their generalization is parameterized by an equivalence on traces, an equivalence on processes, and a way of abstracting High's behavior from a process. Depending on the instantiation of these parameters one obtains weak versions of non-interference that allow Low to determine High's behavior up to the equivalence on traces. It may be interesting to recast our definition of indistinguishability into that framework.

Gray [3] defines probabilistic non-interference (PNI), and Jürjens [6] proves a compositionality theorem for a variant of that definition. The condition for PNI basically states that the probability of a high user producing a particular observation of a low user is the same for all behaviors of the high user. Because it formalizes the fact that certain behaviors cannot be distinguished probabilistically, this condition of PNI is similar to our refinement condition *IP*. The difference is that *IP* requires equal probabilities of (indistinguishable) concrete behavior only for implementations of indistinguishable abstract behavior.

Lowe [8] recently investigated how to quantify information flow from high to low while staying in a possibilistic setting. Using a discretely timed version of CSP, he can analyze timing channels, which we currently ignore. The aim of Lowe's work is similar to ours in that he does not per se require no information to flow from high to low: he puts bounds on the capacity of channels whereas (by the abstract window) we restrict the ways in which information may flow from high to low.

Graham-Cumming and Sanders [2] discuss the preservation of non-interference under data refinement. They specify systems using the specification language Z [15] and define security as indistinguishability on system traces with respect to a given user. They give conditions under which a refinement of the internal data of the system preserves indistinguishability. Their approach is possibilistic, and, in contrast to our setting, they consider only refinements of the internal state of a system but not of the input and output data. We emphasize refining the inputs and outputs, because an implementation must be designed in such a way that choosing particular representations of inputs and outputs does not allow adversaries to infer more information about the system than they are allowed to.

Mantel [9] considers the preservation of information flow properties under refinement. It is well-known that CSP-style refinement does not preserve information flow properties in general [5]. Mantel shows how refinement operators tailored for specific information flow properties can modify an intended refinement such that the resulting refinement preserves the given flow property. Working top-down from the specification to an implementation, the refinement operators may lead to concrete specifications that are practically hard to implement, because the changes in the refinement they induce are hard to predict and may not be easy to realize in an implementation.

Jürjens [7] uses stream processing functions to model systems, and he defines a possibilistic notion of secrecy in that setting. He identifies conditions under which certain refinement operators on stream processing functions preserve his notion of secrecy.

6 Conclusions

Security-aware engineering of systems and software needs a notion of refinement which comprises not only integrity and availability, but confidentiality as well. To contribute to providing a firm basis for security-aware engineering, we developed a precise notion of confidentiality-preserving refinement (CPR). CPR inherits all properties of behavioral refinement and additionally introduces indistinguishability preservation, which is the probabilistic characterization of confidentiality-preservation.

At the end of Section 4.1, we have identified two questions concerning the composition and refinement of secure systems: (i) Does the composed system still fulfill the desired security properties? (ii) Does the replacement of the abstract by the concrete system in a given context compromise confidentiality? Question (i) should be investigated in more detail, taking into account our definition of CPR. For this investigation, one can build on the work of Mantel [10] and Ryan and Schneider [13]. Concerning the investigation of compositionality of refinement in a probabilistic setting, i.e., question (ii), we know of no work prior to ours, as Graham-Cumming and Sanders [2], Mantel [9], and Jürjens [7] consider possibilistic refinement only.

The present paper shows that confidentiality-preserving refinement is transitive, but not compositional in general. An analysis of the situation shows that this result is not surprising. It is even inevitable, because confidentiality properties are of a fundamentally different nature than integrity (and – to a certain extent – availability) properties, which correspond to the notion of correctness as considered in classical refinement. Refining a subsystem that is embedded in a context yields a refinement of the composed system, because the refined subsystem always behaves in a way that is consistent with the behavior of the abstract subsystem. A corresponding property does not hold for confidentiality. As Section 4.2 shows, it is possible to refine a subsystem in such a way that additional ways of obtaining information about the subsystem become possible on the concrete level as compared to the abstract level. This is due to the facts that, first, the context adds an additional window to the system, and second, "non-confidential" data may be refined to data that permits an adversary to gain additional information as compared to the abstract system.

In such a situation, the only possible remedy is to investigate the conditions that must hold in addition to the confidentiality-preserving refinement of the subsystem. If proving these conditions is easier than proving the CPR for the composed systems from scratch, then the notion of confidentiality-preserving refinement is still useful for stepwise development using a divide-and-conquer approach.

With the notion of non-disclosure, we capture the additional condition that must hold to guarantee the compositionality of CPR. This condition corresponds well to the intuition that (i) the "information leaks" introduced by the context can be represented by the additional data visible in the context's window, and (ii) that adding a new window must not change the relative probabilities of indistinguishable traces. Non-disclosure does not only give insight into our notion of CPR, but also into the relationship between refinement and compositionality in general.

We can therefore conclude that the notion of CPR may be useful for security-aware engineering of systems and software, even if it cannot be compositional in general. The work presented in this paper lays the foundations for an engineering approach to CPR:

identifying architectures that guarantee non-disclosure by construction will facilitate the task of proving non-disclosure. Further research must apply our definitions and theorems to examples of larger scale, further investigate the relation of indistinguishability of traces to other security properties, as well as start the development of tools supporting our approach.

Acknowledgements. We thank Peter Ryan for extensive and constructive feedback on a previous version of the paper. The anonymous reviewers also provided useful comments. The program committee allowed us to considerably rework the paper during the acceptance process. Thanks also go to Sandra Steinbrecher and Elke Franz for asking good questions and finding some typos.

References

[1] J. A. Goguen and J. Meseguer. Security policies and security models. In *IEEE Symposium on Security and Privacy*, pages 11–20. IEEE Computer Society Press, 1982.

[2] 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.

[3] J. W. Gray. Toward a mathematical foundation for information flow security. *Journal of Computer Security*, 1992.

[4] 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.

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

[6] J. Jürjens. Secure information flow for concurrent processes. In *CONCUR 2000*, LNCS 1877. Springer-Verlag, 2000.

[7] 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.

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

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

[10] H. Mantel. On the composition of secure systems. In *IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 2002. to appear.

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

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

[13] 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.

[14] T. Santen, M. Heisel, and A. Pfitzmann. Compositionality of confidentiality-preserving refinement. Technical Report 10/2002, Technische Universität Berlin, 2002.

[15] J. M. Spivey. *The Z Notation – A Reference Manual*. Prentice Hall, 2nd edition, 1992.

[16] J. T. Wittbold and D. M. Johnson. Information flow in nondeterministic systems. In *IEEE Symposium on Security and Privacy*, pages 144–161. IEEE, 1990.