Available online at www.sciencedirect.com

ScienceDirect

Information and Computation 206 (2008) 213–249

Information and Computation

www.elsevier.com/locate/ic

Preservation of probabilistic information flow under refinement

Thomas Santen

Institut für Softwaretechnik und Theoretische Informatik, Technische Universität Berlin, Germany

Received 10 November 2006; revised 02 May 2007
Available online 28 November 2007

Abstract

Information flow properties, which describe confidentiality requirements, are not generally preserved under behavior refinement. This article describes a formal framework for refinement relations between nondeterministic probabilistic processes that capture sufficient conditions to preserve information flow properties. In particular, it uses information-theoretic concepts to investigate the refinement of a probabilistic, entropy-based information flow property. The refinement relation considers the abstract and concrete models as views on the same stochastic process. Probabilistic CSP provides the semantic basis for this investigation.

Keywords: Information flow property; Refinement; Confidentiality; Security; Information theory; CSP; Process calculus

1. Introduction

Confidentiality requirements on an IT system can be expressed as *possibilistic* or *probabilistic* information flow properties. The latter, as for example proposed by Gray [1], express the fact that an adversary cannot gain information about certain system behaviors in terms of Shannon's [2] information theory. The former are derivatives of Goguen and Meseguer's [3] noninterference property, e.g. [4,5,6,7,8,9]. These properties abstract from the stochastic behavior of a system and consider the logically possible behavior of a system only.

The present paper investigates the preservation of information flow properties under refinement. Earlier work [10] established a formal framework to investigate the conditions of preserving arbitrary information flow properties. The present article uses that framework for *confidentiality-preserving refinement* (CPR) to address the preservation of probabilistic information flow properties.

The following sections informally summarize and motivate the contributions of the present paper.

1.1. Behavior refinement

The paradigm of system and software development by stepwise refinement is a long standing one. Although rarely practiced rigorously, it provides the formal justification for modern software engineering techniques such

E-mail address: santen@acm.org

0890-5401/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.ic.2007.07.008

as behavioral subtyping for object-oriented inheritance [11], design by contract [12], and model-driven development. In its rigorous form, it has been applied, among others, in safety-critical applications [13].

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 realization may be more deterministic (*process refinement*), work on different data types (*data refinement*), or replace atomic actions by sequences of "more primitive" actions (*action refinement*). This process terminates with a completely refined model, the *implementation*, which is supposed to be easily transformable to a conventional program with equivalent semantics. In the present article, we consider a combination of process refinement and refinement of the communicated data, which we call *behavior refinement*.

A refinement relation should preserve the essential properties of the specification, be a *preorder*, and be *compositional*. Development by stepwise refinement only makes sense if the refinement relation is a preorder, i.e., it is reflexive and transitive. If a refinement relation is compositional, then all contexts are monotonic functions with respect to the refinement preorder. This means that sub-specifications can be refined to implementations independently of each other, because replacing a sub-specification by a realization in the context of a model yields a realization of that model.

Starting with Hoare [14], there is a vast body of research (e.g. [15–18]) 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.

1.2. Possibilistic information flow under process refinement

A possibilistic information flow property basically requires that, given the true system behavior and the adversary's observation of that behavior, there exist one or several alternatively possible system behaviors that produce the same observation for the adversary as the true system behavior does. Thus, the system keeps the true behavior secret from the adversary within the set of alternatives, which all produce the same adversary observation.

It is well known that refinement relations that allow one to reduce nondeterminism [19] do not preserve confidentiality properties. In particular, possibilistic information flow properties of nondeterministic specifications are not preserved under process refinement. Roscoe [20] called this the *refinement paradox*. Several approaches to deal with this deficiency of classical refinement relations have been proposed.

Roscoe et al. [6] avoid the problem by requiring the adversary's view of the system to be deterministic. This means that responses of the system to adversary inputs only depend on those inputs and not on information in the system provided by other sources. Jürjens [21] distinguishes "specification nondeterminism" from "implementation nondeterminism". He disallows specification nondeterminism whenever it influences the validity of a security property. Mantel [22] shows how refinement operators tailored for specific possibilistic information flow properties can modify an intended realization such that the resulting realization preserves the desired flow property. Ryan and Schneider [8] 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 reflects the distinction between nondeterminism for specification purposes and the kind of nondeterminism induced by probabilistic choices at run-time. The distinction between high and system nondeterminism reflects the specifiers' goal. Apparently, it is hard to reflect that difference semantically. To our knowledge, there is no established formalism that semantically distinguishes those kinds of nondeterminism and proposes a suitable refinement relation.

1.3. Probabilistic information flow in nondeterministic systems

Probabilistic information flow properties take the stochastic behavior of systems into account. Thus, they more closely reflect the probabilistic nature of information in Shannon's sense than possibilistic properties do. They not only require alternative possible system behaviors to be possible, but they also require that the rela-

tive probabilities of those behaviors satisfy certain constraints. They thus make it sufficiently unlikely for the adversary to guess the true system behavior on the basis of his or her observations of the system. Possibilistic properties disregard the stochastics of a system. Thus, they implicitly assume that all behaviors producing the same observation for the adversary are equiprobable.

Furthermore, probabilistic properties can be adapted more easily than possibilistic ones because they require bounds on a stochastic measure, which quantifies information content or information flow. This facilitates finding a compromise between idealized confidentiality requirements ("no information flow") and other kinds of requirements.

Nondeterminism is an important means of avoiding premature design bias in a specification. Moreover, certain modeling operators, such as hiding, introduce nondeterminism. Unless one considerably restricts the modeling language, it is hard to avoid nondeterministic specifications.

In short, we need to consider nondeterministic models, because we are interested in the chain of models from an abstract specification to a concrete implementation. Probabilistic properties, however, cannot be directly assigned to a nondeterministic system model, because the stochastics of nondeterministic choice are unknown. The same is true for models containing external choice, which is resolved by the environment of the model.

These observations raise the question under which conditions a nondeterministic model satisfies a probabilistic information flow property. The answer we propose is twofold.

First, following Zave and Jackson [23], we consider a *system* to consist of a *machine* in its *environment*. The machine is to be implemented, whereas the environment models assumptions on the working conditions of the machine. In an *adversary model*, 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 on user and adversary behavior, and thus to make these assumptions explicit. This system model is an extension of the one we proposed before [24,25]. It has some similarity with the one proposed by Backes et al. [26], which we discuss in Section 10.

Second, to obtain a deterministic, probabilistic model, for which we can determine the validity of a probabilistic information flow property, we consider the realizations of the system components under process refinement. Designers can influence how nondeterminism in the machine is resolved. Therefore, it is sufficient to require that there *exists* at least one deterministic machine realization that is secure. Nondeterminism in the environment model, in contrast, expresses lack of information about the environment behavior. Therefore, *all* possibilities of resolving nondeterminism in the environment need to be considered when determining the security of the system: we must require that the chosen machine realization satisfies the desired information flow property if composed with any deterministic realization of the environment. We call those compositions of machine and environment realizations *variants* of the adversary model.

In summary, a *confidentiality property* within the framework of CPR is existentially quantified over machine realizations and universally quantified over environment realizations. The composition of machine and environment realizations must satisfy a desired (probabilistic) information flow property, the *basic confidentiality property*. There is no single basic confidentiality property that is adequate in all contexts, but the choice of a particular basic confidentiality property depends on the confidentiality requirement that the property is supposed to formalize.

1.4. Probabilistic information flow under behavior refinement

Given that a confidentiality property is an existential proposition, the framework of CPR determines the conditions under which a behavior refinement of an adversary model preserves the existence of a variant that satisfies a given basic confidentiality property.

A behavior refinement reduces nondeterminism and changes the types of communicated data. Behavior refinement considered as a relation on adversary models may additionally introduce *new* means of observation for the "concrete" adversary. The more detailed description of data in the realization may give rise to new possibilities for the adversary to observe the system behavior.

A confidentiality property expressed in terms of the specification model cannot directly be applied to the realization model, because it cannot interpret the refined data and the new adversary observations. Section 8.3 discusses this additional source of complexity. It introduces "refined" versions of confidentiality properties that

interpret models in terms of a *point of reference*. In a succession of several refinement steps, the point of reference is the most abstract model, in terms of which the required confidentiality property is expressed.

The definition of CPR in the framework is parameterized by an *information flow refinement order* that relates variants of abstract and concrete adversary models, and that preserves the given basic confidentiality property. Confidentiality-preserving refinement preserves the (existential) confidentiality property on adversary models. If the information flow refinement order is a preorder, then CPR also is one.

To investigate probabilistic information flow under refinement, we instantiate our framework with a specific probabilistic information flow property and establish a suitable information flow refinement order. In Section 7, we define a probabilistic information flow property based on the *entropy* of system behaviors, given an adversary observation.

To establish an information flow refinement order for that property, we must investigate whether an adversary can obtain more information in a realization variant than in a variant of the specification. In general, this is true, because in the realization, the adversary has additional means of observing the system. The information flow refinement order we define in Section 9 restricts the *mutual information* between system behavior at the specification level and adversary observations at the realization level, given adversary observations at the specification level. If that mutual information is zero and the specification variant satisfies the probabilistic information flow property, then the realization variant also does. Section 9.1 establishes that the condition on mutual information is reflexive and transitive, which immediately implies that the information flow refinement order is a preorder (Corollary 40).

The main Theorem 44 of the present paper shows that the instantiation of the CPR framework with that information flow refinement order indeed provides a refinement relation that preserves the probabilistic confidentiality property.

1.5. Overview

The following Section 2 briefly recalls the main definitions of information theory.

Section 3 presents the process calculus of Probabilistic Communicating Sequential Processes (PCSP) [27], which extends classical CSP with a probabilistic choice operator. We use PCSP as the basis of our theory.

Section 4 introduces the system model distinguishing machine, honest users, and adversary. PCSP processes describe the behavior of the constituents of a system and their communication.

Section 5 shows how to determine the probabilities of process behaviors for a certain class of processes, which is a prerequisite to analyze the security of a system in terms of information theory.

The general structure of confidentiality properties within the framework of CPR is the topic of Section 6. Two specific confidentiality properties are defined in Section 7: a possibilistic and a probabilistic one. Those properties are prototypical for a range of known information flow properties.

The abstract framework of CPR is described in Section 8. Section 9 establishes the information-theoretic results to instantiate that framework. The instantiation yields a refinement relation that preserves the probabilistic confidentiality property of Section 7.

Section 10 puts the concepts and results presented before in the context of the state of the art. Section 11 summarizes the contributions of the present article.

Appendix A summarizes the CSP notation.

The proofs of all theorems can be obtained upon request from the author.

Remark 1. We use the syntax of the specification language Z [28] to denote sets and formulas. The set $\{x : X | P(x) \bullet f(x)\}$ comprises all $f(x)$ where $x \in X$ and $P(x)$ holds. The universal quantification $\forall x : X | P(x) \bullet Q(x)$ is true if for all $x \in X$ with $P(x)$ the proposition $Q(x)$ is true. Similarly, $\exists x : X | P(x) \bullet Q(x)$ is true if there exists $x \in X$ with $P(x)$ such that $Q(x)$ is true.

2. Information theory

This section recalls the basic definitions of information theory, as established by Shannon [2]. We need those concepts to define probabilistic confidentiality properties in Section 7 and information flow conditions pre-

serving those properties in Section 9. Here, we only touch upon the basic facts without much motivation or justification. For a comprehensive introduction to modern information theory, refer, e.g., to MacKay [29].

An *ensemble* X is a triple $(x, \mathcal{A}_X, \mathcal{P}_X)$, where the *outcome* x is the value of a random variable, which takes on one of a set of possible values, $\mathcal{A}_X = \{x_1, \ldots, x_n\}$, with probabilities $\mathcal{P}_X(x_i) = p_i \in [0,1]$, such that $\Pr(X = x_i) = p_i$ and $\sum_{i=1}^{n} p_i = 1$. Usually, we write just X to mean the ensemble, the random variable, or the set $\mathcal{A}_X$ of possible values of x.

Given two (in general dependent) random variables X and Y, the *joint ensemble* $((x, y), \mathcal{A}_X \times \mathcal{A}_Y, \mathcal{P}_{X,Y})$ describes the experiment of choosing x and y simultaneously. We write X, Y for that ensemble and the corresponding random variable, and we write $\Pr(X = x, Y = y)$ for the joint probability of the outcome (x, y).

Given a random variable X, the information content in the probabilistic event that X assumes the value x with positive probability $\Pr(X = x)$ is the logarithm of the reciprocal of that probability, $\log_2 \frac{1}{\Pr(X=x)}$. Since we will only use the binary logarithm, we subsequently write log for $\log_2$.

The *entropy* $H(X)$ describes the expected information content of all possible outcomes $x \in X$. We will mostly be interested in *conditional entropy*. The entropy $H(X|Y = y)$ of some random variable X given an outcome y of the random variable Y determines the *uncertainty* about X that remains after observing y. The entropy $H(X|Y)$ is the expected uncertainty of X for varying y.

$$H(X) = \sum_{x \in X} \Pr(X = x) \cdot \log \frac{1}{\Pr(X = x)}$$

$$H(X|Y = y) = \sum_{x \in X} \Pr(X = x|Y = y) \cdot \log \frac{1}{\Pr(X = x|Y = y)}$$

$$H(X|Y) = \sum_{y \in Y} \Pr(Y = y) \cdot \sum_{x \in X} \Pr(X = x|Y = y) \cdot \log \frac{1}{\Pr(X = x|Y = y)}$$

Entropy is non-negative. It is maximal if X is uniformly distributed. Then $H(X) = \log |X|$. The entropy of X is zero if there is $x_0 \in X$ with $\mathcal{P}_X(x_0) = 1$.

The *mutual information* $I(X; Y)$ determines the amount of information that Y reveals about X (and vice versa). It measures the information flow between X and Y. As for the entropy, there are conditional variants of mutual information given a third random variable Z.

$$I(X; Y) = \sum_{x \in X; y \in Y} \Pr(X = x, Y = y) \cdot \log \frac{\Pr(X = x, Y = y)}{\Pr(X = x)\Pr(Y = y)} \tag{1}$$

$$I(X; Y|Z = z) = \sum_{x \in X; y \in Y} \Pr(X = x, Y = y|Z = z) \cdot \log \frac{\Pr(X = x, Y = y|Z = z)}{\Pr(X = x|Z = z)\Pr(Y = y|Z = z)} \tag{2}$$

$$I(X; Y|Z) = \sum_{x \in X; y \in Y; z \in Z} \Pr(X = x, Y = y, Z = z) \cdot \log \frac{\Pr(X = x, Y = y|Z = z)}{\Pr(X = x|Z = z)\Pr(Y = y|Z = z)} \tag{3}$$

Fact 2

(1) Mutual information is non-negative: $I(X; Y) \geq 0$.
(2) Two random variables X and Y are independent iff $I(X; Y) = 0$.
(3) Mutual information is symmetric.

$$I(X; Y) = I(Y; X) \quad I(X; Y|Z) = I(Y; X|Z)$$

(4) Mutual information can be expressed in terms of entropy.

$$I(X; Y) = H(X) - H(X|Y) \quad I(X; Y|Z) = H(X|Z) - H(X|Y, Z)$$

(5) *Chain Rule* for mutual information: Let $X_1, \ldots, X_n, Y$ be random variables.

$$I(X_1, \ldots, X_n; Y) = \sum_{i=1}^{n} I(X_i, \ldots, X_n; Y | X_1, \ldots, X_{i-1}) \tag{4}$$

In Section 9, we need to exploit the fact that two random variables X and Z are independent given knowledge about a third one, Y. This is true if Y provides at least as much information about Z as X does, and therefore knowledge about X does not contribute to determining Z. That kind of independence is captured by the concept of a Markov Chain.

Definition 3 (*Markov Chain*)**.** Let X, Y, and Z be random variables. They form a *Markov Chain* $X \to Y \to Z$ if X and Z are independent given Y, i.e., for all x, y, and z:

$$\Pr(X = x, Y = y, Z = z) = \Pr(X = x) \cdot \Pr(Y = y | X = x) \cdot \Pr(Z = z | Y = y)$$

The following fact restates the independence of the random variables in a Markov Chain in terms of conditional probabilities.

Fact 4. *Let X, Y, and Z be random variables forming a Markov Chain $X \to Y \to Z$. Then for all x, y, z with* $\Pr(X = x, Y = y) > 0$*:*

$$\Pr(Z = z | Y = y) = \Pr(Z = z | X = x, Y = y)$$

There is a close correspondence between Markov Chains and the conditional mutual information between the involved random variables: there is no information flow from the first to the last random variable in a Markov Chain, and all information from the first to the last variable flows through the intermediate random variable.

Fact 5. *Let X, Y, and Z be random variables.*

(1) *$X \to Y \to Z$ is a Markov Chain iff $I(X; Z|Y) = 0$*
(2) *If $X \to Y \to Z$ is a Markov Chain then*

$$I(X; Z) \leq I(X; Y) \tag{5}$$

$$I(X; Z) \leq I(Y; Z) \tag{6}$$

3. Probabilistic communicating sequential processes

To formally model the systems we reason about, we use the probabilistic extension PCSP of the process algebra CSP [30,18], which Morgan et al. [27] 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 built upon the concept of refinement. This supports well our investigation of the relationship of refinement and probabilistic information flow.

Appendix A summarizes the PCSP notation we use.

3.1. 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 notations $c?d$ for an incoming event and $c!d$

for an outgoing event indicate the intended direction of a communication. The semantics of CSP, however, does not distinguish input from output: both, $c?d$ and $c!d$, are semantically equal to $c.d$.

The set of traces of P is $traces(P)$. It is closed under prefixing. The length of a trace t is $\#t$. The set $traces(P) \downarrow k$ consists of the traces of P with a length less than or equal to k.

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 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 unified because an event contains a variable), then the parallel composition generates that event, otherwise the parallel composition deadlocks. As long as P or Q produce events not in X they proceed asynchronously.

The composition $P \Vdash_X Q$ 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 $P \Vdash_X Q$ 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.

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[1] $P[\![R]\!]_D$ 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 $P \backslash X$ is distinguished from P by *hiding* the channels in X. The traces of $P \backslash X$ are the traces of P where all events over channels in X are removed. Similarly, the process $P \lceil X$ is P restricted to the channels in X.

The external choice $P \sqcap Q$ 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. In particular, any trace of Q also is a trace of P. For details, see Roscoe [18].

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 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:

$$P \uparrow = \{Q : CSP | P \sqsubseteq Q\} \tag{7}$$

Example 6. The following serves as a running example throughout the paper.

Consider the scoring system of a bank, which they use to determine the credit-worthiness of account holders. Upon each transfer between accounts, the system determines a new score for the debited account based on some unknown criteria.

Fig. 1 shows a CSP model of the scoring system. The set *HOLDER* of account holders comprises three people, *alice*, *bob*, and *yves*. For simplicity, we identify account holders with their accounts. A *SCORE* is either *good* or *poor*, and the *RESULT* of a transfer is *ok* or *error*.

The process $System_0$ is the parallel composition of the processes $Bank_0$ and $Customers_0$, which synchronize on the channel *tr*.

The process $Bank_0$ models the behavior of the scoring system. Initially, it nondeterministically chooses scores for the three account holders to initialize the state of the process $Transfer_0$, which contains the scores of all account holders. The process $Transfer_0$ obtains a transfer from channel *tr*. The events on *tr* abstract from the transferred amount and just specify the debited (*from*) and the credited (*to*) accounts. How the result *res* and the new score *ns* are determined, is left unspecified: they are chosen nondeterministically, and written to the channels *score* and *result*, respectively. The function $trans_0(state, from, ns)$ produces a new state from the input parameter *state* by updating the score of the account *from* to the score *ns*. With the resulting state, the process $Transfer_0$ recurs.

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

$$Bank_0 \mathrel{\widehat{=}} \bigsqcap_{s_1,s_2,s_3:SCORE} Transfer_0(\{(alice, s_1), (bob, s_2), (yves, s_3)\})$$

$$Transfer_0(state) \mathrel{\widehat{=}} tr?from.to \rightarrow \bigsqcap_{ns:SCORE;\ res:RESULT} score!ns \rightarrow result!res \rightarrow Transfer_0(trans_0(state, from, ns))$$

$$Customers_0 \mathrel{\widehat{=}} \bigsqcap_{from:HOLDER;\ to:HOLDER-\{from\}} tr!from.to \rightarrow Customers_0$$

$$System_0 \mathrel{\widehat{=}} Bank_0 \,|[\,\{tr\}\,]|\, Customers_0$$

Fig. 1. Abstract account scoring system of a bank.

Similarly, the environment process $Customers_0$ nondeterministically generates transfers between accounts on the channel tr.

The process $System_0$ has the channels tr, $score$, and $result$. Its traces consist of sequences of triples of events like $\langle tr.f.t, score.s, result.r\rangle$:

$$traces(System_0) = \{\langle tr.f_i.t_i, score.s_i, result.r_i\rangle^{*_i}\}$$

We use the notation $s_i^{*_i}$ to denote any finite concatenation of sequences $s_1 \frown \ldots \frown s_n$.

3.2. Probabilistic CSP

Morgan et al. [27] 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 a particular behavior probabilistically. It rather emphasizes that a probabilistic process behaves like a standard CSP 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. Morgan et al. [27] present the full theory.

Informally, a probabilistic process P is a function mapping a "Scott-open" set Y of classical CSP processes to the probability that P behaves like a member of Y. A set Y of processes is Scott-open if, first, for any of its members it contains all processes refining it, and second, for any chain of refinements of processes whose limit is in Y already a member of that chain is in Y. These conditions guarantee that Y is a complete set of "similar" processes. The set of *probabilistic processes PCSP* is the set of continuous evaluations mapping Scott-open sets of standard CSP processes to $[0,1]$.

Let P and Q be probabilistic processes in $PCSP$. The process Q *refines* P iff for all Scott-open $Y \subseteq CSP$, the probability of Q behaving like Y is at least as high as the probability of P behaving like Y: $P(Y) \leq Q(Y)$. Since standard CSP processes can be embedded in PCSP and the refinement orders coincide, we write $P \sqsubseteq Q$ for PCSP refinement, too.

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

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

$$(P_p\oplus Q)(Y) = p \cdot P(Y) + (1-p) \cdot Q(Y)$$

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,

$$F \sqsubseteq P_p\oplus Q = p \cdot (F \sqsubseteq P) + (1-p) \cdot (F \sqsubseteq Q)$$

$$
\begin{aligned}
Bank_1 &\mathrel{\widehat{=}} \bigoplus^{\mathcal{P}_b}_{b_1,b_2,b_3:BALANCE} \\
&\quad Transfer_1(\{(alice, b_1, scr(b_1)), \\
&\qquad (bob, b_2, scr(b_2)), (yves, b_3, scr(b_3))\}) \\
Transfer_1(state) &\mathrel{\widehat{=}} tr?from.to.amount \rightarrow \\
&\quad \bigoplus^{\mathcal{P}_s}_{ns:Scores(state,from,amount)} score!ns \rightarrow \\
&\quad \textbf{if } adm_1(state, from, to, amount) \\
&\quad \textbf{then } result!ok \rightarrow mi!size(amount) \rightarrow \\
&\qquad Transfer_1(trans_1(state, from, to, amount, ns)) \\
&\quad \textbf{else } result!error \rightarrow Transfer(state) \\
Customers_1 &\mathrel{\widehat{=}} \bigsqcap_{from:HOLDER;\ to:HOLDER-\{from\};\ amount:BALANCE} \\
&\quad tr!from.to.amount \rightarrow Customers_1 \\
System_1 &\mathrel{\widehat{=}} Bank_1 \,[|\,\{tr\}\,|]\, Customers_1
\end{aligned}
$$

Fig. 2. Concrete scoring system.

Furthermore, any non-recursive probabilistic process can be expressed as a probabilistic choice of finitely many standard processes, because probabilistic choice distributes through all operators of CSP. We use this fact in Section 3.3.

Note 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.

$$P \sqcap Q \sqsubseteq \begin{cases} P \oplus_p Q \\ P \,\Box\, Q \end{cases}$$

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

A process is *deterministic* if it (semantically) does not contain nondeterministic choice. A deterministic process P is fully refined: there is no process $Q \neq P$ such that $P \sqsubseteq Q$.

Example 7. The processes in Fig. 2 specify the banking system of Fig. 1 in more detail. An account data item now contains not only the name of the holder and a score but also the balance of the account. The process $Bank_1$ chooses the initial balances probabilistically, according to the distribution $\mathcal{P}_b$. The function *scr* determines the initial scores from the initial balances.

A transfer event on channel *tr* now also contains the amount to transfer. When a transfer is initiated by an event *tr.from.to.amount*, the process $Transfer_1$ chooses a new score for *from* according to the distribution $\mathcal{P}_s$ from a set of possible scores, which the function *Scores* determines from the current state of the account and the transferred amount. The idea is that *Scores* implements an algorithm to determine possible scores, whereas the probabilistic choice models how the bank takes other "soft" criteria into account to determine scores.

The chosen score *ns* is published on the channel *score*, but the score of the debited account is only updated to *ns* if the transfer is admissible (determined by the predicate adm_1) and is actually carried out (by the function $trans_1$ that yields the updated state). In that case, qualitative information about the amount of money transferred is published on the channel *mi*: the function *size* returns the value *small* or *large*.

If the transfer is not admissible, e.g., because the balance of the account to debit is too low, then only the negative result of the transfer is published on the channel *result*.

The environment process $Consumers_1$ behaves similarly to $Consumers_0$. In addition to the accounts involved, it also chooses the amount to transfer nondeterministically.

3.3. Probabilistic linear processes

We consider probabilistic confidentiality properties that refer to the probability of a process QE performing a trace t. Thus we interpret the process QE as 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 may contain 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* [31,32]. Let X be a set of channels. We call a process E an *admissible* environment if $Q \Vdash_X E$ is deterministic and does not contain any external choices. For an admissible environment E, the k-approximation $(Q \Vdash_X E) \downarrow k$ is *probabilistic linear*, i.e., there is a probability function $\mathcal{P}_E$ such that

$$(Q \Vdash_X E) \downarrow k = \bigoplus\nolimits_{t \in traces(Q)\downarrow k}^{\mathcal{P}_E} \mathrm{Fin}_k(t)$$

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.

4. System model

Security, and confidentiality in particular, is a system-wide property, which depends not only on the behavior of the implemented IT system but also on the behavior of the environment in which the system works. Therefore, we consider a *system* to consist of a *machine* in its *environment* [23]. The environment model must express assumptions on the behavior of the users and the adversary. Accordingly, a system model consists of three PCSP processes, as shown in Fig. 3: 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. The union of the adversary interfaces $W = AI \cup EI \cup MI$ is the *adversary window*.

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. That bound models the maximal time that the adversary spends on observing the system before drawing conclusions from the observation.

The set $\mathcal{E}^{EI,W}_{P,k}(H, A)$ comprises all deterministic probabilistic realizations of the environment process $H|[EI]|A$ that are admissible for P (cf. Section 3.3). A proposition about the probabilistic behavior of an adversary model must therefore consider the probabilistic linear processes $(Q \Vdash_{MI} E) \downarrow k$ where $Q \in P^\top$ and $E \in \mathcal{E}^{EI,W}_{P,k}(H, A)$.

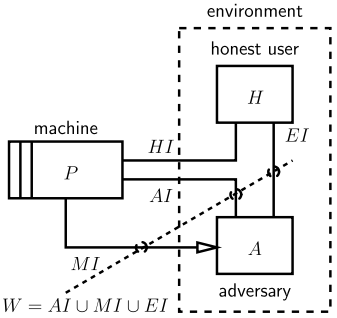

Fig. 3. A system consists of a machine and its environment.

Those processes (process-) refine the system in its environment, and we call them the *variants* of the adversary model:

$$\forall Q : P^{\top}; E : \mathcal{E}_{P,k}^{EI,W}(H,A) \bullet P \Downarrow_{MI} (H|[EI]|A) \sqsubseteq Q \Downarrow_{MI} E \tag{8}$$

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

$$s \equiv_W t \iff s\restriction W = t\restriction W \tag{9}$$

where $s\restriction W$ 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 *indistinguishability class* $J_W^{P,k}(o)$ 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 that P produces with traces consisting of at most k events.

$$J_W^{QE,k}(o) = \{t : traces(QE) \mid t\restriction W = o \wedge \#t \leq k\} \tag{10}$$

$$\mathrm{Obs}_W^k(QE) = \{t : traces(QE) \mid \#t \leq k \bullet t\restriction W\} \tag{11}$$

Given an observation o, the adversary does not immediately know which member of $J_W^{P,k}(o)$ caused the observation (unless that set is a singleton). In their work on possibilistic information flow properties [5], Zakinthinos and Lee call indistinguishability classes the *low level equivalence sets* of a specification.

Example 8. An adversary model AM_0 for Yves in the abstract scoring system of Fig. 1 consists of the machine model $Bank_0$, the environment model $Customers_0$, the adversary interface $AI_0 = \{tr.yves, tr.alice.yves, tr.bob.yves\}$, and the monitoring interface $MI_0 = \{result\}$. The environment interface between Yves and the honest users Alice and Bob is empty.

The definition of $Customers_0$ does not directly reflect the distinction between honest users and adversary. It is possible to provide an equivalent but slightly more elaborate process definition which makes this distinction explicit. For brevity, we do not present such a process definition here. We also take a liberal view on the distinction between channels and events in the "dotted" notation: Yves has access to all transfer events on *tr* where his account is debited (*tr.yves*) or where his account is credited (*tr.alice.yves*, *tr.bob.yves*).

The adversary window for Yves is $W_0 = \{tr.yves, tr.alice.yves, tr.bob.yves, result\}$. An observation on W_0 is a sequence of either event-pairs $\langle tr.f.t, result.r\rangle$, where Yves is involved in the transfer (as f or t) and r is the result of the transfer, or events on *result*, for transfers in which Yves is not involved. Thus, Yves directly observes the results of transfers in which he is not involved, and (obviously) observes his own actions and their results. Yves cannot directly observe any scores, neither his own nor Alice's or Bob's scores.

To keep the example manageable, we restrict the length of considered traces to $k_0 = 3$.

5. Probabilistic linear processes as random variables

Probabilistic confidentiality properties of adversary models consider the stochastic behavior of processes in terms of traces. This is possible only for probabilistic linear processes, because they choose all events probabilistically. For a probabilistic linear process $QE = (Q \Downarrow_W E) \downarrow k$, the probability $\mathcal{P}_E(t)$ is $\mathrm{Fin}_k(t) \precsim QE$. The probability of QE producing exactly the first $\min\{k, \#t\}$ events of t therefore is

$$\Pr_{QE}^k(t) = (\mathrm{Fin}_k(t) \precsim QE) \tag{12}$$

5.1. Probability of an observation

Let $o \in \mathrm{Obs}^k_W(QE)$ be an observation of QE on the adversary window W. Because we restricted the length of the members of $J^{QE,k}_W(o)$ to k, this implies that QE produces *exactly* o when watched for k steps. Therefore, the *probability of the observation o in k steps on the adversary window W*, $\Pr^{W,k}_{QE}(o)$, is given by

$$\Pr^{W,k}_{QE}(o) = \sum_{t \in J^{QE,k}_W(o)} \Pr^k_{QE}(t) \tag{13}$$

Eq. (13) defines $\Pr^{W,k}_{QE}(o)$ in terms of the members of $J^{QE,k}_W(o)$. It is also possible to express $\Pr^{W,k}_{QE}(o)$ as a single probability of refinement that regards only the events on the adversary window W. To achieve this, we need to embed QE into a context that restricts the visible events to the ones on W, and that also restricts the number of events of the original QE to at most k. Counting those events achieves the latter, while the former amounts to hiding the events on all channels but the members of W. The process term $[QE]^k_W = (QE \parallel \mathit{Step}(k)) \restriction W$ puts QE in a suitable context, where the process $\mathit{Step}(k)$ counts up to k arbitrary events and deadlocks afterwards.

$$[QE]^k_W = \bigoplus\nolimits^{\mathcal{P}_T}_{t \in traces(QE)\downarrow k} ((\mathrm{Fin}_k(t) \parallel \mathit{Step}(k)) \restriction W) \tag{14}$$

A refinement relationship between finite processes characterizes the fact that a trace is an observation of another trace: for all $t \in traces(QE) \downarrow k$ and traces o on W with $\#o \leq k$, the following equivalence holds:

$$o = t \restriction W \iff \mathrm{Fin}_k(o) \sqsubseteq [\mathrm{Fin}_k(t)]^k_W \tag{15}$$

Therefore, we can express the probability $\Pr^{W,k}_{QE}(o)$ in terms of the probability that the standard processes on the right-hand side of Eq. (14) refine the process that produces the observation o. The probability $\Pr^{W,k}_{QE}(o)$ is equal to the probability of QE refining the k-finite process for o on W while producing at most k events.

$$\Pr^{W,k}_{QE}(o) = \mathrm{Fin}_k(o) \sqsubseteq\!\!\!\approx [QE]^k_W \tag{16}$$

5.2. Posterior probability

We now define the conditional probability of a trace t given an observation o. The joint probability of a trace t and its corresponding observation $t \restriction W$ is equal to the probability of t.

$$\Pr(t, o) = \begin{cases} \Pr(t) & \text{if } o = t \restriction W \\ 0.0 & \text{otherwise} \end{cases} \tag{17}$$

Therefore, the conditional probability $\Pr^{W,k}_{QE}(t|o)$ of a trace t given the observation o can be expressed in terms of QE refining linear processes producing t and o.

$$\Pr^{W,k}_{QE}(t|o) = \begin{cases} \frac{\mathrm{Fin}_k(t) \sqsubseteq QE}{\mathrm{Fin}_k(o) \sqsubseteq [QE]^k_W} & \text{if } t \in J^{QE,k}_W(o) \\ 0.0 & \text{otherwise} \end{cases} \tag{18}$$

Finally, we extend the definition (12) of the probability of a trace to the probability of a set T of traces.

$$\Pr^k_{QE}(T) = \sum_{t \in T} \Pr^k_{QE}(t) \tag{19}$$

Similarly, we define the conditional probabilities $\Pr^{W,k}_{QE}(T|o)$, $\Pr^k_{QE}(T|T')$, and $\Pr^{W,k}_{QE}(T|o, T')$ given an observation o or a set of traces T' in terms of the posterior probability (18) of a trace.

5.3. Entropy of a process

The results we established in this section allow us to consider probabilistic linear processes as random variables over k-bounded traces. We let our notation reflect the correspondence between processes and random variables, and call the entropy of the set of traces of a process QE up to the length k the entropy $H(QE,k)$ of that process. Similarly, we define the entropy of a process given a particular observation, or given arbitrary observations.

$$H(QE,k) = \sum_{t \in traces(QE)\downarrow k} \Pr^k_{QE}(t) \cdot \log \frac{1}{\Pr^k_{QE}(t)} \tag{20}$$

$$H((QE,k)|W = o) = \sum_{t \in traces(QE)\downarrow k} \Pr^{W,k}_{QE}(t|o) \cdot \log \frac{1}{\Pr^{W,k}_{QE}(t|o)} \tag{21}$$

$$H((QE,k)|W) = \sum_{o \in \mathrm{Obs}^k_W(QE)} \sum_{t \in J^{QE,k}_W(o)} \Pr^k_{QE}(t) \cdot \log \frac{1}{\Pr^{W,k}_{QE}(t|o)} \tag{22}$$

Eq. (22) holds because of Eq. (17) for the joint probability of a trace and the corresponding observation. (The second sum ranges over the indistinguishability class of o only.)

6. 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. The following Section 7 introduces a possibilistic and a probabilistic property, which are instances of the abstraction defined in the present section.

6.1. Basic confidentiality properties

In earlier work [33], we have discussed several confidentiality properties based on indistinguishability. *Possibilistic* confidentiality properties, such as the various information flow properties that Mantel [9] 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 variant of an adversary model; Eq. (8)), 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. In the following discussion of the structure of confidentiality properties, a predicate $\mathcal{CP}(QE,W,k)$ serves as a placeholder.

6.2. Structure of confidentiality properties

What are the conditions under which an adversary model satisfies a confidentiality property based on $\mathcal{CP}(QE,W,k)$? More precisely, which variants QE of the adversary model must satisfy $\mathcal{CP}(QE,W,k)$ in order to call the adversary model "secure"?

Since it has become known that possibilistic information flow properties are closure properties [9,7], the observation that refinement does not preserve confidentiality in general is not so surprising anymore: refinement reduces nondeterminism and thus reduces the set of traces of a system. A closure property requires that,

given a member of a set, certain other items are also members of that set. Therefore, a process refinement, in general, does not preserve closure properties.

Consequently, we cannot expect *all* variants QE of a machine 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 the machine model 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.

As we cannot avoid nondeterministic adversary models but also cannot expect all variants of an adversary model to satisfy a basic confidentiality property, we take an optimistic view on the development of the machine and a pessimistic view on the behavior of the environment. The implementation of the machine model is under the control of the developers. They can influence the way nondeterminism is resolved by design decisions. Therefore, it suffices to ensure that *there exists* a secure machine refinement. The environment, in contrast to the machine process, must be considered with all variations that the adversary model permits, because nondeterminism in the environment model is resolved by the actual environment at run-time. Therefore, *all* $E \in \mathcal{E}_{P,k}^{EI,W}(H, A)$ need to be considered for evaluating the security of a system. In particular, this allows a security analysis to consider an 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.

In summary, 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.

Definition 9 (*Confidentiality Property*)**.** Given an adversary model $(P, H, A, HI, AI, EI, MI, k)$ and a basic confidentiality property $\mathcal{CP}$, a *confidentiality property based on* $\mathcal{CP}$ has the following general form:

$$\exists Q : P^{\top} \bullet \forall E : \mathcal{E}_{P,k}^{EI,W}(H, A) \bullet \textbf{let } QE = (Q \Vdash_{MI} E) \downarrow k \bullet \mathcal{CP}(QE, W, k) \tag{23}$$

Fig. 4 illustrates Definition 9. The basic confidentiality property $\mathcal{CP}$ must hold for *at least one* probabilistic deterministic realization Q of the machine model P (at the lower left of the figure) when it is executed in *any* admissible environment in $\mathcal{E}_{P,k}^{EI,W}(H, A)$ (at the upper right of the figure).

The rationale justifying Definition 9 considers a machine model as an abstraction of the envisaged machine implementation, which the developers choose carefully so as to avoid insecure implementations. The environment model, on the other hand, limits the behaviors of honest users and adversaries. The developers have only limited influence on the environment. Therefore all possible behavior within the limits that the environment model sets must be considered in a security evaluation.

Definition 9 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 10. 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 high-performing implementation, but not all functionally correct implementations of the model necessarily satisfy the real-time constraints.

7. Possibilistic and probabilistic confidentiality properties

It is not always obvious what confidentiality property can adequately capture the confidentiality requirements of a particular system. Apparently, there is no single property that allows specifiers to capture all desirable attri-

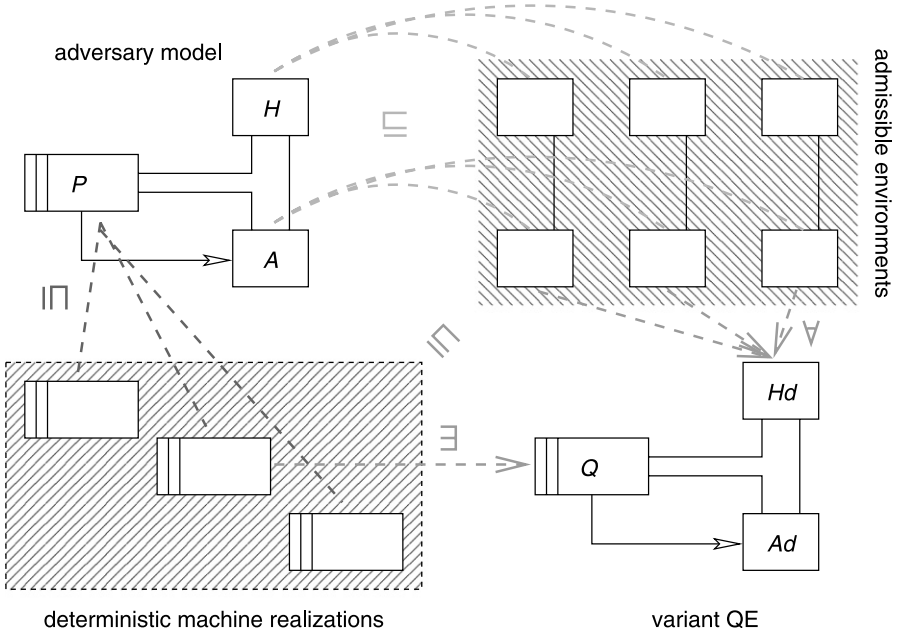

Fig. 4. Confidentiality properties are existential on the machine realizations and universal on the environment realizations.

butes of confidentiality that may be relevant for any conceivable system. Therefore, it is important for a formal model of confidentiality to allow specifiers to express different kinds of confidentiality properties within the same framework. We define two confidentiality properties, which are prototypical for possibilistic and probabilistic properties. Their relation to other properties published before is discussed in Section 10.

(1) The possibilistic property of *concealed behavior* considers differences between traces to be kept confidential if they are not directly distinguishable by different observations.
(2) The probabilistic property of *ensured entropy* builds on information theory. It relates confidentiality to the uncertainty about the true system behavior that remains after observing the system through the adversary window.

The concept of indistinguishable behavior is the foundation of those definitions: a confidentiality property needs to refer to the indistinguishability relation that an adversary model induces. However, being a property that an adversary model may or may not satisfy, a confidentiality property must relate the indistinguishability that the adversary model induces to the *required* indistinguishability of system behaviors that reflects the confidentiality requirement in question. We use the concept of a *mask* to express the required indistinguishability of system behaviors.

Definition 11 (*Mask*)**.** A *mask* $\mathcal{M}$ for an adversary model $(P, H, A, HI, AI, EI, MI, k)$ is a set of subsets of the traces over the alphabets of P, H, and A such that the members of each set are indistinguishable by the adversary window $W = AI \cup EI \cup MI$:

$$\forall M : \mathcal{M}; t_1, t_2 : M \bullet t_1 \equiv_W t_2 \tag{24}$$

A mask need *not* partition the set of system traces. Indeed, $\bigcup \mathcal{M}$ may contain traces that the system process cannot produce, and the system process may produce traces not in the mask. The idea motivating the definition of a mask is that the adversary model is required to conceal the differences between the members of a set in $\mathcal{M}$ to the extent specified by a specific confidentiality property. There is no confidentiality requirement for the traces of the system process that are not contained in $\bigcup \mathcal{M}$. These traces are "don't cares."

Example 12. Consider the following confidentiality requirement for the adversary model AM_0 of Example 8.

R1: "Yves does not learn Alice's or Bob's scores."

If Yves observes the single event *result.ok* on W_0, then he knows that exactly one transfer between Alice and Bob has taken place and was successful. All traces of the form

$$t_0(f,t,s) = \langle tr.f.t, score.s, result.ok\rangle$$

where f and t are *alice* or *bob*, produce the observation *result.ok* for Yves.

According to Requirement *R*1, this observation should not allow Yves to infer Alice's or Bob's scores. From the definition of the process $Transfer_0$, Yves knows that s is the new score of f if $System_0$ performs $t_0(f,t,s)$. A mask $\mathcal{M}_0$ supporting *R*1 consequently needs to require that for a given f all variations of s in the parameters of t_0 are possible causes of the observation *result.ok*. Therefore, the sets

$$M_0 = \{t_0(alice, bob, good), t_0(alice, bob, poor)\}$$

$$M_1 = \{t_0(bob, alice, good), t_0(bob, alice, poor)\}$$

should be members of $\mathcal{M}_0$.

Taking $M_0 \cup M_1$ as a single member of $\mathcal{M}_0$ would additionally require that Yves cannot distinguish the direction of the transfer. This would be a requirement stronger than *R*1.

The mask should contain similar sets for the other observations Yves can make.

7.1. Concealed behavior

The confidentiality property of concealing a mask $\mathcal{M}$ directly relates to the set-theoretic properties of indistinguishability. If an adversary model is to keep confidential the differences between the traces in a set $M \in \mathcal{M}$, then those traces must—at least—be indistinguishable by its adversary window. The following Definition 14 formalizes that property by set inclusion: $M \subseteq J_W^{QE,k}(o)$, where the process QE is a variant of the adversary model.

The preceding set inclusion requires the system to be capable of producing *all* members of M. Because the members $M \in \mathcal{M}$ of the mask are not required to produce the same observation, this inclusion will not hold for all $M \in \mathcal{M}$ and all indistinguishability classes $J_W^{QE,k}(o)$ of the adversary model. Therefore, we require that a member of $\mathcal{M}$ is either completely contained in an indistinguishability class or not at all. We say that the set of indistinguishability classes $\mathcal{I}$ *covers* $\mathcal{M}$.

Definition 13 (*Coverage*)**.** Let $\mathcal{M}$ and $\mathcal{I}$ be two sets of sets of traces. Then $\mathcal{I}$ *covers* $\mathcal{M}$, written $\mathcal{I} \ni \mathcal{M}$, if $\forall M : \mathcal{M}; I : \mathcal{I} \bullet M \cap I = \emptyset \vee M \subseteq I$.

The confidentiality property of *concealed behavior* requires that an adversary model can cover a given mask, i.e., there is a machine implementation Q such that the indistinguishability classes of all variants of the adversary model derived from Q cover the mask.

Definition 14 (*Concealed Behavior*)**.** The adversary model (P,H,A,HI,AI,EI,MI,k) *conceals* the mask $\mathcal{M}$, written $Conceals_{\mathcal{M}}(P,H,A,HI,AI,EI,MI,k)$, if

$$\exists Q : P^{\top} \bullet \forall E : \mathcal{E}_{P,k}^{EI,W}(H,A) \bullet \textbf{let } QE = (Q \Vdash_{MI} E) \downarrow k \bullet \{o : \mathrm{Obs}_W^k(QE) \bullet J_W^{QE,k}(o)\} \ni (\mathcal{M} \downarrow k)$$

Depending on the mask that is to be covered, concealed behavior can be used to formalize a wide range of confidentiality requirements. The strength of the requirement that a mask imposes depends on the fraction of the unobservable traces $traces(P \backslash W)$ that each member of the mask contains. It also depends on the fraction of observations the mask addresses at all.

In one extreme case, each M contains all possible sequences of unobservable events, i.e., $M \setminus W = traces(P \setminus W)$, and the observations covered are all possible observations, i.e., $\bigcup_{M:\mathcal{M}} M \restriction W = \mathrm{Obs}^k_W(P)$. Then the observations on W must be completely independent of the unobservable behavior of the system.

In the other extreme, the mask does not restrict the system behavior at all because the members of $\mathcal{M}$ are singletons or because they are associated with observations that the system does not produce.

In the general case, concealed behavior does not prevent information flow from the machine to the adversary: the distinction between observations on the adversary window, in general, allows the adversary to restrict the possible behavior that caused a particular observation: the adversary can exclude all behavior that does not produce that observation, and can thus infer information about the unobservable workings of the system.

Example 15. To determine whether the adversary model AM_0 conceals the mask $\mathcal{M}_0$, we need to find a deterministic machine realization $Bank^1_0$ of $Bank_0$ such that its composition with all realizations of the environment process $Customers_0$ covers $\mathcal{M}_0$.

For $Bank^1_0$, we choose the implementation of $Bank_0$ that resolves all nondeterministic choices of $Bank_0$ by probabilistic choices with equal probabilities for all alternatives.

The admissible environments comprise realizations of $Customers_0$ that produce only one particular sequence of transfers, e.g., only transfers from Alice to Bob, and also realizations of $Customers_0$ that probabilistically choose the accounts to debit and to credit.

The members M_0 and M_1 of $\mathcal{M}_0$ are covered by the indistinguishability classes of all resulting variants of $System_0$, because $Bank^1_0$ chooses the score of Alice (or Bob) independently of the transfers that the environment initiates.

The union $M_0 \cup M_1$, however, is *not* covered by all variants, because the variant whose environment process deterministically produces only transfers from Alice to Bob performs the traces in M_0 but not the ones in M_1. This means that with this environment given, Yves can infer who transfers money to whom.

7.2. Ensured entropy

Possibilistic information flow properties require certain system traces to exist that produce the same observation as a given one, but they do not consider the probability of the system to actually produce one of those traces. If adversaries know something about the relative probabilities of indistinguishable traces, then a particular observation o may tell them more than just the fact that the system performs some member $J^{QE,k}_W(o)$. The adversary's *uncertainty* about the true system behavior that remains given an observation o varies with the relative probabilities of the members of $J^{QE,k}_W(o)$. The conditional entropy is a measure of the adversary's uncertainty about the true system behavior given an observation o. Since entropy is a probabilistic concept, it is clear that it is well-defined only on the variants of an adversary model.

Definition 16 (*Entropy of a Variant Given an Observation*)**.** Let $(P, H, A, HI, AI, EI, MI, k)$ be an adversary model, $E \in \mathcal{E}^{EI,W}_{P,k}(H, A)$ be an admissible environment, $QE = (Q \Vdash_{MI} E) \downarrow k$ be a variant of the adversary model, and $o \in \mathrm{Obs}^k_W(QE)$ be an observation on W. The *entropy of the variant QE given the observation o* is the conditional entropy $H((QE, k) | W = o)$ as defined in Eq. (21).

The entropy $H((QE, k) | W = o)$ of a variant QE is maximal if all $t \in J^{QE,k}_W(o)$ have the same posterior probability; then $H((QE, k) | W = o) = \log\left(\left|J^{QE,k}_W(o)\right|\right)$. In this case, the only information about the system behavior flowing to the adversary is that *some* member of $J^{QE,k}_W(o)$ caused the observation o, but there is no reason to prefer any of them over the others, and probabilistic reasoning does not provide more insight for the adversary than possibilistic reasoning does.

On the other hand, if $H((QE, k) | W = o) = 0$, then there is a trace $t_0 \in J^{QE,k}_W(o)$ with $\Pr^{W,k}_{QE}(t_0 | o) = 1$. In this case, the observation o actually tells the adversary for sure that the process P performed t_0.

In summary, the entropy of a variant QE given an observation o measures the degree to which the variant keeps confidential the differences between the members of $J^{QE,k}_W(o)$: the higher that entropy, the better the variant hides the differences of those behaviors; the smaller the entropy, the more probabilistic reasoning makes a difference for the adversary in drawing conclusions from an observation.

Definition 16 does not take the relative probabilities of observations into account. Those, however, are also important to evaluate the degree to which a system keeps certain information confidential. The entropy of a variant given *any* observation is the mean of the entropies given an observation. It thus takes the distribution of observations into account.

Definition 17 (*Entropy of a Variant*)**.** Let $(P, H, A, HI, AI, EI, MI, k)$ be an adversary model, $E \in \mathcal{E}_{P,k}^{EI,W}(H, A)$ be an admissible environment, $QE = (Q \Vdash_{MI} E) \downarrow k$ be a variant of the adversary model. The *entropy of the variant* QE is the conditional entropy $H((QE, k)|W)$ as defined in Eq. (22).

Turning the concept of entropy into a probabilistic confidentiality property, the property of *ensured entropy* takes both measures, the entropy of a variant and the entropy of a variant given an observation into account. It imposes lower bounds on the entropy of the indistinguishability classes that cover the member sets of a mask $\mathcal{M}$, and on the entropy of a variant as well. If an adversary model also conceals the given mask $\mathcal{M}$, it thus guarantees a prescribed degree of uncertainty for an observing adversary.

Instantiating Definition 9, we obtain an entropy-based confidentiality property: we say that an adversary model ensures the entropy specified for the classes of a mask $\mathcal{M}$, if there is a maximal refinement of the system process such that the entropy of all variants obtained by putting that maximal refinement in an admissible environment is bounded from below by the entropy associated with the members of the mask $\mathcal{M}$. Additionally, we require the entropy of all those variants to be bounded from below as well.

None of these conditions is redundant. The restrictions on the entropy of a variant given an observation allow one to distinguish confidentiality requirements for individual indistinguishability classes whose expected entropy is equal (because the associated observations are equally probable or because their entropies are equal).

Technically, a total function mapping the members of $\mathcal{M}$ to non-negative real values specifies lower bounds on the entropy of the classes in $\mathcal{M}$.

Definition 18 (*Ensured Entropy*)**.** Let $\mathcal{H}$ map classes of the mask $\mathcal{M}$ and the entire mask as well to possible values of entropy. The adversary model $(P, H, A, HI, AI, EI, MI, k)$ *ensures the entropy* $\mathcal{H}$ *for* $\mathcal{M}$, written $Ent_{\mathcal{M}}^{\mathcal{H}}(P, H, A, HI, AI, EI, MI, k)$, if

$$
\begin{aligned}
\exists Q : P^{\top} \bullet \forall E : \mathcal{E}_{P,k}^{EI,W}(H, A) \bullet \textbf{let } QE = (Q \Vdash_{MI} E) \downarrow k \bullet \\
\big(\{o : \mathrm{Obs}_W^k(QE) \bullet J_W^{QE,k}(o)\} \ni (\mathcal{M} \downarrow k) \wedge \\
\mathcal{H}(\mathcal{M}) \leq H((QE, k)|W) \wedge \\
\forall M : \mathcal{M} \bullet \forall o : \mathrm{Obs}_W^k(QE)|M \downarrow k \subseteq J_W^{QE,k}(o) \bullet \\
\mathcal{H}(M) \leq H((QE, k)|W = o)\big)
\end{aligned}
$$

Example 19. The set $M_0 \cup M_1$ is the largest set of traces that a variant of AM_0 can produce for the observation $\langle result.ok \rangle$. The maximally possible entropy of this set is $H_{01} = \log 4 = 2$. Therefore, it is sensible to choose entropy bounds for M_0 and M_1 in the range of 0 to 4. Requiring $\mathcal{H}(M_0) = 0$ means that we do not care whether Yves can probabilistically infer Alice's score from the observation $\langle result.ok \rangle$, whereas $\mathcal{H}(M_1) = 4$ means that Bob's and Alice's scores must all be equally probable, i.e., Yves must neither be able to infer who transfers money to whom nor be able to determine the score of the debited account.

As the example shows, Definition 18 admits a wide range of entropy bounds. The value assigned to $\mathcal{H}(M)$ may even be greater than the maximal entropy possible for M. In this case, $\mathcal{H}$ requires the containing indistinguishability classes to be larger than M, i.e. it would require the system to produce additional "camouflage" behavior with the same observations as the ones in M.

The entropy of an indistinguishability class depends on the cardinality of that class. To model confidentiality requirements adequately it is instrumental to come up with an adversary model at the right level of abstraction: one that distinguishes the important information by different model elements but that—to the extent possible—does not introduce distinctions that are unnecessary to express the required confidentiality property. Those additional distinctions may well be unavoidable in an implementation. Sections 8 and 9 show how to introduce those distinctions into a model while preserving the confidentiality properties of the original one.

8. Confidentiality preserving refinement (CPR)

This section introduces the framework of confidentiality preserving refinement. We investigate the conditions under which a confidentiality property (Definition 9) is preserved under behavior refinement of adversary models.

In Section 8.1, we define behavior refinement of adversary models. The refinement relation determines whether a "concrete" adversary model is a functionally correct realization of an "abstract" adversary model. The concrete model may be more deterministic than the abstract one. It may also refine the data in communication events.

In a behavior refinement, the concrete adversary window may extend the abstract one. This reflects the fact that an adversary may have different means of accessing and observing the system in a realization than are captured in the specification, because the realization abstracts from less detail of the conceived implementation than the specification does. As a consequence, 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).

In addition to functional correctness, a "secure" refinement relation must 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 interpreted in terms of the data model of the confidentiality property of the specification.

The framework of CPR reduces this condition to a relation on behaviorally matching variants of adversary models. Section 8.2 introduces a *re-abstraction* preorder on the variants of the specification and the realization adversary models that relates those variants that are in the behavior refinement relation.

Section 8.3 introduces *refined confidentiality properties*. These are interpretations of a property of an adversary model in terms of other adversary models.

The confidentiality preserving refinement relation we introduce in Section 8.5 determines an abstract condition to preserve a confidentiality property: it requires that the re-abstraction preorder coincides with another preorder on variants of adversary models, the *information flow refinement order*. The latter is defined in Section 8.4. It depends on the confidentiality property in question and ensures that this property is preserved *on variants*.

Thus, the framework of CPR reduces the question whether a confidentiality property is preserved by a behavior refinement of adversary models to the question whether the property is preserved by behaviorally matching, i.e., re-abstracted, variants of the adversary models.

An instantiation of the framework for a particular confidentiality property must provide an information flow refinement order that is appropriate for that property. Section 9 presents one for ensured entropy (Definition 18).

In the following, we assume that $\mathcal{A} = (P_a, H_a, A_a, HI_a, AI_a, MI_a, EI_a, k_a)$ and $\mathcal{C} = (P_c, H_c, A_c, HI_c, AI_c, MI_c, EI_c, k_c)$ are adversary models.

8.1. 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 [17] 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 20 (*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?

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 21 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 address the question whether those additional observations compromise the security of the system.

Definition 21 (*Behavior Refinement of Adversary Models*)**.** Let $\mathcal{A}$ and $\mathcal{C}$ be two adversary models. 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

$$P_a \Vdash_{MI_a} (H_a|[EI_a]|A_a) \sqsubseteq_{R_{ca}} (P_c \Vdash_{MI_c} (H_c|[EI_c]|A_c))\backslash(W_c - W_a)$$

Example 22. An adversary model AM_1 for Yves consists of the processes defined in Fig. 2. Compared to AM_0, the monitoring interface of this model includes channel *mi* in addition to *result*. The adversary window of AM_1 is the one of AM_0 extended by *mi*. To accommodate the events on *mi*, we consider system traces with a length of $k_1 = 4$. The data on channel *tr* of AM_1 contains the transferred amount, which is not mentioned in AM_0. The retrieve relation R_0^1 establishes a relation between the data communicated in AM_1 and AM_0: it discards the amount a in a transfer event.

$$R_1^0 = \{f, t : HOLDER; a : BALANCE \bullet (f.t.a, f.t)\}$$

With R_1^0, we can prove the behavior refinement on processes

$$System_0 \sqsubseteq_{R_1^0} System_1\backslash\{mi\}$$

This establishes that AM_1 is a behavior refinement of AM_0. (Note that even if AM_0 was deterministic, AM_1 could nondeterministically choose the amounts transferred on *tr*. In this way, behavior refinement can introduce nondeterminism.)

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 $\mathcal{A} \sqsubseteq_{R_{ba}} \mathcal{B} \wedge \mathcal{B} \sqsubseteq_{R_{cb}} \mathcal{C} \Rightarrow \mathcal{A} \sqsubseteq_{R_{cb} \fatsemi R_{ba}} \mathcal{C}$ hold.

8.2. 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 23 (*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 $(QE_a, W_a) \overset{\nwarrow}{\sqsupseteq}_{R_{ca}} (QE_c, W_c)$, is defined by

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

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

Example 24. Consider a process refinement $Bank_0^2$ of $Bank_0$ that chooses all scores probabilistically with the same probabilities as $Bank_1$ does. The composition BC_0 of $Bank_0^2$ with the process refinement of $Customers_0$ which initiates only transfers from Alice to Bob is a variant of AM_0.

Similarly, composing $Bank_1$ with any refinement of $Customers_1$ that initiates only transfers from Alice to Bob (and chooses the amounts to transfer probabilistically) yields variants BC_1^i of AM_1, which are re-abstracted refinements of BC_0.

In contrast, all variants of AM_0 that choose scores with different probabilities than $Bank_1$ do not have any re-abstracted refinements in AM_1 because a distribution of a particular probabilistic choice cannot change in a PCSP refinement.

8.3. Refined confidentiality properties

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

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

Fig. 5 illustrates this setting, which considers three levels of abstraction. First, at the top of the figure, a point of reference (W_r, R_r, k_r) describes the distinctions between system behavior that an adversary is allowed to make. The plain represents the set of possible system behaviors at that level of abstraction, and the dotted line separates indistinguishability classes induced by the adversary window W_r. In a sequence of consecutive refinement steps, this level would coincide with the most abstract model that constitutes the beginning of the sequence of refinements.

Second, the abstract adversary model makes up the center row of the figure. The retrieve relation R_r relates the system behaviors at this level back to the top level: the abstract trace s_1 refines r_1, and s_2 refines r_3 of the point of reference; both, s_3 and s_4 refine r_2.

Third, the bottom level of the figure shows the concrete model, which the retrieve relation R_{ca} connects to the abstract model: the concrete traces t_1, t_2, and t_3 all implement the abstract trace s_4, and—by transitivity of behavior refinement—they also implement the trace r_2 of the point of reference, i.e., the composition $R_{ca} \fatsemi R_r$ relates them to r_2.

8.4. Information flow refinement

Re-abstraction relates the "matching" variants of the abstract and the concrete 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 25 (*Information Flow Refinement*)**.** Let $\mathcal{CP}_r$ be a refined confidentiality property. A preorder $(QE_a, W_a) \preccurlyeq^{k_a,k_c}_{R_{ca}} (QE_c, W_c)$ on pairs of probabilistic linear processes and adversary windows is called an *information flow*

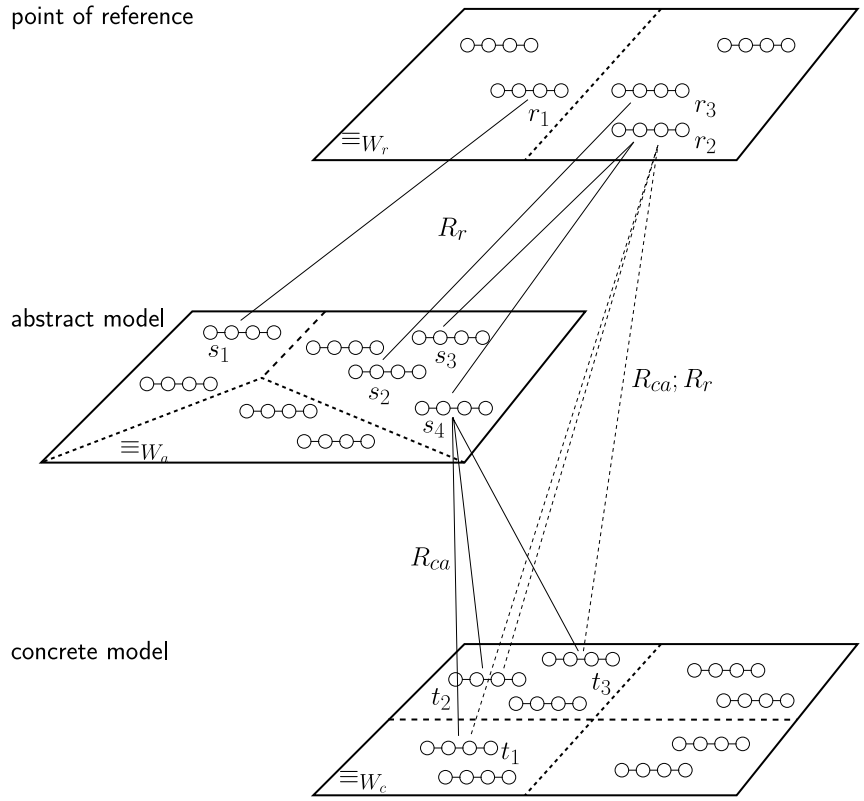

Fig. 5. Levels of abstraction in a refinement with point of reference.

refinement relation for $\mathcal{CP}_r$ with the point of reference (W_r, R_r, k_r) if it strengthens the re-abstraction preorder and is sufficient to preserve $\mathcal{CP}_r$, i.e., for all adversary models $\mathcal{A}$ and $\mathcal{C}$ 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:

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

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.

We postpone a detailed discussion of information flow refinement relations to Section 9, which is devoted to analyzing an information flow refinement for ensured entropy.

8.5. CPR

Under which condition is a behavioral refinement of adversary models a confidentiality-preserving one? Because confidentiality properties are existential propositions (Definition 9), a behavior refinement does not necessarily admit 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 this property.

Definition 26 (*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:

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

In conjunction with the first implication in Definition 25, 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 $\mathcal{C}$. This means that to prove that a behavior refinement preserves confidentiality, one needs to prove that any pair of re-abstracted variants is in the information flow refinement relation.

Because information flow refinement is a preorder, CPR also is a well-behaved refinement relation.

Lemma 27 (*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 28 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 28 (*CPR preserves $\mathcal{CP}_r$*)**.** *Let $\mathcal{CP}_r$ be a refined basic confidentiality property with point of reference (W_r, R_{ar}, k_r). Let $\preccurlyeq$ be an information flow refinement relation for $\mathcal{CP}_r$. If $\mathcal{A} \sqsubseteq^{\preccurlyeq}_{R_{ca}} \mathcal{C}$ then the following implication holds:*

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

Remark 29. A careful analysis of the line-up of the quantifiers in Proposition 28 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 25. Unfortunately, the resulting definition of CPR is not transitive, because the required witnesses for the variant of the intermediate adversary model need not match.

9. Information flow preserving refinement

Section 8 investigated the preservation of arbitrary confidentiality properties under behavior refinement. The main result is that the preservation of a basic confidentiality property from abstract to concrete

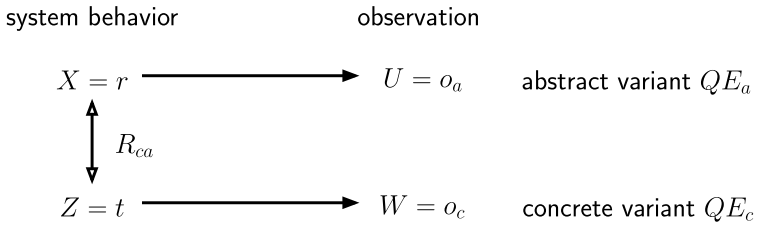

Fig. 6. Random Variables in a Refinement.

variants—established by an information flow refinement relation—is sufficient to preserve the corresponding general confidentiality property under behavior refinement of adversary models.

We now aim at establishing an information flow refinement relation for the confidentiality property of ensured entropy. To this end, we analyze the amount of new information an adversary may obtain from a refined adversary model. We consider the variants of the abstract and concrete adversary models as random variables producing system traces. From these, random variables on observations are derivable. We argue that the mutual information between the abstract system traces and the concrete observations, given an abstract observation, is a suitable measure for the additional information an adversary gains by observing the concrete rather than the abstract system. We describe the relevant random variables and the information flow condition in Section 9.1. Section 9.2 turns that condition into a preorder on pairs of random variables. Section 9.3 applies these information-theoretic results to variants of adversary models and derives a condition that preserves the confidentiality property of ensured entropy in a refinement.

9.1. Mutual information in a refinement

Consider a variant QE_a of an adversary model that is a re-abstraction of a variant QE_c of another adversary model for the adversary windows W_a and W_c via the retrieve relation R_{ca}: $(QE_a, W_a) \sqsupseteq_{R_{ca}} (QE_c, W_c)$. Furthermore, assume that the re-abstraction does not introduce nondeterminism, such that QE_a is equal to the re-abstracted version of QE_c:

$$QE_a = (QE_c \backslash (W_c - W_a))[\![R_{ca}]\!]_D \tag{25}$$

As discussed in Section 5.3, being a probabilistic linear process, a variant of an adversary model can be associated with random variables describing the way the variant chooses to perform a certain trace or to produce a certain observation. For QE_a and QE_c, we introduce the following random variables, whose dependencies Fig. 6 illustrates.

$X = r$	The variant QE_a produces the trace $r \in \mathit{Traces}(QE_a) \downarrow k_a$.
$U = o_a$	The variant QE_a produces the observation $o_a \in \mathrm{Obs}^{k_a}_{W_a}(QE_a)$.
$Z = t$	The variant QE_c produces the trace $t \in \mathit{Traces}(QE_c) \downarrow k_c$.
$W = o_c$	The variant QE_c produces the observation $o_c \in \mathrm{Obs}^{k_c}_{W_c}(QE_c)$.

The interpretation of adversary models depends on their role in a refinement: the abstract model describes what an adversary *may* observe or do, whereas the concrete model describes what an adversary *can* observe or do. Consequently, a secure refinement is one that does not allow an adversary to observe or do more than the abstract model allows the adversary to observe or do. For confidentiality, this means that the *concrete* observations do not provide more information about the *abstract* behavior than the abstract observations do.

Example 30. The variants BC_0 and BC^i_1 of AM_0 and AM_1, respectively, satisfy Eq. (25), because $(BC^i_1 \backslash \{mi\})[\![R^0_1]\!]_D$ are deterministic for all i: abstracting from the amounts in transfer events does not introduce nondeterminism.

Examples for the traces mentioned in Fig. 6 are:

$$r = \langle tr.alice.bob, score.poor, result.ok \rangle$$

$o_a = \langle result.ok \rangle$

$t = \langle tr.alice.bob.100, score.poor, result.ok, mi.alice.bob \rangle$

$o_c = \langle result.ok, mi.large \rangle$

The question now is, how much more information Yves gains about Alice's score by observing o_c than by o_a, i.e., what does the event *mi.large* tell about Alice's score that the event *result.ok* would not reveal already?

Our task now is to find an information-theoretic measure for the information that an adversary may gain by observing QE_c through W_c about the behavior of QE_a in addition to the information the adversary gains through W_a. In this setting, the two processes QE_c and QE_a must be considered models—which are abstractions—of the *same* system executing, because we wish to analyze the amount of information each model provides about the system behavior. This assumption is justified, because QE_c is a re-abstracted refinement of QE_a, which means that the retrieve relation R_{ca} couples the behaviors of the two models.

In terms of the random variables of Fig. 6, we consider the following experiment: the concrete variant produces the system trace t ($Z = t$), and the adversary observes o_c at the concrete window W_c ($W = o_c$). Because the two models describe the same system behavior, the abstract model produces a trace r ($X = r$) with observation o_a ($U = o_a$) such that r is an abstraction of t by the retrieve relation R_{ca}. *Assuming* that the abstract model produces the observation o_a constrains the values of the other random variables. This assumption is justified, because the adversary may know the abstract observations—and may draw conclusions from them. In particular, the adversary may link abstract to concrete observations. Then the question is, given the observation o_a, how much does the observation o_c tell an adversary about the abstract system behavior r? In terms of information theory, this amounts to analyzing the flow of information from X to W given $U = o_a$.

The conditional mutual information $I(X; W|U = o_a)$ is a measure for that information flow. If it is positive then the adversary gains more information from the concrete observations than from the abstract observations. Conversely, the refining variant QE_c does not allow the adversary to gain more information about the abstract system behavior than QE_a if there is no flow from X to W given $U = o_a$, i.e., $I(X; W|U = o_a) = 0$ for all o_a. Since the mutual information is non-negative, this condition is equivalent to considering the expected mutual information given U and requiring

$$I(X; W|U) = 0 \tag{26}$$

Eq. (26) is equivalent to requiring that $X \to U \to W$ is a Markov Chain.

Example 31. In the example, we need to analyze the mutual information from the traces of BC_0 to the new observations on channel *mi*, given the abstract observations on W_0. If, for a successful transfer (*result.ok*), Alice's score depends on the size of the transferred amount (*small* or *large*), then revealing the size to Yves at channel *mi* provides him with more information than he receives through the channel *result*.

Conversely, the additional channel *mi* in the monitoring interface of BC_1^i does not compromise the confidentiality of Alice's and Bob's scores only if the scores are stochastically independent of the size of the transferred amounts.

9.2. Mutual information preorder

The following definition turns Eq. (26) into a preorder on the pairs of random variables.

Definition 32 (*Mutual Information Order*)**.** The *mutual information order* $\preccurlyeq$ is a relation on pairs of random variables defined by

$$(X, U) \preccurlyeq (Z, W) \iff I(X; W|U) = 0$$

The remainder of this section verifies that $\preccurlyeq$ is a preorder. We first establish the transitivity of $\preccurlyeq$, because this needs some effort. Consider two consecutive refinements and the corresponding random variables as illustrated

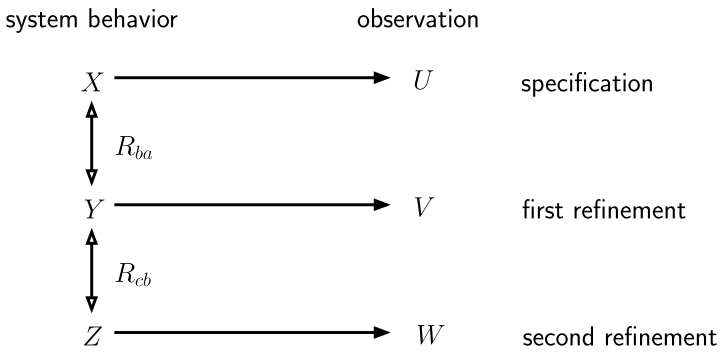

Fig. 7. Random Variables in Two Successive Refinements.

in Fig. 7. Since the random variables X, Y, and Z represent consecutive refinements, and U, V, and W the corresponding observations, we know that, given the state of a process, the corresponding observation is independent of the states of the other processes and of the other observations. We also know that an observation does not provide more information than the corresponding system state does. Therefore, the assumptions (27) through (31) of the following Lemma 33 are valid. They require that certain constellations of the random variables in Fig. 7 are Markov Chains. Furthermore, given the abstract system state X, the refined observations V and W are independent of the abstract observation U (assumptions (32) and (33)).

Given all those assumptions, which are direct consequences of the situation the random variables describe, there is no flow of information from the abstract system state X to the most refined observations W, given the abstract observations U, if the corresponding conditions independently hold for the two refinement steps. This is the statement of Lemma 33.

Lemma 33. *Let X, Y, U, V, and W be random variables that are related as in Fig. 7. In particular, the following are Markov Chains:*

$$X \to Y \to W \tag{27}$$

$$U \to X \to W \tag{28}$$

$$V \to Y \to W \tag{29}$$

$$X \to Y, V \to W \tag{30}$$

$$U \to Y, V \to W \tag{31}$$

Furthermore, if X is given, then the probabilities of W and V are independent of U.

$$\Pr(W|X) = \Pr(W|X, U) \tag{32}$$

$$\Pr(V|X) = \Pr(V|X, U) \tag{33}$$

Under these assumptions, the following implication holds: If

$$X \to U \to V \tag{34}$$

$$Y \to V \to W \tag{35}$$

then

$$X \to U \to W \tag{36}$$

Reflexivity of $\preccurlyeq$ follows from the fact that $H(X|U, U)$ is equal to $H(X|U)$, which justifies:

Corollary 34. The mutual information order $\preccurlyeq$ is a preorder.

9.3. Refinement preserving ensured entropy

We finally use the information-theoretic results of the previous section to instantiate the framework of CPR. This yields a refinement relation on adversary models that preserves ensured entropy. Since concealed behavior is a necessary condition of ensured entropy, we also show that that refinement relation preserves concealed behavior.

We proceed as follows. Sections 9.3.1 and 9.3.2 introduce "refined" versions of concealed behavior and ensured entropy relative to a point of reference. On that basis, Section 9.3.3 defines an information flow refinement relation that is sufficient to preserve ensured entropy.

Throughout this section, we assume that the retrieve relation R_r in the point of reference (W_r, R_r, k_r) is the graph of a function.

We also assume that $\mathcal{C}$ is a behavior refinement of $\mathcal{A}$, $\mathcal{A} \sqsubseteq_{R_{ca}} \mathcal{C}$, and that W_r is a subset of the adversary windows W_a and W_c. We consider two variants QE_a and QE_c of the adversary models $\mathcal{A}$ and $\mathcal{C}$, respectively, such that $(QE_a, W_a) \overset{\nwarrow}{\sqsupseteq}_{R_{ca}} (QE_c, W_c)$ and Eq. (25) holds, i.e., that QE_a not only refines the re-abstraction of QE_c but is equal to it.

9.3.1. Refined concealed behavior

For the adversary model $\mathcal{A}$, the point of reference (W_r, R_r, k_r) can be interpreted as describing an anonymous adversary model which is refined by $\mathcal{A}$: the point of reference abstracts from certain data in $\mathcal{A}$, and it distinguishes a subset $W_r \subseteq W_a$ of the adversary window of $\mathcal{A}$.

A refined version of concealed behavior needs to take the point of reference into account in such a way that, first, it considers the traces whose data are abstracted by R_r, and second, it allows the adversary to distinguish traces through W_r (at the abstract level) and to assume an (abstract) observation o_r on W_r as given. In Fig. 5, for example, o_r could represent the indistinguishability class of r_2, i.e., the right half of the plain representing the traces in the point of reference. This would imply that the adversary can distinguish s_1 from s_2 because s_1 implements r_1, whereas s_2 implements r_3, which the adversary can distinguish from r_1. To capture those effects of a point of reference, we define a version $J[\![R_r]\!]^{W_r}_{W_a}$ of a set of traces J that is abstracted to the point of reference (W_r, R_r, k_r). Because R_r is the graph of a function and W_r is a subset of W_a, the members of $J[\![R_r]\!]^{W_r}_{W_a}$ are indistinguishable by W_r if the members of J are indistinguishable by W_a.

$$J[\![R_r]\!]^{W_r}_{W_a} = (J \backslash (W_a - W_r))[\![R_r]\!]_D \tag{37}$$

In the example of Fig. 5, the set $J^{QE_a,k_a}_{W_a}(s_2 \restriction W_a)[\![R_r]\!]^{W_r}_{W_a}$ is contained in the indistinguishability class of r_2, because s_2, s_3 and s_4 are members of $J^{QE_a,k_a}_{W_a}(s_2 \restriction W_a)$ and $\{(s_1, r_1), (s_2, r_3), (s_3, r_2), (s_4, r_2)\} \subseteq R_r$.

Eq. (37) allows us to define a refined version of concealed behavior (cf. Definition 14). The refined property requires a variant QE to cover the mask $\mathcal{M}$ *in terms of the point of reference*—because the mask contains traces at that level of abstraction. To achieve this, the refined basic confidentiality property $\mathcal{CP}_{cb}(QE, W, k, W_r, R_r, k_r)$ for concealed behavior considers the system of "abstracted" indistinguishability classes of QE.

$$\widehat{\mathcal{CP}}_{cb}(QE, W, k, W_r, R_r, k_r) \iff \left\{ o : \mathrm{Obs}^k_W(QE) \bullet J^{QE,k}_W(o)[\![R_r]\!]^{W_r}_W \right\} \ni (\mathcal{M} \downarrow k_r) \tag{38}$$

Example 35. In Example 15, we saw that AM_0 conceals $\mathcal{M}_0$. To determine whether AM_1 satisfies a similar property, we must re-interpret the traces of AM_1 in terms of AM_0, i.e., in the point of reference (W_0, R^0_1, k_0). For example, it does not make sense to ask whether $\langle tr.alice.bob.42, score.good, result.ok, mi.small \rangle$ is a member of M_0, because the type of data communicated over tr does not match and the channel mi is new in AM_1. Instead, we need to check whether the "abstraction" $\langle tr.alice.bob, score.good, result.ok \rangle$ is a member of M_0.

9.3.2. Refined ensured entropy

Definition 18 introduced the confidentiality property of ensured entropy, which guarantees that the indistinguishability classes that cover the members of a mask have entropies exceeding the lower bounds associated to those members of the mask. For the refined version of the property, those are bounds on entropies with respect

to the point of reference, i.e., of data-abstracted traces that are indistinguishable through W (and W_r). In Fig. 5, the "abstract' entropy of the indistinguishability class of s_3—given some o_r—would consider the probabilities of the r_i occurring in the abstract model. Thus, $\{s_3, s_4\}$ would be *one* probabilistic event representing r_2: the probability of r_2 in the adversary model $\mathcal{A}$ is the probability of the event that QE_a produces the traces s_3 or s_4: $\Pr(r_2) = \Pr(\{s_3, s_4\}) = \Pr(s_3) + \Pr(s_4)$.

Furthermore, the observations on the adversary window of the point of reference and the abstract adversary model are assumed to be known to the adversary. In terms of the random variables of Fig. 6, the entropy we are interested in is $H(X|W = o, U = o_r)$. The following Eq. (39) presents this entropy for a variant QE of an adversary model. The conditional entropy $H((QE, k, W_r, R_r)|W = o, (W_r, R_r) = J_r)$ with respect to the point of reference refers to the probabilities of the traces of QE, where J_r is an indistinguishability class with respect to W_r that consists of traces in the data range of R_r. Thus, J_r represents an observation o_r in the point of reference.

$$
\begin{aligned}
&H((QE, k, W_r, R_r, k_r)|W = o, (W_r, R_r, k_r) = J_r) = \\
&\quad \textbf{let } J = \{t : J_W^{QE,k}(o) \mid R_r(t\backslash(W - W_r)) \in J_r\}; \\
&\qquad \mathcal{T} = \{t : J \bullet \{t' : J \mid R_r(t'\backslash(W - W_r)) = R_r(t\backslash(W - W_r))\}\} \bullet \\
&\qquad \sum_{T \in \mathcal{T}} \left(\Pr^k_{QE}(T|J) \cdot \log \frac{1}{\Pr^k_{QE}(T|J)} \right)
\end{aligned}
\tag{39}
$$

To reflect the assumption that both, the observation o and the observation represented by J_r are given, Eq. (39) constructs the set J of all traces t of QE that produce o on W and that have an abstraction $R_r(t\backslash(W - W_r))$ in J_r. The set J describes the "given" stochastic event.

The system $\mathcal{T}$ partitions J into the sets of traces of QE that represent the same trace in the point of reference. We are interested in the entropy in the point of reference. Therefore, those traces need to be considered as one stochastic event, namely that QE produces their common abstraction. Consequently, the entropy is calculated in terms of the probabilities of the members T of $\mathcal{T}$ given J. If $J_r = \{r | r \equiv_{W_r} r_2\}$ in Fig. 5, then the class $J = \{s_2, s_3, s_4\}$ and the system $\mathcal{T} = \big\{\{s_2\}, \{s_3, s_4\}\big\}$.

The complexity of Eq. (39) is due to references to two levels of abstraction: the point of reference and the adversary model. The analysis in Section 9.1—with Eq. (26) suitably instantiated—reveals that the entropy $H((QE, k, W_r, R_r, k_r)|W = o, (W_r, R_r, k_r) = J_r)$ can be simplified if the mutual information between the traces of the point of reference and the observations of the adversary model is zero.

The following Eq. (40) instantiates Eq. (2). It defines the mutual information $I(X; W|U)$ (in the terminology of Fig. 6) in terms of two variants QE_a and QE_c satisfying Eq. (25). For a given trace r of QE_a, and observations o_c and o_a, the set T comprises all traces of QE_c that contribute to the stochastic event "$X = r, W = o_c, U = o_a$." The set O_a represents the event "$U = o_a$," and the indistinguishability class $J^{QE_c,k_c}_{W_c}(o_c)$ represents "$W = o_c$."

$$
\begin{aligned}
&I((QE_a, k_a); (QE_c, k_c, W_c)|(QE_a, k_a, W_a)) = \\
&\sum_{r \in traces(QE_a)\downarrow k} \quad \sum_{o_c \in \mathrm{Obs}^{k_c}_{W_c}(QE_c)} \quad \sum_{o_a \in \mathrm{Obs}^{k_a}_{W_a}(QE_a)} \\
&\textbf{let } T = \{t \in traces(QE_c) \downarrow k_c | \\
&\qquad t\restriction W_c = o_c \wedge r = R_{ca}(t\backslash(W_c - W_a)) \wedge r\restriction W_a = o_a\}; \\
&\quad O_a = \{t \in traces(QE_c) \downarrow k_c | o_a = R_{ca}(t\backslash(W_c - W_a))\restriction W_a\} \bullet \\
&\Pr^{k_c}_{QE_c}(T) \cdot \log \frac{\Pr^{k_c}_{QE_c}(T|O_a)}{\Pr^{W_a,k_a}_{QE_a}(r|o_a) \cdot \Pr^{k_c}_{QE_c}(J^{QE_c,k_c}_{W_c}(o_c)|O_a)}
\end{aligned}
\tag{40}
$$

With Lemma 2, we immediately have the following presentation of the mutual information in Eq. (40) as a difference of conditional entropies.

$$
\begin{aligned}
&I((QE_a, k_a); (QE_c, k_c, W_c)|(QE_a, k_a, W_a)) \\
&\qquad = H((QE_a, k_a)|(QE_a, k_a, W_a)) \\
&\qquad\quad - H((QE_a, k_a)|(QE_c, k_c, W_c), (QE_a, k_a, W_a))
\end{aligned}
\tag{41}
$$

With those preliminaries, we can define the refined version of ensured entropy. The following condition directly rephrases the basic confidentiality property of ensured entropy in terms of a point of reference.

$$\begin{aligned}
&\widehat{\mathcal{CP}}_{cb}(QE, W, k, W_r, R_r, k_r) \wedge \\
&\mathcal{H}(\mathcal{M}) \leq H((QE, k, W_r, R_r, k_r)|W, (W_r, R_r, k_r)) \wedge \\
&\forall M : \mathcal{M}; o : \mathrm{Obs}^k_W(QE) \bullet \mathbf{let}\ J_r = J^{QE,k}_W(o)[\![R_r]\!]^{W_r}_W \bullet \\
&\quad M \downarrow k_r \subseteq J_r \Rightarrow \\
&\quad \mathcal{H}(M) \leq H((QE, k, W_r, R_r, k_r)|W = o, (W_r, R_r, k_r) = J_r)
\end{aligned} \tag{42}$$

The following Definition 36, however, uses a stronger condition that nevertheless is equivalent to $\mathcal{CP}_{ee}$ for the trivial point of reference. In addition to the Predicate (42), the refined basic property $\widehat{\mathcal{CP}}_{ee}(QE, W, k, W_r, R_r, k_r)$ requires the mutual information between behavior at the point of reference and observations of the variant QE to be zero. Under that condition, the following equality between entropies holds:

$$\begin{aligned}
&\forall M : \mathcal{M}; o : \mathrm{Obs}^k_W(QE) \bullet \mathbf{let}\ J_r = J^{QE,k}_W(o)[\![R_r]\!]^{W_r}_W \bullet \\
&M \downarrow k_r \subseteq J_r \Rightarrow \\
&H((QE_r, k)|(QE_r, k, W_r) = J_r) \\
&= H((QE, k, W_r, R_r, k_r)|W = o, (W_r, R_r, k_r) = J_r)
\end{aligned} \tag{43}$$

Eq. (43) justifies to formulate Definition 36 in terms of the entropy at the point of reference only.

Definition 36 (*Refined Ensured Entropy*)**.** Let $\mathcal{H} : \mathcal{M} \to \mathbb{R}^+$ map classes of $\mathcal{M}$ to possible values of entropy. The adversary model $(P, H, A, HI, AI, MI, EI, k)$ *ensures the entropy* $\mathcal{H}$ *for* $\mathcal{M}$ *with the point of reference* (W_r, R_r, k_r), if $W_r \subseteq W$, R_r is the graph of a function, and

$$\begin{aligned}
&\exists Q : P^\top \bullet \forall E : \mathcal{E}^{EI,W}_{P,k}(H, A) \bullet \mathbf{let}\ QE = (Q \Downarrow_W E) \downarrow k \bullet \\
&\qquad \widehat{\mathcal{CP}}_{ee}(QE, W, k, W_r, R_r, k_r)
\end{aligned} \tag{44}$$

where

$$\begin{aligned}
&\widehat{\mathcal{CP}}_{ee}(QE, W, k, W_r, R_r, k_r) \iff \\
&\quad \left\{o : \mathrm{Obs}^k_W(QE) \bullet J^{QE,k}_W(o)[\![R_r]\!]^{W_r}_W\right\} \ni (\mathcal{M} \downarrow k_r) \wedge \\
&\quad \big(\mathbf{let}\ QE_r = (QE \backslash (W - W_r))[\![R_r]\!]_D \bullet \\
&\qquad I((QE_r, k_r); (QE, k, W)|(QE_r, k_r, W_r)) = 0 \wedge \\
&\qquad \mathcal{H}(\mathcal{M}) \leq H((QE_r, k_r)|(QE_r, k_r, W_r)) \wedge \\
&\qquad \forall M : \mathcal{M}; o : \mathrm{Obs}^k_W(QE) \bullet \mathbf{let}\ J_r = J^{QE,k}_W(o)[\![R_r]\!]^{W_r}_W \bullet M \downarrow k_r \subseteq J_r \Rightarrow \\
&\qquad\quad \mathcal{H}(M) \leq H((QE_r, k_r)|(QE_r, k_r, W_r) = J_r)\big)
\end{aligned} \tag{45}$$

Example 37. The elementary stochastic events in refined ensured entropy are the sets of traces in an indistinguishability class that "realize" the same trace in the point of reference. Consider the "abstract" trace $t_a = \langle tr.alice.bob, score.good, result.ok\rangle$ of AM_0 (which we interpret as the point of reference for AM_1). For the observation $o = \langle result.ok, mi.small\rangle$ of AM_1, all traces of AM_1 of the form $t_N = \langle tr.alice.bob.N, score.good, result.ok, mi.small\rangle$ represent the trace t_a in the indistinguishability class of o. Therefore, to determine the entropy of the indistinguishability class of AM_1 *in the point of reference*, we take the probability of t_a (in the point of reference) to be the sum of the probabilities of all t_N (in a variant of AM_1).

The following lemma shows that $\widehat{\mathcal{CP}}_{ee}(QE, W, k, W, \mathrm{id}, k)$ indeed is a refined version of the basic confidentiality property of ensured entropy.

Lemma 38. *Let QE be a probabilistic linear process and let W be a set of channels of QE. Then the refined property* $\widehat{\mathcal{CP}}_{ee}(QE, W, k, W, \mathrm{id}, k)$ *for the point of reference* (W, id, k) *is equivalent to the basic confidentiality property of ensured entropy.*

9.3.3. Information flow refinement for ensured entropy

We finally define an information flow refinement relation for ensured entropy. The following Definition 39 specializes the mutual information preorder of Definition 32 for random variables derived from variants of two adversary models.

Definition 39 (*Information Flow Refinement Preserving Ensured Entropy*)**.** Let QE_a and QE_c be variants of adversary models such that $(QE_a, W_a) \overset{\nwarrow}{\sqsupseteq}_{R_{ca}} (QE_c, W_c)$. Then (QE_c, W_c) is an information flow refinement of (QE_a, W_a) preserving ensured entropy, written $(QE_a, W_a) \overset{k_a,k_c}{\precsim}_{R_{ca}} (QE_c, W_c)$ if the following condition holds:

$$\begin{aligned} &QE_a = (QE_c \backslash (W_c - W_a))[\![R_{ca}]\!]_D \wedge \\ &I((QE_a, k_a); (QE_c, k_c, W_c) | (QE_a, k_a, W_a)) = 0 \end{aligned} \tag{46}$$

Corollary 34 immediately implies the following one.

Corollary 40. The relation $(QE_a, W_a) \overset{k_a,k_c}{\precsim}_{R_{ca}} (QE_c, W_c)$ is a preorder.

The main result of this paper is the following Theorem 44. It states that $\overset{k_a,k_c}{\precsim}_{R_{ca}}$ preserves the confidentiality property of ensured entropy. It relies on Proposition 43, which states that $\overset{k_a,k_c}{\precsim}_{R_{ca}}$ preserves the confidentiality property of concealed behavior.

The following lemmas are essential to prove Proposition 43. Lemma 41 shows how to determine the conditional probability of an "abstract" trace of QE_a in terms of the probability of "concrete" traces of the process QE_c. In particular, it establishes that the set of those concrete traces is non-empty if the corresponding abstract trace has a positive probability.

Lemma 41. *Let* $(QE_a, W_a) \overset{k_a,k_c}{\precsim}_{R_{ca}} (QE_c, W_c)$*, and let the abstract observation* $o_a \in \mathrm{Obs}^{k_a}_{W_a}(QE_a)$ *and the concrete observation* $o_c \in \mathrm{Obs}^{k_c}_{W_c}(QE_c)$ *have a positive joined probability:* $\Pr((QE_c, k_c, W_c) = o_c, (QE_a, k_a, W_a) = o_a) > 0$*. Then for all* $r \in traces(QE_a)$*:*

$$\begin{aligned} &\Pr^{W_a,k_a}_{QE_a}(r|o_a) = \\ &\quad \textbf{let } T = \{t \in traces(QE_c) \downarrow k_c | \\ &\qquad\qquad t\restriction W_c = o_c \wedge r = R_{ca}(t\backslash(W_c - W_a))\}; \\ &\qquad O_{ac} = \{t \in traces(QE_c) \downarrow k_c | \\ &\qquad\qquad t\restriction W_c = o_c \wedge o_a = R_{ca}(t\backslash(W_c - W_a))\restriction W_a\} \bullet \\ &\quad \Pr^{k_c}_{QE_c}(T|O_{ac}) \end{aligned}$$

Furthermore, if $\Pr^{W_a,k_a}_{QE_a}(r|o_a) > 0$*, then the set* T *is nonempty.*

The following Lemma 42 is needed to show that $\overset{k_a,k_c}{\precsim}_{R_{ca}}$ preserves the coverage of a mask. It establishes that each re-abstracted indistinguishability class of QE_c is an indistinguishability class of QE_a. Because $W_a \subseteq W_c$, the abstract observation o_a associated with the re-abstracted indistinguishability class is the projection of the concrete observation o_c to W_a. The re-abstracted class contains the complete indistinguishability class of o_a, because Eq. (25) guarantees that the re-abstraction of QE_c produces exactly the traces of QE_a.

Lemma 42. *Let* $(QE_a, W_a) \overset{k_a,k_c}{\precsim}_{R_{ca}} (QE_c, W_c)$*. Then*

$$\forall o_c : \mathrm{Obs}^{k_c}_{W_c}(QE_c) \bullet J^{QE_c,k_c}_{W_c}(o_c)[\![R_{ca}]\!]^{W_a}_{W_c} = J^{QE_a,k_a}_{W_a}(o_c\restriction W_a)$$

Proposition 43 establishes that $\overset{k_a,k_c}{\precsim}_{R_{ca}}$ preserves the refined confidentiality property of concealed behavior. Together with Corollary 40, this means that $\overset{k_a,k_c}{\precsim}_{R_{ca}}$ is an information flow refinement order for concealed behavior.

Proposition 43 (*Preservation of Concealed Behavior*). *Let* (W_r, R_r, k_r) *be a point of reference for* QE_a *such that* $W_r \subseteq W_a$ *and* R_r *is the graph of a function, and let* $\mathcal{M}$ *be a mask in that point of reference. If* $(QE_a, W_a) \precsim^{k_a,k_c}_{\approx R_{ca}} (QE_c, W_c)$ *and* $\widehat{CP}_{cb}(QE_a, W_a, k_a, W_r, R_r, k_r)$ *then* $\widehat{CP}_{cb}(QE_c, W_c, k_c, W_r, R_{ca} \fatsemi R_r, k_r)$.

Theorem 44 establishes the main result of this paper, namely that $\precsim^{k_a,k_c}_{\approx R_{ca}}$ preserves ensured entropy, and with Corollary 40, that it is an information flow refinement property for ensured entropy.

Theorem 44 (*Preservation of Ensured Entropy*). *Let* (W_r, R_r, k_r) *be a point of reference for* QE_a *such that* $W_r \subseteq W_a$ *and* R_r *is the graph of a function, and let* $\mathcal{M}$ *be a mask in that point of reference. If* $(QE_a, W_a) \precsim^{k_a,k_c}_{\approx R_{ca}} (QE_c, W_c)$ *and* $\widehat{CP}_{ee}(QE_a, W_a, k_a, W_r, R_r, k_r)$ *then* $\widehat{CP}_{ee}(QE_c, W_c, k_c, W_r, R_{ca} \fatsemi R_r, k_r)$.

With Theorem 44 and Corollary 34, we finally know that the refinement relation $\mathcal{A} \sqsubseteq^{\precsim}_{\approx R_{ca}} \mathcal{C}$ is a preorder and preserves ensured entropy, i.e., it is a secure refinement order on adversary models for the property of ensured entropy.

Example 45. Example 31 showed that the mutual information between abstract behavior of BC_0 and concrete observations of BC_1^i is zero if the scores are determined stochastically independent of the size qualification on *mi*. Therefore, if BC_0 ensures the entropy bounds $\mathcal{H}$ for a mask $\mathcal{M}$ (Example 19), then Theorem 44 guarantees that BC_1^i also has that property (for a suitable point of reference).

At the level of adversary models, Proposition 28 makes a similar but slightly more complex statement: if the behavior refinement admits a secure process refinement, then ensured entropy is preserved under the behavior refinement. The entropy bounds we discussed in Example 19 are only preserved by the refinement, if the probabilistic choices in $System_1$, which are determined by the distributions $\mathcal{P}_b$ and $\mathcal{P}_s$, are such that the "abstracted" behavior (without the amount in transfers and without the events on *mi*) meets the entropy bounds. Under this condition, CPR ensures that the new observations (on *mi*) and the data refinement (on *tr*) do not compromise confidentiality.

10. Related work

10.1. Reactive simulatability

The system model introduced Section 4 has some similarities with the system model underlying reactive simulatability [34,26], which addresses the cryptographically secure implementation of "ideal" cryptographic protocols by "real" ones using cryptographic algorithms [35]. 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 reflect 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 high-resolution 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 [31] 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.

10.2. Possibilistic information flow properties

Possibilistic information flow properties require that a system that can perform a certain behavior can also perform certain other behaviors. Thus, such a property ensures that an observation does not immediately reveal the system behavior that caused the observation.

Mantel's thesis [9] presents a comprehensive overview over different types of possibilistic information flow properties. His *Modular Assembly Kit for Security Properties* (MAKS) identifies basic properties that can be conjoined to produce different information flow properties.

Information flow properties like the ones Mantel investigates are usually discussed using examples from the area of multi-level security. Confidentiality of events should be ensured regarding two types of users which are named *High* and *Low* after their corresponding security levels. User High is assumed to manipulate confidential information whereas user Low also has access to the system but should not be able to deduce certain types of confidential information.

The system model on which the definitions of information flow properties is based describes the possible traces over a given set of events, which are classified as inputs, outputs, and internal events. A *view* partitions the set of events into confidential, visible, and non-visible events. A *basic security predicate* is a closure property requiring certain traces to exist that differ from a given trace essentially in confidential events.

Standard information flow properties such as *generalized noninterference* [36] and *forward correctability* [37] formally capture the intuition that observing only the visible events of a system trace does not allow an adversary to gain information about the confidential events of that system trace. Such properties can be expressed as conjunctions of basic security predicates in MAKS.

There is a close but not an exact match between the concepts underlying basic security predicates and concealed behavior [33]. The most significant difference is the existential quantification in basic security predicates. This requires *at least one* alternative behavior τ' to be possible instead of a given behavior τ. If there is more than one candidate for τ' then any one suffices, and the remaining ones need not be possible system behaviors. A mask, however, prescribes *all* required alternatives to a given behavior. If a system produces one behavior in a member of a mask, it must be able to produce them all. The particular way Definition 13 defines coverage is adequate to define probabilistic confidentiality properties such as ensured entropy.

Another difference between the standard information flow properties and concealed behavior lies in the granularity of modeling secrets. A view classifies *events* as confidential, whereas concealed behavior keeps the differences between indistinguishable traces confidential. Those differences may not only concern the occurrence or non-occurrence of single events but also the relationship of several events. In Example 12, the requirement is not to keep scores as such confidential but the scores of account holders, i.e., the relation between an event *tr.from.to* and immediately following event *score.s*. In the context of a development process based on refinement, it may not always be possible to associate secrets with atomic events.

10.3. Probabilistic noninterference

Gray [38,1] worked out the conditions of probabilistic noninterference for a specific system model, which captures sequences of low and high events associated with internal system states. Gray's [1] condition of probabilistic noninterference requires that for all sequences[2] α_H, α_L of input events, and all sequences β_H, β_L of output events on high or low channels, respectively, that end at time $t-1$, and any low output event l_t at time t, the following condition holds:

$$\Pr(\alpha_L \cap \beta_L \cap \alpha_H \cap \beta_H) > 0 \Rightarrow \Pr(l_t|\alpha_L \cap \beta_L \cap \alpha_H \cap \beta_H) = \Pr(l_t|\alpha_L \cap \beta_L) \tag{47}$$

Gray [1] shows that Condition (47) implies that the mutual information between the high behavior preceding t and the low output l_t at time t given the low behavior preceding t is zero.

$$I(\alpha_H \cap \beta_H; l_t|\alpha_L \cap \beta_L) = 0 \tag{48}$$

[2] Strictly speaking, α_H, β_H, α_L, and β_L are sets of sequences of system behaviors with equal inputs or outputs on H or L, respectively.

By this equality, the channel capacity from high behavior to low outputs also is zero.

Similar to the possibilistic noninterference properties, the model underlying probabilistic noninterference does not consider the environment explicitly but models the machine only. The machine model is a probabilistic deterministic one, and the environment is supposed to resolve all external choices in the machine. Compared to Definition 9, probabilistic noninterference considers only a single machine implementation. Condition (47) quantifies over all possible environment behaviors, which corresponds to the universal quantification over environment variants in Definition 9. Similar to the parameter k of an adversary model, Gray [1] considers system behavior up to a time t, where the sequencing of events models the passing of time. In contrast to our system model, Gray's [1] model distinguishes inputs and outputs. Therefore, the following analysis of the relation between probabilistic noninterference and ensured entropy refers to a modified version of Condition (47), which does not distinguish input and output:

$$\Pr(\alpha\beta_L \cap \alpha\beta_H) > 0 \Rightarrow \Pr(l_t|\alpha\beta_L \cap \alpha\beta_H) = \Pr(l_t|\alpha\beta_L) \tag{49}$$

where $\alpha\beta_H$ and $\alpha\beta_L$ refer to the complete behaviors at the interfaces of *High* and *Low*, respectively, and l_t is a low event at time t. In our terminology, $\alpha\beta_H$ corresponds to a member of an indistinguishability class for the low behavior $\alpha\beta_L$, and $\alpha\beta_L$ is an observation on the adversary window. The time bound t and the trace bound k of an adversary model do not match exactly, because Gray [1] assumes that there are high and low events at each point in time, whereas our system model—following CSP—admits only single events at each place of a trace.

To capture Condition 49 by ensured entropy, we analyze the entropy of the high behavior given certain low behaviors. Ensured entropy considers only behaviors up to a given point in time, but it does not relate time t to the preceding point in time $t-1$. To establish that relation nevertheless, we express the entropy of high behavior given low behavior up to time t in terms of entropies preceding time t.

First, derive an equation for the entropy of $\alpha\beta_H$ given the low behavior up to time t.

$$\begin{aligned} H(\alpha\beta_H, l_t|\alpha\beta_L) &= \langle\text{chainrule}\rangle \\ &\quad H(\alpha\beta_H|\alpha\beta_L) + H(l_t|\alpha\beta_H, \alpha\beta_L) \qquad (50) \\ H(\alpha\beta_H, l_t|\alpha\beta_L) &= \langle\text{commutativity, chainrule}\rangle \\ &\quad H(l_t|\alpha\beta_L) + H(\alpha\beta_H|\alpha\beta_L \frown \langle l_t\rangle) \qquad (51) \end{aligned}$$

The two equations imply

$$H(\alpha\beta_H|\alpha\beta_L \frown \langle l_t\rangle) = H(\alpha\beta_H|\alpha\beta_L) + H(l_t|\alpha\beta_H, \alpha\beta_L) - H(l_t|\alpha\beta_L) \tag{52}$$

With Eq. (52), we relate the entropy of high behavior up to time t to the entropy up to time $t-1$:

$$\begin{aligned} & H(\alpha\beta_H \frown \langle h_t\rangle|\alpha\beta_L \frown \langle l_t\rangle) \qquad (53) \\ &= \langle\text{chainrule}\rangle \\ &\quad H(\alpha\beta_H|\alpha\beta_L \frown \langle l_t\rangle) + H(h_t|\alpha\beta_H, \alpha\beta_L \frown \langle l_t\rangle) \qquad (54) \\ &= \langle\text{Eq.(52)}\rangle \\ &\quad H(\alpha\beta_H|\alpha\beta_L) + H(l_t|\alpha\beta_H, \alpha\beta_L) - H(l_t|\alpha\beta_L) \\ &\quad + H(h_t|\alpha\beta_H, \alpha\beta_L \frown \langle l_t\rangle) \qquad (55) \\ &= \langle\text{Lemma2}\rangle \\ &\quad H(\alpha\beta_H|\alpha\beta_L) - I(l_t; \alpha\beta_H|\alpha\beta_L) + H(h_t|\alpha\beta_H, \alpha\beta_L \frown \langle l_t\rangle) \qquad (56) \end{aligned}$$

By Eq. (48), the mutual information $I(l_t; \alpha\beta_H|\alpha\beta_L)$ is zero if and only if Condition (49) is true. By Eq. (56), this is the case if and only if

$$H(\alpha\beta_H \frown \langle h_t\rangle|\alpha\beta_L \frown \langle l_t\rangle) = H(\alpha\beta_H|\alpha\beta_L) + H(h_t|\alpha\beta_H, \alpha\beta_L \frown \langle l_t\rangle) \tag{57}$$

Mutual information always is non-negative. Therefore, $H(\alpha\beta_H \frown \langle h_t\rangle | \alpha\beta_L \frown \langle l_t\rangle)$ is maximal if and only if Condition 49 holds.

In summary, Gray's [1] condition of probabilistic noninterference is similar to ensured entropy with an entropy bound that assigns maximal entropies (according to Eq. (57)) to all indistinguishability classes. To make this result a formal theorem, it would be necessary to elaborate the relationship between our system model and the one underlying probabilistic noninterference in detail, but we will not pursue this issue further here.

10.4. Discrete quantification of information flow

Lowe [39] quantifies information flow discretely, without referring to probabilities. Thus his work mediates between a possibilistic yes/no concept of information flow and one based on probabilistic information theory. Similar to our definition of a confidentiality property, his information flow property quantifies over the possible refinements of a specification, which are similar to the variants of an adversary model. In contrast to our view, he uses a pessimistic approximation and considers the worst case, i.e., maximal, flow of information produced by all variants. Our view is pessimistic with respect to the environment but takes an optimistic view with respect to the machine.

10.5. Refinement

Most research on the relationship of confidentiality properties and refinement addresses trace refinement and ways to avoid the refinement paradox. Few researchers discuss the effects of more complex kinds of refinement.

Graham-Cumming and Sanders [40] discuss the preservation of noninterference under data refinement. They specify systems using the specification language Z [28] 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 communicated data, 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.

Jacob [41,42] compares the confidentiality established by a CSP process relative to a *window*, which is a set of events. A system keeps more information confidential than another with respect to a given window if it allows for more traces given an observation on that window. Based on this concept, Jacob [41, Sect. 3.3, Def. 4] defines a *security ordering*: a system is at least as secure as another if it allows for more traces given any observation that is a possible observation of both systems. This ordering is similar to the preservation of concealed behavior in Proposition 43. However, instantiating Lemma 42 for $W_a = W_c$ reveals that our ordering actually is stronger than Jacob's [41] security ordering. The relationship $(QE_a, W) \precsim^k_{\mathrm{id}} (QE_c, W)$ entails that QE_c may have fewer indistinguishability classes than QE_a, which is similar to the security ordering, but it also requires that all indistinguishability classes of QE_a that have a counterpart in QE_c are preserved completely: in contrast to the security ordering, $\precsim^k_{\mathrm{id}}$ does not permit the indistinguishability classes of QE_c to be larger than the corresponding ones of QE_a. This is a consequence of the fact that QE_c is a behavior refinement of QE_a and that W_a is completely contained in W_c. Therefore, indistinguishability classes can only become smaller in a refinement (for the trivial retrieve relation id).

The contributions of this paper are based on earlier research on confidentiality-preserving refinement [24,25]. The main progress compared to that work is that we now have clarified the relationship between the refinement relation and the confidentiality property that it preserves: the entire framework is parameterized by the confidentiality property in question whereas the refinement relation of Heisel et al. [24] does not refer to a particular confidentiality property. Instead, it requires—in our terminology—the concrete model to ensure the maximally possible entropy of its indistinguishability classes, and that (the inverse of) the retrieve relation maps an abstract indistinguishability class either completely to a concrete one or not at all. The latter requirement corresponds to the possibilistic statement at the end of Lemma 41. Thus, the refinement relation of Heisel et al. [24] is a special case of ours for ensured entropy. Furthermore, the technical foundations, in particular concerning the probabilistic behavior of processes, have not been worked out in detail in those earlier publications.

11. Conclusions

We have established a formal framework for behavior refinement relations between adversary models that preserve confidentiality properties. Behavior refinement allows the concrete model to communicate different data on its channels than the abstract model. It also permits the concrete adversary window to allow for more observation channels than the abstract adversary window. In this way, behavior refinement of adversary models captures two crucial issues that come up in the transition from an abstract model to one that captures more detail of the real world: the particular choice of data type implementations may compromise security, and possibilities of an adversary to observe the system may become relevant from which the abstract model abstracts.

The concept of an information flow refinement relation is crucial to concisely capture the conditions under which a behavior refinement preserves a particular confidentiality property. By definition, an information flow refinement relation expresses a sufficient condition to preserve a (refined) basic confidentiality property at the level of variants of adversary models.

The definition of confidentiality-preserving refinement (CPR) relies on the information flow refinement relation and makes precise which variants of the two adversary models must be related by information flow refinement in order to preserve the confidentiality property at the level of nondeterministic adversary models. It is intellectually satisfying to see that the preservation condition of CPR in essence requires the information flow refinement relation and the re-abstraction relation to coincide.

The framework of CPR suggests a division of labor between security theoreticians and security engineers: the former are responsible to define a confidentiality property that is adequate to capture a relevant class of confidentiality requirements and find an information flow refinement relation for it; the latter must establish the conditions of CPR for each refinement step in a concrete system development.

Furthermore, we have investigated the information flow between variants of two adversary models in a behavior refinement. Informally, the idea is to allow no more "abstract" distinctions for an adversary who has access to "concrete" observations than for an adversary who only knows about "abstract" observations. Formally, the mutual information between "abstract behavior" and "concrete observations" must be zero. Although the primary purpose of that investigation is to establish general results that allow us to identify conditions for preserving ensured entropy in a refinement, it turns out that the analysis of the information flow between variants is quite involved. Proving the transitivity of that relation, which is a routine task in many other contexts, relies on a complex interaction of many facts about the modeling context that make Proposition 33 much more than a simple lemma on the way of establishing "ensured-entropy-preserving refinement."

Finally, Theorem 44 is the culmination point of our entire theory. To prove that the definition of CPR for ensured entropy indeed preserves that confidentiality property relies on many aspects of the theory that we have developed before. To define the information flow refinement relation for ensured entropy, we need to interpret the random variables for which we abstractly analyzed the mutual information in terms of the PCSP processes that make up an adversary model. The proof of Theorem 44 combines lemmas about the properties of indistinguishability classes with the general information-theoretic results established before. The theorem justifies to instantiate the framework for CPR and thus joins all results of this paper to form a single statement about preserving the probabilistic confidentiality property of ensured entropy under behavior refinement.

To the best of our knowledge, confidentiality-preserving refinement (and its predecessor [24,25]) is the first refinement relation that considers changing data types and adversary windows as well, and that preserves probabilistic confidentiality properties.

Conditions under which CPR is compositional are currently investigated. One cannot expect CPR to be unconditionally compositional. The task here is to find practically relevant sufficient conditions under which refining a part of a system yields a refinement of the complete system.

Appendix A. CSP notation

Process parameters

Ch	channel names
D	set of data transmitted over channels
Σ	set of all events over Ch and D, $\Sigma = \{c.x \mid c \in Ch \wedge x \in D\}$
αP	set of all channel names used in events of P

Process constructors	
$Stop$	the deadlocked process
$Skip$	the terminating process
div	the diverging process
$e \to P$	event prefixing
$P[\![S]\!]$	(relational) event renaming
$P[\![R]\!]_D$	data renaming, $P[\![R]\!]_D := P[\![\{(c.a, c.b) \mid c \in Ch \wedge a\ R\ b\}]\!]$
$P \backslash X$	hiding channels X in P
$P \restriction X$	hiding all channels in P but the ones in X
$P; Q$	sequential composition
$P \mid [X] \mid Q$	parallel composition of P and Q with synchronization over the events on the channels X
$P \Vdash_X Q$	asynchronous, unidirectional communication from P to Q via channels X
$P \Box Q$	external choice
$?x : X \to P(x)$	prefix choice
$P \sqcap Q$	internal choice
$\prod_{i:I} P(i)$	indexed internal choice
$P \downarrow k$	finite approximation of P diverging after k events
$P_p \oplus Q$	probabilistic choice with probability p of choosing P
$\bigoplus^{\mathcal{P}}_{x:X} P(x)$	indexed probabilistic choice
$P \uparrow$	the cone of P, i.e., the set of all Q refining P
$F \sqsubseteq P$	the probability that P refines F

References

[1] J.W. Gray, Toward a mathematical foundation for information flow security, Journal of Computer Security (1992) 255–294.
[2] C.E. Shannon, A mathematical theory of communication, The Bell System Technical Journal 27 (1948) 379–423, 623–656.
[3] J.A. Goguen, J. Meseguer, Security policies and security models, in: IEEE Symposium on Security and Privacy, IEEE Computer Society Press, 1982, pp. 11–20.
[4] J. Graham-Cumming, Laws of non-interference in CSP, Journal of Computer Security 2 (1993) 37–52.
[5] A. Zakinthinos, E.S. Lee, A general theory of security properties, in: Proceedings of IEEE Symposium on Security and Privacy, 1997, pp. 94–102.
[6] A.W. Roscoe, J.C.P. Woodcock, L. Wulf, Non-interference through determinism, in: D. Gollmann (Ed.), European Symposium on Research in Computer Security (ESORICS), LNCS, vol. 875, Springer-Verlag, 1994, pp. 33–53.
[7] J. McLean, A general theory of composition for a class of "possibilistic" properties, IEEE Transactions on Software Engineering 22 (1) (1996) 53–67.
[8] P.Y.A. Ryan, S.A. Schneider, Process algebra and non-interference, in: 12th IEEE Computer Security Foundations Workshop, IEEE Computer Society, 1999, pp. 214–227.
[9] H. Mantel, A Uniform Framework for the Formal Specification and Verification of Information Flow Security, Ph.D. Thesis, Universität des Saarlandes, 2003.
[10] T. Santen, A formal framework for confidentiality-preserving refinement, in: D. Gollmann, J. Meier, A. Sabelfeld (Eds.), Proceedings of 11th European Symposium On Research In Computer Security (ESORICS), LNCS, vol. 4189, Springer-Verlag, 2006, pp. 225–242.
[11] B. Liskov, J. Wing, A behavioral notion of subtyping, ACM Transactions on Programming Languages and Systems 16 (6) (1994) 1811–1841.
[12] B. Meyer, Applying "design by contract", IEEE Computer (1992) 40–51
[13] P. Behm, P. Benoit, A. Faivre, J.-M. Meynadier, Météor: a successful application of B in a large project, in: J. Wing, J. Woodcock, J. Davies (Eds.), FM'99—Formal Methods, vol. I, LNCS, vol. 1708, Springer-Verlag, 1999, pp. 369–387.
[14] C.A.R. Hoare, Proof of correctness of data representations, Acta Informatica 1 (1972) 271–281.
[15] J.-R. Abrial, The B-Book: Assigning Programs to Meanings, Cambridge University Press, 1996.
[16] J. Derrick, E. Boiten, Refinement in Z and Object-Z, Springer-Verlag, London, 2001.
[17] C.B. Jones, Systematic Software Development using VDM, second ed., Prentice Hall, 1990.
[18] A.W Roscoe, The Theory and Practice of Concurrency, Prentice Hall, 1998.

[19] J. Jacob, On the derivation of secure components, in: IEEE Symposium on Security and Privacy, IEEE Press, 1989, pp. 242–247.
[20] A.W. Roscoe, CSP and determinism in security modelling, in: Proceedings of IEEE Symposium on Security and Privacy, IEEE Computer Society Press, 1995, pp. 114–127.
[21] J. Jürjens, Secrecy-preserving refinement, in: J.N. Oliveira, P. Zave (Eds.), FME 2001: Formal Methods for Increasing Software Productivity, LNCS, vol. 2021, Springer-Verlag, 2001, pp. 135–152.
[22] H. Mantel, Preserving information flow properties under refinement, in: IEEE Symposium on Security and Privacy, IEEE Computer Society Press, 2001, pp. 78–91.
[23] P. Zave, M. Jackson, Four dark corners of requirements engineering, ACM Transactions on Software Engineering and Methodology 6 (1) (1997) 1–30.
[24] M. Heisel, A. Pfitzmann, T. Santen, Confidentiality-preserving refinement, in: 14th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, 2001, pp. 295–305.
[25] T. Santen, M. Heisel, A. Pfitzmann, Confidentiality-preserving refinement is compositional—sometimes, in: D. Gollmann, G. Karjoth, M. Waidner (Eds.), Computer Security—ESORICS 2002, LNCS, vol. 2502, Springer-Verlag, 2002, pp. 194–211.
[26] M. Backes, B. Pfitzmann, M. Waidner, Secure asynchronous reactive systems, IACR ePrint Archive, March 2004. Available from: <http://eprint.iacr.org/2004/082.ps>.
[27] C. Morgan, A. McIver, K. Seidel, J.W. Sanders, Refinement-oriented probability for CSP, Formal Aspects of Computing 8 (6) (1996) 617–647.
[28] J.M. Spivey, The Z Notation—A Reference Manual, second ed., Prentice Hall, 1992.
[29] D. MacKay, Information Theory, Inference, and Learning Algorithms, Cambridge University Press, 2003.
[30] C.A.R Hoare, Communicating Sequential Processes, Prentice Hall, 1985.
[31] R. Segala, N. Lynch, Probabilistic simulations for probabilistic processes, Nordic Journal of Computing 2 (2) (1995) 250–273.
[32] F. Ciesinski, M. Größer, On probabilistic computation tree logic, in: C. Baier, B.R. Haverkort, H. Hermanns, J.-P. Katoen, M. Siegle (Eds.), Validation of Stochastic Systems: A Guide to Current Research, LNCS, vol. 2925, Springer-Verlag, 2004, pp. 147–188.
[33] T. Santen, Probabilistic confidentiality properties based on indistinguishability, in: H. Federrath (Ed.), Proc. Sicherheit 2005—Schutz und Zuverlässigkeit, Lecture Notes in Informatics, Gesellschaft für Informatik, 2005, pp. 113–124.
[34] B. Pfitzmann, M. Waidner, A model for asynchronous reactive systems and its application to secure message transmission, in: IEEE Symposium on Security and Privacy, IEEE Computer Society, 2001, pp. 184–201.
[35] M. Backes, B. Pfitzmann, M. Waidner, A composable cryptographic library with nested operations (extended abstract), in: Proceeding of 10th ACM Conference on Computer and Communications Security, 2003, pp. 220–230.
[36] J. McLean, A general theroy of composition for trace sets closed under selective interleaving functions, in: Proceedings of IEEE Symposium on Research in Security and Privacy, 1994, pp. 73–93.
[37] D.M. Johnson, F.J. Thayer, Security and the composition of machines, in: Proceedings of IEEE Computer Security Foundations Workshop, 1988, pp. 72–89.
[38] J.W. Gray, Toward a mathematical foundation for information flow security, in: Proceedings of IEEE Symposium on Security and Privacy, 1991, pp. 21–34.
[39] G. Lowe, Quantifying information flow, in: 15th IEEE Computer Security Foundations Workshop, IEEE Computer Society, 2002, pp. 18–31.
[40] J. Graham-Cumming, J.W. Sanders, On the refinement of non-interference, in: 9th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, 1991, pp. 35–42.
[41] J. Jacob, Security specifications, in: IEEE Symposium on Security and Privacy, IEEE Press, 1988, pp. 14–23.
[42] J. Jacob, Basic theorems about security, Journal of Computer Security 1 (1992) 385–411.