Innovations in Computer Science 2011

Abstract Cryptography

Ueli Maurer[1] Renato Renner[2]
[1]Department of Computer Science, ETH Zurich, Switzerland
[2]Institute for Theoretical Physics, ETH Zurich, Switzerland
maurer@inf.ethz.ch renner@phys.ethz.ch

Abstract: In the spirit of algebraic abstraction, this paper advocates the definition and use of higher levels of abstraction in cryptography (and beyond). If contrasted with the standard bottom-up approach to defining models of computation, algorithms, complexity, efficiency, and then security of cryptographic schemes, our approach is top-down and axiomatic, where lower abstraction levels inherit the definitions and theorems (e.g. a composition theorem) from the higher level, but the definition or concretization of low levels is not required for proving theorems at the higher levels. The goal is to strive for simpler definitions, higher generality of results, simpler proofs, improved elegance, possibly better didactic suitability, and to derive new insights from the abstract viewpoint.
In particular, we propose a general framework for defining and proving that a system satisfying an (abstract or ideal) specification is constructed from some systems satisfying certain (concrete or real) specifications. This puts the well-known "ideal-world real-world" paradigm on a new theoretical foundation, applicable in various cryptographic settings. Existing frameworks for proving composable security can be explained as special cases of our framework, thereby allowing to distinguish between relevant and less relevant aspects of the underlying technical definitions and to prove a single common composition theorem.
Some properties of our framework are as follows. It is independent of particular models of computation, communication, and adversary behavior. It can be instantiated in many different ways, for example to arrive at different notions of security or of efficiency and infeasibility. It can precisely capture settings with no central adversary where entities have potentially conflicting goals (e.g. a coercion scenario). The relation between the ideal and the real setting is tight, via an isomorphism notion for settings. The (desired) asymmetry between real and ideal is captured in a formal abstraction notion (the ideal setting is an abstraction of the real setting). A main theorem states that such an abstraction statement can be proved by using local (as opposed to monolithic) simulators.

Keywords: foundations of cryptography, composable security.

1 Introduction

1.1 The quest for simplicity and abstraction

In every mathematical discipline one tries to identify the key concepts and to formalize them in an abstract manner. Abstraction means to eliminate irrelevant details from consideration, thereby focusing only on the relevant aspects of a problem or context. The purpose of abstraction is to provide, at the same time, simpler definitions, higher generality of results, simpler proofs, improved elegance, better didactic suitability, and, perhaps most importantly, new insights.

In many contexts, the highest achievable level of abstraction, once identified, appears natural and stable. For example, the natural mathematical concepts of a relation, a function, a graph, a group, a field, or a vector space capture exactly, in a minimal manner, the relevant notions.

In contrast, in cryptography (and in other areas of computer science), definitions, theorems, and proofs are generally highly technical and have substantial complexity due to the technical artifacts of the particular model (e.g. defining the computational model via Turing machines and communication via tapes, using asymptotic definitions of systems and protocols, defining efficiency as polynomial-time, using a particular adversarial model, etc.). This is often considered unavoidable since, undoubtedly, definitions must be precise. Nevertheless, more abstraction seems possible. For example the notion of polynomial-time,

even though universally accepted and reasonable, can probably not be seen as the ultimate (and only) definition and abstraction of efficiency and of feasibility.

In view of this, it seems desirable to make cryptographic definitions in an abstract way, for example non-asymptotic, and to leave the technical aspects to a lower level of abstraction where systems and their composition are defined, and where unavoidable artifacts (like asymptotics, required for a notion of composability of efficient systems or algorithms) are introduced. This allows for simplicity (e.g. non-asymptotic definitions and proofs) and generality (e.g. simultaneous treatment of different security notions, like information-theoretic and computational).

1.2 Contributions and outline of this paper

The paper makes several independent contributions, some of which are explained in more detail in the following subsections. This section also provides an outline of the paper.

- We propose (see Section 1.4) as a new paradigm for developing a theory of cryptography (and beyond) to define levels of abstraction top-down rather than bottom-up. We also propose a hierarchy of levels of abstraction in a general theory of systems (see Section 1.5), of which we only formalize and use the top level in this paper.
- We define and investigate (in Section 3) the most abstract notion of a *reduction*, which captures that realizing (or implementing) a certain object (or module) is reduced to realizing some other (simpler) objects.[1] When used iteratively, reducing the simple objects to yet simpler objects, this is called *step-wise refinement*, a central paradigm in any constructive discipline. A reduction is *composable* (Definition 7) if it satisfies some simple natural properties. Composable reductions allow us to capture formally the *soundness* of the step-wise refinement paradigm.
- We define the general notion of a so-called *n-choice setting* (see Section 4) which models any setting where some number n of independent choices (each from a specific domain) can be made, and where for every fixed list of choices an *effect* is defined.[2] Formally, such a choice setting is a function from n choice domains to an effect space. A choice can consist of fixing an entire (interactive) strategy.[3] We refer to Section 1.3 for a discussion of what this could mean in cryptography.
  Moreover, we introduce the notion of an isomorphism between such $n$choice settings. Roughly speaking, two n-choice settings are isomorphic if each choice made in one has a corresponding choice in the other, in such a way that the effects remain the same. This equivalence is the strongest possible and holds no matter why the particular choices are made (e.g. because parties making the choices are honest, dishonest, corrupted, collude, or are coerced).
- We consider the notion of a set of choice settings, called a *specification* (see Section 5), and define that a specification $\mathcal{S}$ is an *abstraction* of another specification $\mathcal{R}$ if, roughly, every element of $\mathcal{R}$ has an isomorphic element in $\mathcal{S}$.[4] Stated differently, an (ideal) specification $\mathcal{S}$ is an abstraction of another (real) specification $\mathcal{R}$ if any element of $\mathcal{R}$ also meets the specification $\mathcal{S}$, which implies that *the real specification inherits every property of the ideal specification.*
  We also discuss the concept of an ϵ-region $\mathcal{R}^\epsilon$ of a specification $\mathcal{R}$, allowing to define approximate (within ϵ) abstraction.
  Theorem 1 states that abstraction (or approximate abstraction) is a composable reduction.
- Section 6 presents a *theory of abstract systems*, which is also of independent interest. At this highest level of abstraction, a system is an abstract object with interfaces. Two systems can be composed (similar to an algebraic composition operation) into a single system by connecting one interface from each system.
  At this abstract level we define the notion of a *cryptographic algebra*, consisting of a set of *resource* systems, a set of (so-called) *converter*

[1] This includes as a special case the classical notion of a reduction in complexity theory, but also various types of reductions in cryptography, or any other constructive discipline.

[2] One can think of the choices as being made by parties, where one party can also make several of the choices. Also, the choices of several parties modeled to act consistently can me modeled as a single choice. The notion of an n-choice setting is more general than the notion of an n-party game in game theory where the effect is described specifically by utility functions of the parties making the choices.

[3] Modeling such a strategy as one choice or as many choices is both possible and corresponds, for example, to modeling whether or not a party can (or is assumed to) change its goals during time, for example due to coercion.

[4] This abstraction notion is discussed here only in our specific context but is actually applicable in more general contexts where an isomorphism (or equivalence) notion is defined on a set.

systems, and a set of *distinguisher* systems, satisfying certain axioms. Resources (e.g. a communication channel or a commitment functionality) can be understood as providing (at their interfaces) certain functionalities to parties connected to the resource. A converter system attached to an interface of a resource corresponds to a program (or protocol engine, e.g. an encryption engine) transforming the interface into a different interface (see Section 1.3). A (local) simulator is also a converter.
Cryptographic algebras can be instantiated in different ways, corresponding to different security notions. For example, if all three system classes are arbitrary (no computational restriction), this results in the notion of information-theoretic security. If the classes are restricted to efficiently implementable systems, for an arbitrary composable notion of efficiency, then this results in the notion of computational security. Other notions also exist.

- Section 7 brings together all the previously defined concepts: n-choice settings correspond to n-interface resources, choices correspond to converters, an effect is again an n-interface resources (resulting when the converters are attached), and a pseudo-metric on the set of resource systems is defined as the maximal distinguishing advantage of a certain distinguisher class.
We define an important class of (resource) specifications, called $filtered$ specifications. Roughly, a filtered specification is described by a resource system and, for each interface, a filter (a converter), where the meaning is that the party connected to the interface is *guaranteed* to have filtered access to the resource but that it is also *possible* (but not guaranteed) that it has more powerful access, up to possibly unfiltered access. This allows to capture exactly what it means that a system satisfying an (ideal) specification is constructed from a list of systems satisfying certain (real) specifications, putting the well-known "ideal-world real-world" paradigm on an exact theoretical foundation, applicable in any constructive setting (including, for example, channel coding in information theory).
A main theorem (Theorem 2) of this part shows how to prove abstraction statements by invoking the concept of local simulators.

Existing frameworks, such as Canetti's UC framework [3], can be explained as special cases of our framework. For example, the different composition theorems can be seen as implications of our general composition theorem.

Our theory can explain a large number of results in a unified and generalized manner. For example, in Appendix C we prove by algebraic equations a generalization of the celebrated impossibility results of realizing UC secure commitments [5]. In Appendix D we prove the impossibility result of Canetti et al. [6] of realizing a random oracle.

The paper also makes a number of observations following from the abstract viewpoint. Some implications on the "ideal-world real-world" paradigm are discussed in Section 1.6, and possible implications on conventional cryptography are discussed in Section 1.7.

1.3 A cryptographic example

We give a first high-level example of how the concepts of the previous section could be interpreted in a concrete cryptographic context. We also refer to [12] for a discussion of concrete examples of resources and converters in a cryptographic context.

Consider a resource system consisting of an authenticated (but not confidential) channel from A to B, accessible to an eavesdropper E, and (in parallel) a system providing a secret key to A and B (and nothing to E). A choice for A is to apply (at its interface of the channel) a converter system, with an outside and an inside interface, that fetches a key at the inside interface, accepts messages at the outside interface, encrypts each message with the key, and sends it at the inside interface as input to the channel. A choice for B is to apply a similar decryption converter. We want to argue that a secure channel (one that does not leak the transmitted message, only its length) is an *abstraction of* the above resource (authenticated channel and key).

Here we have assumed somehow that A and B indeed make these choices, and we make no statement otherwise. However, in a setting where A's (and B's) other choices should also be considered (because we want to make a statement even if A deviates from the normal behavior), then it will turn out that secure channel which is the abstraction has the special feature that A and B could potentially leak the message to E (e.g. by pressing a certain button). This captures the well-known fact that by encrypting a message, one may have the capability of convincing E about the

message sent. This is usually captured by terms like coercibility, but in our view it is just an interpretation of properties of the abstracted resource.

1.4 Levels of abstraction: bottom-up vs. top-down

The traditional approach in theoretical computer science, and more specifically in complexity theory and cryptography, is bottom-up. One first defines (at a low level) a computational model (e.g. a Turing machine or a circuit), based on which one defines the concept of an algorithm for the model and a communication model (e.g. based on tapes). One then defines a complexity notion for an algorithm (e.g. the number of steps), and then a notion of efficiency (e.g. polynomial in some parameter(s), in an asymptotic sense). Finally, based on all these notions, one defines the security of a cryptosystem, typically as the infeasibility of a specific game.

This approach is based on the view that without defining all these low-level concepts, one can not state precise definitions of higher-level concepts, let alone prove theorems. However, while this established approach is perfectly sound in principle, it is perhaps fair to say that many papers using this approach fall short of being completely precise and are in some cases even wrong in technical details, thus missing the promise of the bottom-up approach of being precise down to the last detail.

The paradigm shift we propose is to use a top-down approach. In order the state definitions and develop a theory, one starts from the other end, the highest possible level of abstraction, and proceeds downwards, introducing in each new lower level only the minimal necessary specializations of that level necessary for expressing what one wants to capture.

In the context of this paper, the concept of a reduction can be seen as standing at the highest level of abstraction in our paper. The notion of an n-choice setting (including the concepts of isomorphism, specification, and abstraction of specifications) is independent of the reduction concept, but in the way it is used here it appears at a next lower level. The abstraction hierarchy of a theory of systems (Section 1.5) can also be seen as independent, but again in the way it is used here it can also be seen as constituting the next lower levels.

1.5 Abstraction levels of a system theory

We define the following abstraction levels of a theory of systems, to be used beyond the scope of this paper, for example in complexity theory.

- **Level 1** captures the most general notion of a *system* and of the *composition* of systems. The composition laws are described by simple algebraic rules.
- **Level 2** captures the most general notion of a *discrete* system, the typical level at which cryptographic systems are considered. We need a theory of discrete systems corresponding to an extension of Maurer's random system framework [8] for single-interface systems to multiple-interface systems and to the composition of systems (work in progress). The language in which discrete systems are described (e.g. text, pseudo-code, or conditional probability distributions) is not directly relevant.
- **Level 3** defines the *implementation* of systems. At this level one can define (still) abstract complexity and efficiency notions (e.g. polynomial-time). This level is not required if one considers only information-theoretic security.

A lower level can define the particular computational model, the particular complexity or cost function for that model, etc. A still lower level can define timing aspects, and a further level can define physical aspects, including for example side channels in cryptography.

It is important to point out that theorems proved at a certain (high) level of abstraction are completely precise (as they are mathematical theorems). This is true without instantiations of the lower levels, which is exactly the point of abstraction. The definitions and theorems are inherited by the lower levels, provided (of course) that the lower levels satisfy the postulated properties or axioms of the higher levels. It is hence strictly more desirable to prove theorems at higher levels of abstraction; nothing is lost by doing this, and certainly not precision.

For example, if one proves a theorem at level 1 for an algebra of composable systems, then this theorem holds in particular for a specific theory of efficiently implementable discrete systems where efficiency is defined in a composable manner, and it holds also in the more specialized setting where efficiency is defined as (some form of) polynomial-time. A main benefit from

such an abstract treatment is that concrete results (e.g. the proof of a composition theorem or an impossibility proof) become at the same time substantially simpler and more general.

This paper deals with level 1, while examples also live at level 2, without this level being formalized as a full formalization of level 2 would allow. But our level of formality (e.g. in describing the specification of a commitment functionality) is comparable to the formality of papers written in the traditional style, and it is sufficient to be precise.

1.6 Implications on the "ideal-world real-world" paradigm and related Work

The so-called "ideal-world real-world" paradigm, as explained in the frameworks of Canetti [3] (universal composability) and of Backes, Pfitzmann and Waidner [2, 13] (reactive simulatability) is of paramount importance in cryptography. It is also used in other frameworks, for example indifferentiability theory by Maurer, Renner, and Holenstein [11], the generalized UC framework of Canetti, Dodis, Pass, and Walfish [4], collusion-free computation by Alwen, shelat, and Visconti [1], or for the notion of splittability [14]. These frameworks can be explained as special cases of our framework.

In these frameworks, the basic and ingenious idea [3, 13] is to consider an ideal system (often called a functionality) capturing the goal one wants to securely realize, when given a complete asynchronous network and possibly a set-up, using a protocol specifying what the parties have to do. The definition of what it means to "securely realize" involves an adversary who can corrupt certain parties, and captures the idea that whatever the adversary can achieve in the ideal world he could also achieve in the real world. This is made precise by means of a so-called *simulator*, an ingenious concept introduced by Goldwasser, Micali, and Rackoff [7] to define zero-knowledge protocols.

Despite their success, the current simulation-based frameworks ([1-4, 11, 13] and others) have several limitations and drawbacks, in addition to the above mentioned lack of abstraction. Some of the limitations, which are eliminated by our framework, are briefly discussed below.

1. The concept of a (central) adversary is essential in most previous frameworks. However, this makes it impossible to model settings where parties have conflicting goals but are *not* corrupted by the same adversary.[5] In contrast, in our framework there is no adversary (but such a setting can be modeled as a special case). A consequence of our approach is that in our framework, there is no monolithic simulator, but a (local) simulator for every party.

2. The communication model (the network), and other aspects, are an intrinsic part of the model. This makes it impossible to consider the *absence* of certain channels, as is for example required in the context of collusion-free computation [1] (a notion that is not composable) and in other contexts, or other types of network models. In contrast, our theory of reductions between resources makes every resource explicit, including a possibly available network.

3. The security definition (e.g. computational or information-theoretic security) is usually hard-coded into the model.

4. The ideal functionality is a fixed system. This means that a certain action is either guaranteed to be available or guaranteed to be unavailable to a certain party. In contrast, a specification in our framework models that certain choices are guaranteed, while others might be available. This allows to model notions like incoercibility or coercibility naturally as a property of the ideal specification, not as a separate security notion [15].

5. The complexity notions of efficiency (what honest parties can do) and feasibility (what an adversary is assumed to be able to do) are both defined in a fixed manner as (some form of) polynomial-time. It is not even clear that one could (and should) distinguish between these notions. We point out that these notions are distinct and can be instantiated arbitrarily as composable notions, for example efficient as polynomial-time and feasible as some form of subexponential time.

6. Security is defined via the notion of a (monolithic) simulator. In our framework, simulators are local (one simulator per choice domain), and they are not part of the *definition* of what it means to securely realize a certain functionality, but it is "only" a tool in the proof.

[5] A first paper giving up the notion of a central adversary was [1].

1.7 Implications on conventional cryptography

Security definitions of conventional cryptographic primitives can (and should?!) be made as constructive statements rather than by describing attack games. This approach, which can be derived as a special case of our framework, was called *constructive cryptography* in [10] and is used in [12] to investigate what encryption achieves when used on an insecure channel as in the authenticate-then-encrypt paradigm.

The problem with attack-based definitions is that the security of the composition of several individually secure schemes may not be clear. In a constructive definition, composability is guaranteed. For example, a message authentication scheme can be defined to be secure if it *constructs* (or realizes) an authenticated communication channel from a resource consisting of an insecure communication channel and a secret key, and a symmetric encryption scheme can be defined to be secure if it constructs a secure communication channel from an authenticated communication channel and a secret key. The general composition theorem implies that the combination of a secure MAC and secure encryption scheme constructs a secure channel from an insecure channel and two secret keys (which can be constructed from a single secret key using a pseudo-random generator).

1.8 Limitations and future work

This paper describes the basic theory of abstract cryptography, without providing many examples. We are aware that it may not be easy to judge the usefulness of the framework without seeing a list of convincing examples, but we hope that the few examples mentioned here give at least an idea and a taste. Also, the comparison with prior work is not yet worked out in detail, and many references relevant in such a comparison are still missing. It is the goal of future papers, hopefully also by other authors, to formalize lower levels of abstraction, to discuss new application areas of this theory, to work out examples, and to establish the relation to (and generalizations of) prior work at a technical level. We welcome feedback on this first (primarily theoretical) paper on the subject and pointers to related work.

2 Preliminaries

In the following two subsections, we briefly review some well known facts about relations, equivalence relations, and pseudo-metrics.

2.1 Relations

A *relation ρ from a set A to a set B* (also called an *(A,B)-relation*) is a subset of $A \times B$. An (A,A)-relation is also called a *relation on A*. If (a,b) is in the relation r, i.e., $(a,b) \in \rho$, one writes $a\ r\ b$ or, alternatively, $r(a,b)$. An (A,B)-relation is also an (A',B')-relation for any $A' \supseteq A$ and $B' \supseteq B$.

Definition 1. An (A,B)-relation ρ is called *complete* if it has full domain (i.e., $\forall a \in A\ \exists b \in B:\ a\ \rho\ b$) and full range (i.e., $\forall b \in B\ \exists a \in A:\ a\ \rho\ b$).

A function $A \to B$ corresponds to the (A,B)-relation $\{(a, f(a)) :\ a \in A\}$. Note that a relation is complete if and only if it contains (as subsets) a function $A \to B$ and a function $B \to A$.

Since relations are sets, all operations defined for sets (e.g. $\cup$ and $\cap$) are also defined for relations. In addition, one can define the composition operation $\circ$ and a direct product $\times$ for relations. The *composition* of an (A,B)-relation ρ and a (B,C)-relation σ is the (A,C)-relation $\rho \circ \sigma$ defined by

$$a\ (\rho \circ \sigma)\ c\ :\Longleftrightarrow\ \exists b \in B:\ (a\ \rho\ b)\ \wedge\ (b\ \sigma\ c).$$

We usually write $\rho\sigma$ instead of $\rho \circ \sigma$. The *inverse* of an (A,B)-relation ρ is the (B,A)-relation ρ^{-1} defined by $b\rho^{-1}a \iff a\rho b$, and we have $(\rho\sigma)^{-1} = \sigma^{-1}\rho^{-1}$. The *direct product* of an (A_1,B_1)-relation ρ and an (A_2,B_2)-relation σ is the $(A_1 \times A_2, B_1 \times B_2)$-relation $\rho \times \sigma$ defined by

$$(a_1,a_2)\ (\rho \times \sigma)\ (b_1,b_2)\ :\Longleftrightarrow\ (a_1\ \rho\ b_1)\ \wedge\ (a_2\ \sigma\ b_2).$$

The operations $\circ$ and $\times$ are both associative.

The following lemma is a direct consequence of the distributive law for relations,

$$(\rho \cup \sigma) \times (\rho' \cup \sigma') = (\rho\times\rho') \cup (\sigma\times\sigma') \cup (\sigma\times\rho') \cup (\rho\times\sigma').$$

Lemma 1. *For relations $\rho_1, \ldots, \rho_n$ and $\sigma_1, \ldots, \sigma_n$ we have*[6]

$$\bigotimes_{i=1}^{n} (\rho_i \cup \sigma_i) = \bigcup_{M \subseteq \{1,\ldots,n\}} \left(\bigotimes_{j \in M} \rho_j \times \bigotimes_{j \notin M} \sigma_j \right).$$

[6] Note that the terms on the right hand side of the equality should be ordered according to their indices i. We hope that our abbreviated notation does not cause confusion.

2.2 Pseudo-metrics and embeddings

A *pseudo-metric*[7] on a set Ω is a function $\delta : \Omega \times \Omega \to \mathbb{R}^+$ such that for all $a, b, c \in \Omega$

(i) $\delta(a, a) = 0$
(ii) $\delta(a, b) = \delta(b, a)$ (symmetry)
(iii) $\delta(a, c) \leq \delta(a, b) + \delta(b, c)$ (triangle inequality).

Note that any pseudo-metric δ on a set Ω induces an equivalence relation Ω,:

$$a \simeq b \quad :\Longleftrightarrow \quad \delta(a, b) = 0.$$

If a set Ω is equipped with a pseudo-metric we can define an induced metric on the set of functions with codomain Ω.

Definition 2. Let $\mathcal{F}$ be a set of functions with codomain Ω and let δ be a pseudo-metric on Ω. We define the *induced metric* on $\mathcal{F}$ by

$$\delta(f, g) := \sup_{x \in \mathcal{X}} \delta\big(f(x), g(x)\big) \ .$$

whenever $f, g \in \mathcal{F}$ have identical domains $\mathcal{X}$, and $\delta(f, g) = \infty$ otherwise.

It is often natural to consider a pair of elements of a set Ω as an element of Ω itself. For example, a pair of random variables can naturally be considered as a single random variable. Formally, we can consider the set Ω to be equipped with an embedding (or concatenation) operation Ω mapping $\Omega \times \Omega$ to Ω.[8]

Definition 3. A pseudo-metric δ for a set Ω with operation $\|$ is called $\|$*-non-expanding* if $\delta(a\|a', b\|b') \leq \delta(a, b) + \delta(a', b')$ for all $a, a', b, b' \in \Omega$.[9]

As an example, let Ω be the set of probability distributions, with $P\|P'$ defined as the product distribution between P and P'. Then the statistical distance is $\|$-non-expanding.

2.3 Isomorphisms of functions

Let ρ be an (A, B)-relation and let σ be an (A', B')-relation. These relations naturally induce a relation, denoted (ρ/σ), between functions $A \to A'$ and $B \to B'$ as follows. Two functions, $f : A \to A'$ and $g : B \to B'$, are related, denoted $f \ (\rho/\sigma) \ g$, if and only if all related inputs a and b are mapped to related outputs. That is, formally,

$$f \ (\rho/\sigma) \ g \quad :\Longleftrightarrow \quad a \ \rho \ b \ \rightarrow \ f(a) \ \sigma \ g(b) \ . \tag{1}$$

We are interested in the particular case where all functions have the same codomain Ω and where σ is an equivalence relation on Ω, which we denote by $\simeq$. Any (A, B)-relation ρ then naturally gives rise to the relation $(\rho/\simeq)$ between functions with domains A and B, respectively. If ρ is complete then the relation $(\rho/\simeq)$ can be seen as an isomorphism between functions. This isomorphism is denoted by $\overset{\rho}{\simeq}$ and will play a crucial role in this work (see Section 4).

Definition 4. Let ρ be a complete relation between two sets A and B, and let Ω be a set equipped with an equivalence relation $\simeq$. We say that two functions, $f : A \to \Omega$ and $g : B \to \Omega$, are *isomorphic (via ρ and relative to $\simeq$)*, denoted $f \overset{\rho}{\simeq} g$ if and only if all inputs a and b related by ρ are mapped to equivalent outputs, i.e.,

$$f \overset{\rho}{\simeq} g \quad :\Longleftrightarrow \quad \big(a \ \rho \ b \ \rightarrow \ f(a) \ \simeq \ g(b)\big) \ .$$

Often it is convenient to think of the equivalence relation $\simeq$ as being simply equality (i.e., $=$). Indeed, an alternative (and correct) view would be to consider equivalence classes (of $\simeq$) and to use the symbol "=".

Furthermore, note that the images $f(A)$ and $g(B)$ of two functions are identical if and only if there exists a complete (A, B)-relation ρ such that $f \overset{\rho}{=} g$.

3 Reductions: formalizing step-wise refinement

A central paradigm in any constructive discipline, for example in software design, the construction of machines, and also in security engineering and cryptography, is to construct a complex system from simpler component systems, which each may consist of yet simpler component systems, and so on. This important iterative construction paradigm is sometimes called *step-wise refinement*.

One can describe the overall construction as a tree where each inner (non-leaf) node is labeled by a component S and a *constructor* α describing how component S is obtained from components, say R_1 and R_2,

[7] A *metric* is a pseudo-metric for which $\delta(a, b) = 0$ implies $a = b$.

[8] Typically one would extend this consideration to lists (e.g. lists of random variables), in which case one assumes Ω to be associative. However, while thinking of Ω as an associative mapping is helpful and justified, in our treatment we do not have to make this assumption.

[9] In some contexts $\|$-non-expanding is called *stabilized*. Note that the condition is equivalent to $\delta(a\|c, b\|c) \leq \delta(a, b)$ and $\delta(c\|a, c\|b) \leq \delta(a, b)$ for all $a, b, c \in \Omega$, which could be used as an alternative definition of $\|$-non-expanding.

at the children nodes. Then we write

$$R_1 \| R_2 \xrightarrow{\alpha} S.$$

This can be generalized to nodes with more than two children.[10]

There are two ways to think about such a construction step: top-down or bottom-up. Depending on the view one can either say that S is *reduced* via use of α to R_1 and R_2, or that S is *realized from* R_1 and R_2 via α.

We mention a few instantiations of the reduction concept. For example, the purpose of a software component is to construct a system S from given library components $R_1, \ldots, R_n$ such that if these components are correct, then S is correct. Similarly, the purpose of applying error-correcting codes can be seen as constructing a reliable channel S from an unreliable (error-prone) channel R. In the context of cryptography, the purpose of encryption can be seen as constructing (according to a simulation-based notion) a secure communication channel S from two components, R_1 and R_2, where one is an authenticated channel and the other a secret key R_2.

In the following, we formalize these notions.

Definition 5. A *component set* is a set Ω equipped with a (parallel composition) operation, denoted $\|$. A *constructor set* Γ is a set equipped with a (serial composition) operation $\circ$, a (parallel composition) operation $|$, and a special (neutral) element id.

Definition 6. A *reduction* for a component set Ω and a constructor set Γ is a subset of $\Omega \times \Gamma \times \Omega$. A reduction is often denoted by an arrow, for example$\longrightarrow$, as follows: If (R, α, S) is in the reduction, then we write $R \xrightarrow{\alpha} S$ and say that S can be *reduced to* R by α or, equivalently, that S *can be realized (or constructed) from* R *by* α. We write $R \longrightarrow S$ if there exists an $\alpha \in \Gamma$ such that $R \xrightarrow{\alpha} S$.

A reduction $\longrightarrow$ for a component set Ω and a constructor set Γ can equivalently be interpreted as a collection $\{\xrightarrow{\alpha}\}_{\alpha\in\Gamma}$ of relations on Ω, indexed by elements of Γ.

We now define natural composability properties of a reduction.

[10] If there are d children $R_1, \ldots, R_d$, one can write $R_1 \| \cdots \| R_d \xrightarrow{\alpha} S$. If $\|$ is not associative, appropriate parentheses are required in the expression $R_1 \| \cdots \| R_d$.

Definition 7. A reduction $\longrightarrow$ for Ω and Γ is called *serially composable* if the following properties hold for all $R, S, T \in \Omega$ and $\alpha, \beta \in \Gamma$:

(i) $R \xrightarrow{\alpha} S \ \wedge \ S \xrightarrow{\beta} T \implies R \xrightarrow{\alpha\circ\beta} T$
(ii) $R \xrightarrow{\mathsf{id}} R$.
Furthermore, Γ is called *context-insensitive* if
(iii) $R \xrightarrow{\alpha} S \implies R\|T \xrightarrow{\alpha|\mathsf{id}} S\|T \ \wedge \ T\|R \xrightarrow{\mathsf{id}|\alpha} T\|S.$

A reduction that is both serially composable and context-insensitive is called *generally composable* (or just *composable*).

Property (i) implies transitivity of the relation $\longrightarrow$ and Property (iii) states that $R \xrightarrow{\alpha} S$ remains true independently of the environment or context, i.e., independently of which systems are available in parallel. Properties (i) to (iii) together imply that the reduction $\longrightarrow$ is *parallelly composable*, in the following sense:[11]

$$R \xrightarrow{\alpha} S \ \wedge \ R' \xrightarrow{\alpha'} S' \implies R\|R' \xrightarrow{\beta} S\|S',$$

where $\beta = (\alpha|\mathsf{id})(\mathsf{id}|\alpha')$.

It is intuitively clear that general composability is sufficient (and necessary) for the step-wise refinement paradigm to work in the following sense. Given a tree representing a construction as described above where the relation $R_1\|R_2 \xrightarrow{\alpha} S$ holds for any node S and its children R_1 and R_2, the relation also holds between the root and the leaves. We refer to Appendix A for a formal version of this claim.

4 Choice settings

4.1 Choices and their effects

We consider a general setting where n *choices* are made, e.g., by n parties who do not necessarily collaborate. Each n-tuple of choices results in a certain *effect*, which we simply model as an element from a set Ω. For example, Ω could be the set of real numbers, the set of vectors of real numbers, or anything else. We will call such a setting an n*-choice setting* or, simply a *setting*.

Definition 8. An n*-choice setting* is a function

$$R\colon\ A_1 \times \cdots \times A_n \to \Omega,$$

where $A_1, \ldots, A_n$ are arbitrary sets and Ω is a set equipped with an equivalence relation $\simeq$. The set A_i

[11] One could naturally postulate that $(\alpha|\mathsf{id})(\mathsf{id}|\alpha') = \alpha|\alpha'$, but this is not necessary.

is called the *ith choice domain*, and Ω is called the *effect space*.

The notion of a choice is very general; for example, in a setting where parties $p_1, \ldots, p_n$ interact with each other in multiple rounds, a choice $a_i \in A_i$ may be the entire reactive strategy of party p_i for computing the answers depending on the inputs received from the other parties.

The effect space Ω will often be equipped with a pseudo-metric δ which is compatible with the equivalence relation $\simeq$ in the sense that

$$\delta(\omega, \omega') = 0 \iff \omega \simeq \omega'. \tag{2}$$

This pseudo-metric on the effect space Ω induces a pseudo-metric on the choice settings (see Definition 2), which we also denote by δ.

4.2 Isomorphisms between choice settings

We now consider isomorphisms between choice settings. Roughly speaking, two settings, R and S, are isomorphic if the same effects can be obtained by an appropriate relabeling of the individual choices. The idea is that if R and S are isomorphic then they can be considered equivalent, in the sense that no choice-making party would prefer one setting over the other.

A relabeling of the choices is naturally represented using the notion of complete relations. Let $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ be the choice sets of R and S, respectively. Each choice $a_i \in A_i$ shall be related to a corresponding choice $b_i \in B_i$, and vice-versa. The resulting correspondence between the n-tuple of choices can be seen as a complete relation $\boldsymbol{\rho}$ from $A_1 \times \cdots \times A_n$ to $B_1 \times \cdots \times B_n$, which is *factorizable* in the following sense.

Definition 9. Let $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ be sets. A relation $\boldsymbol{\rho}$ from $A_1 \times \cdots \times A_n$ to $B_1 \times \cdots \times B_n$ is called *factorizable* if $\boldsymbol{\rho} = \rho_1 \times \cdots \times \rho_n$, where ρ_i are (A_i, B_i)-relations.

A complete factorizable relation $\boldsymbol{\rho}$ is abbreviated as *CFR*; it is the product of complete relations ρ_i from A_i to B_i. For CFRs $\boldsymbol{\rho} = \rho_1 \times \cdots \times \rho_n$ and $\boldsymbol{\rho}' = \rho'_1 \times \cdots \times \rho'_n$, we have $\boldsymbol{\rho} \circ \boldsymbol{\rho}' = (\rho_1 \circ \rho'_1) \times \cdots \times (\rho_n \circ \rho'_n)$, as is easily verified.

Lemma 2. *The composition $\boldsymbol{\rho} \circ \boldsymbol{\rho}'$ of CFRs $\boldsymbol{\rho}$ and $\boldsymbol{\rho}'$ is a CFR.*

Any CFR $\boldsymbol{\rho}$ induces an isomorphism $\overset{\boldsymbol{\rho}}{\simeq}$ between choice settings (see Definition 4). More precisely, let $\boldsymbol{\rho} = \rho_1 \times \cdots \times \rho_n$ be a CFR from $A_1 \times \cdots \times A_n$ to $B_1 \times \cdots \times B_n$. Two n-choice settings R and S with choice domains $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$, respectively, are *isomorphic (via $\boldsymbol{\rho}$ and relative to $\simeq$)* if any choices $(a_1, \ldots, a_n)$ and $(b_1, \ldots, b_n)$ related by $\boldsymbol{\rho}$ result in equivalent effects, i.e.,

$$R \overset{\boldsymbol{\rho}}{\simeq} S \quad :\Longleftrightarrow \quad (\forall i : a_i \rho_i b_i) \to R(a_1, \ldots, a_n) \simeq S(b_1, \ldots, b_n) \ .$$

In other words, two n-choice settings R and S are isomorphic via $\boldsymbol{\rho}$ if the same effects can be obtained by a relabeling of the individual choices according to the prescription $\boldsymbol{\rho}$. This isomorphism plays a crucial role in our framework. It captures the idea that, as long as all relevant consequences of an n-tuple of choices $a_1, \ldots, a_n$ are described as part of the effect, $R(a_1, \ldots, a_n)$, then a party (who can, for example, make a choice a_i from the ith choice domain A_i) would not prefer R over an isomorphic setting S (where she could make a choice b_i from a set B_i), or vice versa. This argument is described in more detail in Appendix B.

Consider an n-choice setting R with choice domains $A_1, \ldots, A_n$. A simple example of a setting $\bar{R}$ that is isomorphic to R, i.e., $\bar{R} \overset{\boldsymbol{\rho}}{\simeq} R$, is obtained by simply extending the choice domains A_i to choice domains $\bar{A}_i \supseteq A_i$ by adding further choices that are equivalent to choices in A_i, i.e., such that several choices of $\bar{A}_i$ can correspond to a single choice in A_i. In this case, the relation $\boldsymbol{\rho}$ is simply the one relating each choice in $\bar{A}_i$ to an equivalent one in A_i.

For later use, we define the *extension* of R, denoted $\mathrm{ext}(R)$, as the set consisting of all settings $\bar{R}$ that can be obtained from R by such trivial domain extensions.[12]

5 Specifications and their abstraction

5.1 Specifications and guaranteed choice spaces

In the following we consider *sets* of n-choice settings. In general, a set captures those properties (of its elements) that are relevant in a certain step of an abstraction, while ignoring those aspects that are not

[12] Formally, this set is given by

$$\mathrm{ext}(R) := \{\bar{R} : \exists \boldsymbol{\rho} \supseteq \mathsf{id} : \ \bar{R} \overset{\boldsymbol{\rho}}{\simeq} R\}$$

where $\mathsf{id} = \mathbf{1}_{A_1} \times \cdots \times \mathbf{1}_{A_n}$ is the CFR consisting of the identity relations on A_i.

relevant (namely, which element of the set it is). One does not care which elements of the set is given, as long as it is guaranteed that it is one of the set.

More specifically, we will introduce the notion of a *specification*, which is simply the set of all settings that have the properties we are interested in on a certain level of abstraction.

Definition 10. An *n-choice setting specification* (or simply *n-specification*) $\mathcal{R}$ is a set of n-choice settings, together with an n-tuple of sets, $(\hat{A}_1, \ldots, \hat{A}_n)$, such that for all i, $\hat{A}_i$ is a subset of the ith choice domain of every setting in $\mathcal{R}$. The set $\hat{A}_i$ is called the *ith guaranteed choice domain of* $\mathcal{R}$.

Consider, as an example, a setting consisting of n parties $p_1, \ldots, p_n$, each of whom can make an individual choice. The effects of their choices can then be specified by an n-specification $\mathcal{R}$ consisting of n-choice settings R with choice spaces $A_1^R, \ldots A_n^R$. This specification tells us that p_i's choice will be an element from the union of all ith choice domains, $\cup_R A_i^R$. Furthermore, the specification guarantees that p_i can always make a choice from the ith guaranteed set $\hat{A}_i \subseteq \cap_R A_i^R$. In other words, party i can count on having any choice from the guaranteed set available, but may have other choices available.

5.2 Abstraction of specifications

Abstraction means to simplify a context by ignoring certain details considered irrelevant. Here we consider abstracting an n-specification $\mathcal{R}$ by another n-specification $\mathcal{S}$ that is (generally) simpler to analyze. Any property that holds for the abstract specification $\mathcal{S}$ also holds for the concrete specification $\mathcal{R}$ (but not necessarily vice versa). This means that one can perform an analysis in the abstract setting and carry it over to the concrete setting.

Definition 11. Consider n-specifications $\mathcal{R}$ and $\mathcal{S}$ with guaranteed choice spaces $\hat{A}_1, \ldots, \hat{A}_n$ and $\hat{B}_1, \ldots, \hat{B}_n$, respectively. Let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$ be an n-tuple where

$$\pi_i : \hat{B}_i \to \hat{A}_i$$

is a function ($i = 1, \ldots, n$). We say that $\mathcal{S}$ *is a* $\boldsymbol{\pi}$*-abstraction of* $\mathcal{R}$, denoted[13]

$$\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}, \tag{3}$$

if for every $R \in \mathcal{R}$ there exists an $S \in \mathcal{S}$ and a CFR $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_n)$, such that

(i) $R \overset{\boldsymbol{\rho}}{\simeq} S$
(ii) $\pi_i^{-1} \subseteq \rho_i$ for all i.[14]

If $\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$, then $\mathcal{R}$ and $\mathcal{S}$ are often called the *concrete* and the *abstract* specification, respectively. Furthermore, the abstract specification $\mathcal{S}$ will often be defined as the ϵ-extension of another specification $\bar{\mathcal{S}}$, i.e., the set $\bar{\mathcal{S}}^\epsilon$ consisting of all settings that approximate $\bar{\mathcal{S}}$ to accuracy ϵ (see Definition 12 below).

The relation $\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$ captures the idea that the concrete specification $\mathcal{R}$ has all the relevant properties of the abstract specification $\mathcal{S}$. In other words, if an analysis is performed for the abstract specification $\mathcal{S}$, leading for example to the insight that a cryptographic scheme is secure, than this conclusion also holds for the concrete specification $\mathcal{R}$.

As a generic example, consider a situation where each of n parties $p_1, \ldots, p_n$ can make a choice. Assume that party p_i has performed an analysis of the abstract specification $\mathcal{S}$ from which she concludes that she should make a certain choice b_i, which is an element of the guaranteed choice domain $\hat{B}_i$. Relation (3) now tells her that if she is confronted with a concrete setting R that satisfies specification $\mathcal{R}$, then she could equivalently make the choice $a_i = \pi(b_i)$. In fact, Property (i) guarantees that there is a setting S satisfying $\mathcal{S}$ that is isomorphic to R (see Section 4.2). Furthermore, Property (ii) ensures that the choice a_i in the concrete setting can be inferred from the choice b_i in the abstract setting, and that a_i is among the guaranteed choices, i.e., $a_i \in \hat{A}_i$.

5.3 Composability of abstraction

As we shall see, the relation $\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$ defines a reduction (as defined in Section 3) between n-specifications. The component set of the reduction will be the set of specifications, and the constructor set will be the set of n-tuples $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$, where π_i are functions from choice domains to choice domains. In the following, we define this reduction and show that it is generally composable.

We first need to equip the set of specifications, which will form the component set, with a parallel

[13] The symbol $\sqsubseteq$ shows that abstraction is a generalized subset relation.

[14] Note that only the part of the relations ρ_i defined by π_i is common for all $R \in \mathcal{R}$, the rest of the relation ρ_i can be different for every R.

composition operation. To define this relation, consider two n-choice settings, $R : A_1 \times \cdots \times A_n \to \Omega$ and $S : B_1 \times \cdots \times B_n \to \Omega$. Furthermore, assume that the effect space is equipped with an embedding operation $\Omega \times \Omega \to \Omega$, denoted by $\|$. The direct product of R and S (as functions) then yields, as the combined n-choice setting $R \times S$, the function $(A_1 \times B_1) \times \cdots \times (A_n \times B_n) \to \Omega$ with

$$(R \times S)((\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n)) = \\ R(\alpha_1, \ldots, \alpha_n) \parallel S(\beta_1, \ldots, \beta_n) \ .$$

The composition operation $\parallel$ for specifications can then be defined as

$$\mathcal{R} \| \mathcal{S} := \{R \times S : \ R \in \mathcal{R}, S \in \mathcal{S}\} \ .$$

Similarly, we equip the set of n-tuples $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$, which will form the set of constructors, with a serial and a parallel composition operation. We do this by simply defining these operations as the corresponding serial and parallel composition operation on functions. That is, for $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$ and $\boldsymbol{\pi}' = (\pi'_1, \ldots, \pi'_n)$,

$$\boldsymbol{\pi} \circ \boldsymbol{\pi}' = (\pi_1 \circ \pi'_1), \ldots, (\pi_n \circ \pi'_n)$$

and, similarly,

$$\boldsymbol{\pi} | \boldsymbol{\pi}' = (\pi_1 \times \pi'_1, \ldots, \pi_n \times \pi'_n) \ .$$

Finally, the neutral element in the set of constructors is defined as the n-tuple $\mathsf{id} = (\mathbf{1}, \ldots, \mathbf{1})$, where $\mathbf{1}$ are the identity functions.

Then, according to Definition 6, $\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$ is a reduction, where the component set is the set of specifications, and the set of constructors is the set of n-tuples $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$.

Theorem 1. *$\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$ is a generally composable reduction.*

Proof. We have already shown that $\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$ is a reduction. To prove Condition (i) of Definition 7, assume that $\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$ and $\mathcal{S} \sqsubseteq^{\boldsymbol{\pi}'} \mathcal{T}$. Then, for every $R \in \mathcal{R}$ there exists $S \in \mathcal{S}$ and $\boldsymbol{\rho} = \rho_1 \times \cdots \times \rho_n$ with $\pi_i^{-1} \subseteq \rho_i$ such that $R \overset{\boldsymbol{\rho}}{\simeq} S$. For this S, there exists $T \in \mathcal{T}$ and $\boldsymbol{\rho}' = \rho'_1 \times \cdots \times \rho'_n$ with $\pi_i'^{-1} \subseteq \rho'_i$ such that $S \overset{\boldsymbol{\rho}'}{\simeq} T$. It follows that for every R there exists $T \in \mathcal{T}$ with $R \overset{\boldsymbol{\rho}\boldsymbol{\rho}'}{\simeq} T$, and where $(\pi_i \pi'_i)^{-1} \subseteq \boldsymbol{\rho}\boldsymbol{\rho}'$.

Condition (ii) of Definition 7 is trivially satisfied because R is isomorphic to itself via the identity relation, id.

To prove Condition (iii) of Definition 7, we need to show that $\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}$ implies $\mathcal{R} \| \mathcal{T} \sqsubseteq^{\boldsymbol{\pi}|\mathsf{id}} \mathcal{S} \| \mathcal{T}$. This is however a direct consequence of the fact that $R \overset{\boldsymbol{\rho}}{\simeq} S$ implies $R \times T \overset{\boldsymbol{\rho} \times \mathsf{id}}{\simeq} S \times T$, for any $T \in \mathcal{T}$. □

5.4 Approximation and abstraction

As we have seen, the notion of an abstraction captures the idea of describing an object only by certain relevant properties without specifying (unimportant) details. In other words, an abstract specification of an object is the set all objects that have certain properties we are interested in.

When specifying the relevant properties of an object, it is often irrelevant whether these properties are met precisely or whether they are only approximated. This means that, given a specification, we may also want to consider all settings that approximate the specification.

In this section, we formalize this idea by introducing the notion of an *ϵ-extension* $\mathcal{R}^\epsilon$ of an n-specification $\mathcal{R}$. It is defined as the set consisting of all n-settings that are close to an element in the extension $\mathrm{ext}(\mathcal{R}) := \cup_{R \in \mathcal{R}} \mathrm{ext}(R)$ of $\mathcal{R}$, with respect to some metric.

Definition 12. Let $\mathcal{R}$ be a specification for effect space Ω, equipped with a pseudo-metric δ compatible with the equivalence relation $\simeq$ on Ω (see Eq. (2)). The *ϵ-extension of* $\mathcal{R}$, denoted $\mathcal{R}^\epsilon$, is given by

$$\mathcal{R}^\epsilon := \left\{R : \ \exists \bar{R} \in \mathrm{ext}(\mathcal{R}) : \ \delta(R, \bar{R}) \leq \epsilon\right\} \ .$$

where δ is the pseudo-metric on the set of n-choice settings induced by the pseudo-metric δ on Ω (see Definition 2).

In an argument consisting of several abstraction steps, we typically want to combine the approximations made within the individual steps into one for the entire abstraction, in the sense that

$$\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}^\epsilon \wedge \mathcal{S} \sqsubseteq^{\boldsymbol{\pi}'} \mathcal{T}^{\epsilon'} \quad \Longrightarrow \quad \mathcal{R} \sqsubseteq^{\boldsymbol{\pi}\boldsymbol{\pi}'} \mathcal{T}^{\epsilon+\epsilon'} \ .$$

This property is essentially a consequence of the following lemma and the triangle inequality (see the remark below).

Lemma 3. *For any specifications $\mathcal{R}$ and $\mathcal{S}$ and for any $\epsilon \geq 0$,*

$$\mathcal{R} \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S} \quad \Longrightarrow \quad \mathcal{R}^\epsilon \sqsubseteq^{\boldsymbol{\pi}} \mathcal{S}^\epsilon \ .$$

Proof. Let R be any element from $\mathcal{R}^\epsilon$. By Definition 12, there exists $R' \in \bar{\mathcal{R}}$ such that $\delta(R, R') \leq \epsilon$, where

δ is the induced pseudo-metric on the effect space. Hence, there exists a CFR $\boldsymbol{\rho} = \rho_1 \times \cdots \times \rho_n$ and an element $S' \in \mathcal{S}$ such that $R' \overset{\boldsymbol{\rho}}{\simeq} S'$ and $\pi_i^{-1} \subseteq \rho_i$.

Let $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ be the choice spaces of R' and S', respectively. Since the relations ρ_i (for any $i \in \{1, \ldots, n\}$) are complete, there exist functions $f_i : B_i \to A_i$ such that $f_i^{-1} \subseteq \rho_i$ and which are identical to π_i^{-1} on their common domain. Furthermore, by choosing an appropriate setting $\bar{S}' \in \bar{\mathcal{S}}$, the functions f_i can be extended to functions $\bar{f}_i : \bar{B}_i \to A_i$ that have full range, i.e., $\bar{f}_i(\bar{B}_i) = A_i$. We can now define a choice setting $\bar{S}$ with choice spaces $\bar{B}_1, \ldots, \bar{B}_n$ by

$$\bar{S}(b_1, \ldots, b_n) := R(\bar{f}_1(b_1), \ldots, \bar{f}_n(b_n))$$

It is easy to verify that $\bar{S}$ is isomorphic to R via a CFR $\boldsymbol{\rho'} = \rho'_1 \times \cdots \times \rho'_n$ satisfying $\pi_i^{-1} \subseteq \rho'_i$, i.e.,

$$R \overset{\boldsymbol{\rho}'}{\simeq} \bar{S} \ .$$

Furthermore, because δ is an induced metric, the distance cannot increase when prepending a function, i.e.,

$$\delta(\bar{S}, \bar{S}') \leq \delta(R, R') \leq \epsilon \ .$$

This implies that $\bar{S} \in \mathcal{S}^\epsilon$ and thus concludes the proof. □

Remark 1. Lemma 3 together with the triangle inequality implies

$$\mathcal{S} \sqsubseteq^{\boldsymbol{\pi}'} \mathcal{T}^{\epsilon'} \quad \Longrightarrow \quad \mathcal{S}^{\epsilon} \sqsubseteq^{\boldsymbol{\pi}'} \mathcal{T}^{\epsilon+\epsilon'}$$

for any $\epsilon, \epsilon' \geq 0$.

6 Theory of systems

6.1 Systems: resources, converters, and distinguishers

At the highest level of abstraction, a *system* is an abstract object with *interfaces* by which it interacts with its environment and with other systems. Interfaces are labeled with elements of a label set. Two systems can be composed into a single system by connecting one interface from each system.[15]

A key property of a reasonable abstract theory of systems is *composition-order independence*, which is a type of generalized associativity.

Definition 13. A set of systems with composition operations is *composition-order independent* if for any system composed of several systems, the order in which the systems are composed does not matter.[16]

In this paper we consider only three special types of systems: resource systems, converter systems, and distinguishers.[17] A *resource system* (or simply *resource*), usually denoted by capital letters (e.g. R or S), is a system with interface label set $\mathcal{I}$ (e.g. $\mathcal{I} = \{1, \ldots, n\}$ or $\mathcal{I} = \{A, B, E\}$). Typically (but not only) one can think of each interface being accessible to one party. A *converter system* (or simply converter)[18], usually denoted by a Greek letter (e.g. α or β), is a system with two interfaces, where one interface is designated as the outside interface and the other as the inside interface. The inside interface of a converter α can be connected to interface $i \in \mathcal{I}$ of a resource system R; the outside interface of α serves as the new interface i of the combined system, which is again a resource system and is denoted $\alpha^i R$.[19]

We can consider a resource construction, say Z, which takes as an argument a resource, say R. (Formally, the system Z is a special type of system with $2n$ interfaces.) The combined resource is denoted as $Z(R)$ or simply ZR. In particular, let $[\cdot \| S]$ be the resource construction which simulates the resource S and makes it available in parallel (on the right side) to the argument of the construction. In other words, $[\cdot \| S]R = R \| S$ for any R. Moreover, $[S \| \cdot]$ is defined analogously.

We will often consider vectors $\boldsymbol{\alpha} = (\alpha_i)_{i \in \mathcal{I}}$ of converters, one for each interface of a resource. Such a list is denoted by a boldface Greek letter. For convenience, we will often assume that $\mathcal{I} = \{1, , \ldots, n\}$, for

[15] At this level of abstraction there is no need to model different types of interfaces. It need not be defined how systems interact (analog or digital communication, one message or a ping-pong protocol, synchronous or asynchronous communication, etc.). All that is required is that the composition of systems is defined, at least for those types of composition we are interested in. Of course, at lower levels of abstraction, the definition of the interface will matter.

[16] This is, for example, what one has in mind when drawing a configurations of several composed systems as a diagram, with systems being boxes and interfaces being lines leaving the boxes. The very fact that the order in which one draws such a diagram is irrelevant implies that one postulates composition-order independence.

[17] They allow to model all systems occurring in a cryptographic context, e.g. protocols, simulators, adversaries, etc.

[18] One can think of such a system as converting or transforming an interface into an interface with a different behavior.

[19] A system composed of a resource and converters has a star-shaped topology, with a resource in the center and a (possibly empty) chain of converter systems attached to each interface. The resulting system is again a resource with the same interface set.

example $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$, but we continue to state results for a general interface set $\mathcal{I}$ when appropriate. When α_i is connected to interface i of R we usually write $\alpha_i R$ instead of $\alpha_i^i R$. Applying $\boldsymbol{\alpha}$ to R is defined naturally: $\boldsymbol{\alpha}R = \alpha_1\alpha_2\cdots\alpha_n R$. For a vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ of converter systems and for a subset $\mathcal{P} \subset \mathcal{I}$ of the interfaces, we denote by $\boldsymbol{\alpha}_\mathcal{P}$ the vector $\boldsymbol{\alpha}$ with components only $\mathcal{P}$, and $\boldsymbol{\alpha}_\mathcal{P} R$ is understood as the resource resulting when for every $i \in \mathcal{P}$, α_i is attached to interface i of R.

A distinguisher D (for n-interface resources) is a system with $n+1$ interfaces, where n interfaces connect to the interfaces of a resource R and the other (outside) interface outputs a bit. When a distinguisher D is connected to resource R we write DR for the resulting system. Note that DR is simply a binary random variable.

The typical pseudo-metrics in cryptography are distinguisher-based metrics, i.e., the distance between two resource systems is the best advantage a distinguisher in a certain class $\mathcal{D}$ of distinguishers can achieve:

$$R \approx_\epsilon S \;:\Longleftrightarrow\; \Delta^\mathcal{D}(R,S) \leq \epsilon,$$

where $\mathcal{D}$ is, for example, either the set of all distinguishers (information-theoretic security) or the set of feasible distinguishers (computational security). Here

$$\Delta^\mathcal{D}(R,S) = \sup_{D\in\mathcal{D}} \Delta^D(R,S),$$

where $\Delta^D(R,S)$ is the advantage of D in distinguishing R nd S, i.e., the statistical distance of the binary random variables DR and DS.

We write

$$R \approx S$$

instead of $R \approx_0 S$ to denote that R and S are equivalent (according to the equivalence relation implied by $\Delta^\mathcal{D}$). We point out that for describing most cryptographic settings it suffices to consider an equivalence relation rather than a (more general) pseudo-metric. In an asymptotic setting, the metric is 0/1-valued (0 corresponding to negligible, and 1 corresponding to not negligible).

6.2 Cryptographic algebras

We are now formalizing the notions introduced above.

Definition 14. A *cryptographic algebra* $\langle \Phi, \Sigma, \approx \rangle$ for an interface set $\mathcal{I}$ consists of a set Φ of $\mathcal{I}$-resources with a parallel composition operation $\|$ [20], a set Σ of converters, and an equivalence relation on the set Φ, together with a mapping[21] $\Sigma \times \Phi \times \mathcal{I} \mapsto \Phi$ defining the resource obtained when converter α is attached to interface i of resource R, denoted as $\alpha^i R$, such that

(i) Converter application at different interfaces commutes: $\alpha^i\beta^j R \approx \beta^j\alpha^i R$ for all $i \neq j$, $R \in \Phi$, and $\alpha \in \Sigma$.
(ii) Attaching no converter is defined as a special *neutral* converter $\mathbf{1} \in \Sigma$: $\mathbf{1}^i R \approx R$ for all $i \in \mathcal{I}$ and $R \in \Phi$.[22]
(iii) If $R \approx S$ then $\alpha^i R \approx \alpha^i S$.
(iv) If $R \approx S$ then $(R\|T) \approx (S\|T)$ and $(T\|R) \approx (T\|S)$.

The commutativity condition of the above definition is the specialization of the previously mentioned composition-order independence to the specific setting with resources and converters. It states that if two converters are connected to distinct interfaces, then the order in which these operations are performed is irrelevant.

One can naturally define serial and parallel composition operations on the converter set Σ as follows:

- serial composition: $\alpha\beta$ (or $\alpha \circ \beta$) is defined by
$$(\alpha\beta)^i R := \alpha^i\beta^i R$$
for all i and R. This composition operation is associative because function composition is: $(\alpha\beta)\gamma = \alpha(\beta\gamma)$. Note that $\mathbf{1}\alpha = \alpha\mathbf{1} = \alpha$.
- parallel composition: $\alpha\|\beta$ is defined by
$$(\alpha\|\beta)^i(R\|S) := \alpha^i R\|\beta^i S$$
for all i and $R, S \in \Phi$.[23]

Definition 15. A *distinguisher* D is a mapping from Φ to the set of binary distributions

$$D : R \mapsto DR\ .$$

A distinguisher D emulating a converter $\alpha \in \Sigma^i$ at interface i induces a new distinguisher, denoted $D\alpha^i$,

[20] For every $i \in \mathcal{I}$, the i-interface of $R\|S$ consists of the two i-interfaces of R and S merged into a single interface, by some addressing mechanism that is not (yet) of interest at this level of abstraction.

[21] This may be a partial mapping, i.e., defined on a subset only.

[22] We do not postulate that there is a converter *system* that behaves like $\mathbf{1}$, for example by forwarding all messages, but it may be reasonable to assume that $\mathbf{1}$ is an actual system.

[23] Note that $(\alpha\|\beta)^i T$ need not be explicitly defined if T is not of a the form $T = R\|S$. Note also that $\alpha\|\mathbf{1} \neq \alpha$.

defined by

$$(D\alpha^i) : R \mapsto D(\alpha^i R) \ .$$

Similarly, a distinguisher emulating a resource $S \in \Phi$ in parallel induces a new distinguisher, denoted $\mathcal{D}[\cdot\|S] \subseteq \mathcal{D}$, and defined by

$$\mathcal{D}[\cdot\|S] : R \mapsto D(R\|S) \ .$$

Definition 16. A distinguisher class $\mathcal{D}$ is said to be *compatible* with a cryptographic algebra $\langle \Phi, \Sigma, \approx \rangle$ if the following holds

- $\mathcal{D}\Sigma^i \subseteq \mathcal{D}$ for $i \in \mathcal{I}$, i.e. $\mathcal{D}$ is closed under emulation of a converter in Σ.[24]
- $\mathcal{D}[\cdot\|\Phi] \subseteq \mathcal{D}$, i.e. $\mathcal{D}$ is closed under emulation of a resource in Φ.
- If $R \approx S$ then $DR = DS$, for any $D \in \mathcal{D}$.

6.3 Efficiency, feasibility and types of security

Systems (resources, converters, distinguishers) can be implemented by algorithms. We need to define two complexity notions for algorithms for implementing systems: *efficient* and *feasible*. Honest parties are generally restricted to *efficient* computation and dishonest parties are (in a computational setting) restricted to *feasible* computation, where the feasible complexity class includes the efficient complexity class. In traditional cryptography, no distinction between efficient and feasible is made and they are both defined as polynomial-time (in some suitable manner). In our abstract context, it suffices to postulate certain closure properties (which a reasonable polynomial-time notion satisfies).

More specifically, we consider a *feasibility notion* which is defined by a class of feasible resource systems, $\Phi^{\mathsf{f}} \subseteq \Phi$, a class of feasible converters $\Sigma^{\mathsf{f}} \subseteq \Sigma$, and a class of feasible distinguishers $\mathcal{D}^{\mathsf{f}} \subseteq \mathcal{D}$. Similarly, an *efficiency notion* is defined by a set of efficient converters, $\Sigma^{\mathsf{e}} \subset \Sigma$. We postulate that these obey certain closure properties.

Definition 17. A feasibility notion $(\Phi^{\mathsf{f}}, \Sigma^{\mathsf{f}}, \mathcal{D}^{\mathsf{f}})$ is said to be *closed* if the cryptographic algebra restricted to the sets Φ^{f} and Σ^{f} is still a cryptographic algebra, and the distinguisher class $\mathcal{D}^{\mathsf{f}}$ is compatible with this algebra.

Furthermore, an efficiency notion Σ^{e} is said to be *compatible* with the feasibility notion, if $\Sigma^{\mathsf{e}} \subseteq \Sigma^{\mathsf{f}}$ and $\Sigma^{\mathsf{e}} \circ \Sigma^{\mathsf{e}} \subseteq \Sigma^{\mathsf{e}}$.

[24] $\mathcal{D}\Sigma^i$ is defined naturally: $\mathcal{D}\Sigma^i = \{D\alpha^i \colon D \in \mathcal{D}, \alpha \in \Sigma\}$.

By choosing the sets in the feasibility notion appropriately, we obtain different security notions:

- The notion defined by $(\Phi^{\mathsf{f}}, \Sigma^{\mathsf{f}}, \mathcal{D}^{\mathsf{f}})$ corresponds to *computational security*, if we request the protocol converters π_i to be in Σ^{e}. The pseudo-metric $\Delta^{\mathcal{D}^{\mathsf{f}}}$ is 0/1-valued, where the value 0 means that the distinguishing advantage for the two (asymptotic families of) resources is negligible.
- The notion defined by the classes of unbounded systems (superscript u), $(\Phi^{\mathsf{u}}, \Sigma^{\mathsf{u}}, \mathcal{D}^{\mathsf{u}})$, corresponds to *information-theoretic security*.
- *Efficient* information-theoretic security is defined like information-theoretic security, but where the protocol converters π_i are required to be in Σ^{e}.

The latter distinction is meaningful and necessary since information-theoretic security does not imply computational security, but efficient information-theoretic security does.

7 Filtered resources as specifications

We now make the n-choice specifications introduced in Section 5 more concrete by describing their elements, the choice settings, using the notion of systems introduced in Section 6.

7.1 Resource systems as choice settings

Cryptographic algebras are a concretization of the concepts introduced in Section 4. A resource system R is a choice setting in the sense introduced in Section 4, namely a function from choice spaces to an effect space. The choices are converters (a party can apply to a resource) and the effect is the resulting resource when the converters are applied to R. More precisely, each party's choice space is Σ (or a subset of Σ), and a resource system R corresponds to the function $\Sigma^n \to \Phi$ defined by $\boldsymbol{\alpha} \mapsto \boldsymbol{\alpha}R$, where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ and α_i is party i's choice. Note that $\Omega = \Phi$, i.e., the effect space Ω is the resource set Φ, equipped with the equivalence relation $\approx$.

7.2 Filtered specifications: guaranteed and possible choice spaces

In the following we consider special resource specifications where for each interface i the choice domain A_i is known to be between a *guaranteed* choice space

of the form

$$\Sigma\phi_i = \{\alpha\phi_i : \ \alpha \in \Sigma\}$$

for some converter ϕ_i, and the *possible* (full) choice space Σ:

$$\Sigma\phi_i \subseteq A_i \subseteq \Sigma. \tag{4}$$

Here ϕ_i can be seen as a filter restricting access to R. A guaranteed choice $\alpha_i\phi_i \in \Sigma\phi_i$ is specified by the converter α_i. Similarly, the possible choices are those potentially (but not guaranteed to be) available to a party behaving dishonestly. Such a party can be thought of as removing the filter ϕ_i and therefore having possibly more powerful access to R than an honest party.

Definition 18. Let $\phi_1, \ldots, \phi_n$ be converters and let $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_n)$. Then $R_{\boldsymbol{\phi}}$ denotes the n-specification defined as the set of all resources R, where at each interface i the choice domain A_i satisfies (4). An n-specification of this form will be called *filtered*.

Note that by considering this set of resources, it is (by definition) considered irrelevant what the actual choice spaces for the parties are, as long as the guaranteed choices are available and not more than the maximal choices are available. As mentioned, the set captures that we do not know (or care to state) the particular actual choice spaces.

A canonical way to think of such filtered n-specifications is as follows: The resource R has buttons for some or all of the parties which, when pressed, gives that party additional functionality. The role of the filters ϕ_i is to shield the button, i.e., to make it unavailable. The act of removing the filter can (but need not) be considered as a "corruption bit" by which a party "declares" to want to have access to the possible choice space.

Example 1. A resource modeling the secret key generated by a typical key-agreement protocol (e.g. standard Diffie-Hellman) has the property that at least one party can influence the key. This can be modeled by a button which, if pressed, gives the party the capability to set the key. It is important that this feature (the button) is *not* guaranteed (i.e.. filtered) since an actual protocol typically does not guarantee (but only not exclude) that a party can freely choose the key. In contrast, a resource without buttons models the stronger primitive generating a common random string.

7.3 Abstraction of filtered specifications

Recall the notion of an abstraction of a resource specification. To apply this notion to filtered specifications $R_{\boldsymbol{\phi}}$ and $S_{\boldsymbol{\psi}}$, we require an explicit mapping from the guaranteed choices of $S_{\boldsymbol{\psi}}$ to the guaranteed choices of $R_{\boldsymbol{\phi}}$, i.e., a mapping from $\Sigma\psi_i$ to $\Sigma\phi_i$, for all $i \in \mathcal{I}$. This mapping is achieved by means of a converter π_i, as follows:

$$\Sigma\psi_i \to \Sigma\phi_i : \quad \alpha\psi_i \mapsto \alpha\pi_i\phi_i.$$

Let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$. If we understand π_i as a function, as just described, then $S_{\boldsymbol{\psi}}$ is a $\boldsymbol{\pi}$-abstraction of $R_{\boldsymbol{\phi}}$, denoted

$$R_{\boldsymbol{\phi}} \ \sqsubseteq^{\boldsymbol{\pi}} \ S_{\boldsymbol{\psi}},$$

if for every $R' \in R_{\boldsymbol{\phi}}$ there exists $S' \in S_{\boldsymbol{\psi}}$ and a CFR $\boldsymbol{\rho} = \rho_1 \times \cdots \times \rho_n$ between the choice spaces $A_1 \times \cdots \times A_n$ of R' and $B_1 \times \cdots \times B_n$ of S' such that

$$R' \overset{\boldsymbol{\rho}}{\simeq} S'$$

and, for all i,

$$\{(\gamma\pi_i\phi_i, \gamma\psi_i) : \ \gamma \in \Sigma\} \ \subseteq \rho_i.$$

7.4 Proving abstraction using local simulators

We can now state a central theorem in our abstract theory of cryptography which allows to prove statements of the form $R_{\boldsymbol{\phi}} \sqsubseteq_{\epsilon}^{\boldsymbol{\pi}} S_{\boldsymbol{\psi}}$. What is crucial is that the simulation can be performed locally (rather than jointly) and, in contrast to [1], the simulation must be ongoing, not only for the final transcript (or view).

Theorem 2. *Let $\langle \Phi, \Sigma, \approx \rangle$ be a cryptographic algebra and, for any $i \in \mathcal{I}$, let $\phi_i, \psi_i, \pi_i, \sigma_i \in \Sigma$ be converters. Then*

$$\forall \mathcal{P} \subseteq \mathcal{I} : \ \boldsymbol{\pi}_{\mathcal{P}}\boldsymbol{\phi}_{\mathcal{P}}R \approx \boldsymbol{\sigma}_{\overline{\mathcal{P}}}\boldsymbol{\psi}_{\mathcal{P}}S \qquad \Longrightarrow \quad R_{\boldsymbol{\phi}} \sqsubseteq^{\boldsymbol{\pi}} S_{\boldsymbol{\psi}}.$$

Proof. For fixed $\mathcal{P} \subseteq \mathcal{I}$, the equation

$$\boldsymbol{\pi}_{\mathcal{P}}\boldsymbol{\phi}_{\mathcal{P}}R \approx \boldsymbol{\sigma}_{\overline{\mathcal{P}}}\boldsymbol{\psi}_{\mathcal{P}}S$$

means that $\boldsymbol{\alpha}R \approx \boldsymbol{\beta}S$ for all $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)$ such that

$$(\alpha_i, \beta_i) \in \begin{cases} \{(\gamma\pi_i\phi_i, \gamma\psi_i) : \gamma \in \Sigma\} & \text{if } \ i \in \mathcal{P} \\ \{(\gamma, \gamma\sigma_i) : \gamma \in \Sigma\} & \text{if } \ i \in \overline{\mathcal{P}}, \end{cases}$$

i.e., such that

$$(\boldsymbol{\alpha},\boldsymbol{\beta}) \in \bigotimes_{i\in\mathcal{P}}\{(\gamma\pi_i\phi_i,\gamma\psi_i):\gamma\in\Sigma\} \times \bigotimes_{i\notin\mathcal{P}}\{(\gamma,\gamma\sigma_i):\gamma\in\Sigma\}.$$

Hence the left side of Theorem 2 (i.e., $\forall\mathcal{P}\subseteq\mathcal{I}: \boldsymbol{\pi}_{\mathcal{P}}\boldsymbol{\phi}_{\mathcal{P}}R \approx \boldsymbol{\sigma}_{\overline{\mathcal{P}}}\boldsymbol{\psi}_{\mathcal{P}}S$) is equivalent to $\boldsymbol{\alpha}R\approx\boldsymbol{\beta}S$ for all $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ with $(\boldsymbol{\alpha},\boldsymbol{\beta})\in\boldsymbol{\rho}$, where

$$\boldsymbol{\rho}=\left(\bigcup_{\mathcal{P}\subseteq\{1,\ldots,n\}}\bigotimes_{i\in\mathcal{P}}\{(\gamma\pi_i\phi_i,\gamma\psi_i):\gamma\in\Sigma\} \times \bigotimes_{i\notin\mathcal{P}}\{(\gamma,\gamma\sigma_i):\gamma\in\Sigma\}\right).$$

According to Lemma 1, we have $\boldsymbol{\rho}=\bigotimes_{i=1}^{n}\rho_i$ with

$$\rho_i=\{(\gamma\pi_i\phi_i,\gamma\psi_i):\gamma\in\Sigma\}\cup\{(\gamma,\gamma\sigma_i):\gamma\in\Sigma\}.$$

Let now R' be any resource in $R_{\boldsymbol{\phi}}$ with choice space $A_1\times\cdots\times A_n$, where $\Sigma\phi_i\subseteq A_i\subseteq\Sigma$. We consider S' in $S_{\boldsymbol{\psi}}$ with choice space $B_1\times\cdots\times B_n$, where

$$B_i=\Sigma\psi_i\cup A_i\sigma_i$$

and claim that $R_{\boldsymbol{\phi}}\sqsubseteq^{\boldsymbol{\pi}} S_{\boldsymbol{\psi}}$. If we restrict the relation ρ_i from above to

$$\rho_i'=\{(\gamma\pi_i\phi_i,\gamma\psi_i):\gamma\in\Sigma\}\cup\{(\gamma,\gamma\sigma_i):\gamma\in A_i\},$$

then the two conditions required by Definition 11 are satisfied: We have both $\{(\gamma\pi_i\phi_i,\gamma\psi_i):\gamma\in\Sigma\}\subseteq\rho_i'$, and $\boldsymbol{\rho}'$ is a CFR on the action spaces of R' and S' such that $R'\overset{\boldsymbol{\rho}}{\approx}S'$. □

Acknowledgments

This work was developed, refined, and used in teaching over many years.[25] Many people have given feedback. In particular, it is a pleasure to thank Joël Alwen, Zuzana Beerliova, Martin Hirt, Dennis Hofheinz, Christoph Lucas, Dominik Raub, Björn Tackmann, Stefano Tessaro, and Vassilis Zikas for very helpful discussions.

References

[1] J. Alwen, A. Shelat, and I. Visconti. Collusion-free protocols in the mediated model. InD. Wagner, ed-itor, *Advances in Cryptology-CRYPTO 2008*, vol-ume 5157 of *Lecture Notes in Computer Science*, pages 497-514. Springer, 2008.

[2] M. Backes, B. Pfitzmann, and M. Waidner. A gen-eral composition theorem for secure reactive sys-tems. In M. Naor, editor, *Theory of Cryptography, TCC 2004*, volume 2951 of *Lecture Notes in Com-puter Science*, pages 336-354. Springer, 2004.

[3] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136-145, 2001.

[4] R. Canetti, Y. Dodis, R. Pass, and S. Walfish. Uni-versally composable security with global setup. In S.P.Vadhan, editor, *Theory of Cryptography, TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 61-85. Springer, 2007.

[5] R. Canetti and M. Fischlin. Universally composable commitments. In J. Kilian, editor, *Advances in Cryptology-CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 19-40. Springer, 2001.

[6] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited (preliminary version). In *STOC*, pages 209-218, 1998.

[7] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAMJ. Comput.*, 18(1): 186-208, 1989.

[8] U. Maurer. Indistinguishability of random systems. In L.R.Knudsen, editor, *Advances in Cryptology-EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 110-132. Springer, 2002.

[9] U. Maurer. Abstraction in cryptography. In S. Halevi, editor, *Advances in Cryptology-CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, page 465. Springer, 2009.

[10] U. Maurer. Constructive cryptography-a primer. In R.Sion, editor, *Financial Cryptography*, volume 6052 of *Lecture Notes in Computer Science*, page1. Springer, 2010.

[11] U. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In M. Naor, editor, *Theory of Cryptography, TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 21-39. Springer, 2004.

[12] U. Maurer and B. Tackmann. On the soundness of authenticate-then-encrypt. In *ACM Conference on Computer and Communications Security*, pages 505-515, 2010.

[25] Some concepts of this paper were presented in an invited lecture at CRYPTO 2009 [9]. A video of the talk is available at `http://www.iacr.org/conferences/ crypto2009/videos/`.

[13] B. Pfitzmann and M. Waidner. Composition and integrity preservation of secure reactive systems. In *ACM Conference on Computer and Communications Security*, pages 245-254, 2000.

[14] M. Prabhakaran and M.Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. In D. Wagner, editor, *Advances in Cryptology-CRYPTO 2008*, vol-ume 5157 of *Lecture Notes in Computer Science*, pages 262-279. Springer, 2008.

[15] D.Unruh and J.Müller-Quade. Universally composable incoercibility. In T.Rabin, editor, *Advances in Cryptology-CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 411-428. Springer, 2010.

A Soundness of step-wise refinement

Here we formalize the claim that general composability (cf. Definition 7) is sufficient for the step-wise refinement paradigm to work.

For this, we think of a construction represented by a tree as follows. Any node v of the tree labels a component, called S_v, as well as, if v is not a leaf, a constructor, called α_v. Furthermore, for any non-leaf node v we define the constructor α_v^* of the underlying tree inductively as

$$\alpha_v^* \;=\; \begin{cases} \mathsf{id} & \text{if } v \text{ is a leaf} \\ \alpha_v \circ \left(\alpha_{v_1}^* \,|\, \cdots \,|\, \alpha_{v_d}^*\right) & \text{otherwise.} \end{cases}$$

We now formalize what it means for the step-wise refinement paradigm to work.

Definition 19. A reduction $\{\xrightarrow{\alpha}\}_{\alpha\in\Gamma}$ for component set Ω and constructor set Γ is called *sound for step-wise refinement* if for every tree the following holds: If at every node v of the tree we have

$$S_{v_1} \| \cdots \| S_{v_d} \xrightarrow{\alpha_v} S_v$$

(a local property in the tree), then for the root r and the leaves $\ell_1, \ldots, \ell_k$ we have

$$S_{\ell_1} \| \cdots \| S_{\ell_k} \xrightarrow{\alpha_r^*} S_r$$

(a global property of the tree).

By induction over the tree it is straightforward to prove the following theorem, which ultimately justifies our definition of *general composability* (Definition 7).

Theorem 3. *A reduction* $\{\xrightarrow{\alpha}\}_{\alpha\in\Gamma}$ (*for component set* Ω *and constructor set* Γ) *is sound for step-wise refinement if and only if it is generally composable.*

B Explanation for the definition of isomorphism

The following is a more detailed description of the argument sketched in Section 4.2, which justifies the use of the relation $R \overset{\boldsymbol{\rho}}{\simeq} S$. For the matter of concreteness, one may think of n parties who, in a setting R, interact using a given device. Each of the parties chooses a strategy a_i from a set A_i of possible strategies. The use of the device according to these strategies will result in a particular effect, specified by $R(a_1, \ldots, a_n)$. We compare this setting to an alternative setting, S, where the n parties use a different device. Here they can choose strategies $b_i \in B_i$, and the effect is given by $S(b_1, \ldots, b_n)$.

We will now show that if the effect captures all aspects that may be relevant to any of the parties, then none of them would prefer one of two isomorphic settings, R or S, over the other. Our argument is by induction over n.

For $n = 1$, i.e., one single party making a choice, the existence of an isomorphism between R and S via a relation $\boldsymbol{\rho} = \rho_1$ immediately implies that equivalent effects can be reached in the two settings, i.e., formally,[26]

$$R \overset{\boldsymbol{\rho}}{\simeq} S \implies R(A_1) \simeq S(B_1).$$

Since, by assumption, all relevant aspects of the setting are captured by the effect, the condition on the right hand side, $R(A_1) \simeq S(B_1)$ is sufficient to infer that no setting is preferable over the other. This already concludes the argument for $n = 1$.

For the induction step, we consider two n-choice settings R and S which are isomorphic via some fixed CFR $\boldsymbol{\rho} = \rho_1 \times \cdots \times \rho_n$. Note that if the nth party fixes a strategy $a_n \in A_n$ in setting R, one obtains the $(n-1)$-choice setting $R(\cdots, a_n) : A_1 \times \cdots \times A_{n-1} \to \Omega$. Hence, we can interpret R as a 1-choice setting, denoted

$$R^{\downarrow n} : A_n \to \Omega^{A_{\bar{n}}} : \; a_n \mapsto R(\cdots, a_n)$$

(for the nth party), whose effect space $\Omega^{A_{\bar{n}}}$ is the set of $(n-1)$-choice settings $A_1 \times \cdots \times A_{n-1} \to \Omega$ (for

[26] The equivalence between sets is defined such that for any element in one there is an equivalent element in the other.

the first $n-1$ parties). Similarly, for the setting S, we can define a 1-choice setting $S^{\downarrow 1}$ with effect space $\Omega^{B_{\bar{n}}} := (B_1 \times \cdots \times B_{n-1}) \to \Omega$).

Let us now compare the $(n-1)$-choice settings $R^{\downarrow n}(a_n)$ and $S^{\downarrow n}(b_n)$ that are obtained when the nth party makes a choice a_n or b_n in the original settings R and S, respectively. By the assumption of our induction step, we already know that none of the two settings is preferable over the other if the two settings are related via a CFR $\boldsymbol{\rho'} = \rho_1 \times \cdots \times \rho_{n-1}$, i.e., if

$$R^{\downarrow n}(a_n) \overset{\boldsymbol{\rho}'}{\simeq} S^{\downarrow n}(b_n) \ .$$

Therefore, in order to show that none of R or S is preferable it is sufficient to verify that any choice a_n is mapped via ρ_n to a choice b_n such that $R^{\downarrow n}(a_n)$ and $S^{\downarrow n}(b_n)$ satisfy the above condition, and vice versa, i.e.,

$$\forall a_n, b_n : \ a_n \, \rho_n \, b_n \ \to \ R^{\downarrow n}(a_n) \overset{\boldsymbol{\rho}'}{\simeq} S^{\downarrow n}(b_n) \ .$$

This corresponds to the relation $R^{\downarrow n} \, (\rho_n / \overset{\boldsymbol{\rho}'}{\simeq}) \, S^{\downarrow n}$ (see Eq. 1), which is equivalent to $R \overset{\boldsymbol{\rho}}{\simeq} S$. We have thus shown that, whenever the latter relation holds for two n-choice settings R and S, then none of them is preferable over the other.

C The Two-party case

The following appendices describe how the abstraction ideas of the framework can be used in explain known contexts and results. However, while here we stay close to the language used in the literature (e.g. the concept of honest and dishonest parties), later descriptions of these results in abstract cryptography my look quite different, focusing on the abstraction notion.

C.1 The equations for unfiltered resources

In this section we look at the special case of two parties ($n = 2$) and where only unfiltered resources (i.e., singleton sets) R and S are considered, i.e., where the guaranteed and the possible choice space are both Σ.

For $n = 2$, one can write the algebraic expressions involving systems in a simple form, by considering the left and the right side of a resource R as the two interfaces. For example, if converter α (β) is attached to the first (second) interface of R, then we can write simply $\alpha R \beta$ instead of $\alpha^1 \beta^2 R$.[27]

For a protocol $\boldsymbol{\pi} = (\pi_1, \pi_2)$ we have

$$\exists \sigma_1, \sigma_2 : \begin{Bmatrix} \pi_1 R \pi_2 \approx & S \\ \pi_1 R \ \approx & S \sigma_2 \\ R \pi_2 \approx & \sigma_1 S \\ R \ \approx & \sigma_1 S \sigma_2 \end{Bmatrix} \iff R \sqsubseteq^{\boldsymbol{\pi}} S. \qquad (5)$$

The direction $\Longrightarrow$ follows from Theorem 2. To see the direction $\Longleftarrow$, consider the relation $\boldsymbol{\rho} = \rho_1 \times \rho_2$ guaranteed to exist according to the definition of $R \sqsubseteq^{\boldsymbol{\pi}} S$, namely such that $R \overset{\boldsymbol{\rho}}{\approx} S$. Since ρ_i is complete, $(\mathbf{1}, \alpha_i) \in \rho_i$ for some α_i. Now we can let $\sigma_i = \alpha_i$.

If the UC framework for two parties is phrased abstractly, then one arrives at the first three equations of (5). Since in the UC framework, the case where both parties are corrupted is not considered, the fourth equation is not present. For $n \geq 3$, the UC framework is in a strict sense a special case since it involves only a single simulator.

C.2 Communication channel as a neutral resource

We now consider a further special case where R is a communication channel C, which is a neutral resource. For example, $\pi_1 C \pi_2 = \pi_1 \pi_2$ since C only connects π_1 and π_2.

Definition 20. A *plain communication channel* is the 2-party resource C for which $\alpha C \beta = \alpha\beta$ for all α, $\beta \in \Sigma$.

Of particular interest in cryptography is the question which resources can be obtained from a communication channel, i.e., for which R we have $C \sqsubseteq^{\boldsymbol{\pi}} R$ for some protocol $\boldsymbol{\pi} = (\pi_1, \pi_2)$.

C.3 An impossibility result

We now prove a general impossibility result which implies, as a special case, the impossibility of realizing universally composable commitments from a communication channel proved originally in [5]. Note that in the 2-party case, a resource can also be considered as a converter (as in the expression $S\gamma S$ below).

Theorem 4. *If $S\gamma S \not\approx S$ for all $\gamma \in \Sigma$, then there*

[27] For $n \geq 3$ interfaces, putting the interface index as a superscripts is necessary to maintain linear expressions for composed systems. An alternative would be to use formulas that are not linear but make use of the two-dimensional plain.

exists no protocol $\boldsymbol{\pi} = (\pi_1, \pi_2)$ *such that* $C \sqsubseteq^{\boldsymbol{\pi}} S$.

Proof. $C \sqsubseteq^{\boldsymbol{\pi}} S$ means that $\pi_1\pi_2 \approx S$ and $\pi_1 \approx S\sigma_2$ and $\pi_2 \approx \sigma_1 S$ for some σ_1 and σ_2. Replacing π_1 in $\pi_1\pi_2 \approx S$ using $\pi_1 \approx S\sigma_2$ yields

$$S\sigma_1\pi_2 \approx S.$$

Now replacing π_2 using $\pi_2 \approx \sigma_1 S$ yields

$$S\sigma_1\sigma_2 S \approx S,$$

which contradicts $\forall\gamma : S\gamma S \not\approx S$ (one can choose $\gamma = \sigma_1\sigma_2$). □

The commitment resource Com is defined as follows. At interface 1 one can input a value v (from a certain set). Then a "committed" message is output at interface 2. After that an "open" message can be input at interface 1, which causes v to be output at interface 2.

Corollary 1. *There exists no protocol* $\boldsymbol{\pi} = (\pi_1, \pi_2)$ *such that* $C \sqsubseteq^{\boldsymbol{\pi}}$ Com.

Proof. To prove that

$$\text{Com}\ \gamma\ \text{Com} \not\approx \text{Com}$$

for all $\gamma \in \Sigma$, we consider the distinguisher for resources Com γ Com and Com which commits to a random message and opens the message and outputs the bit 1 if and only if in the commitment phase the resource outputs "committed" and if in the opening phase the correct message is output.

For the resource Com γ Com, either no message is committed to in the second copy of Com, or the message output at the opening phase is independent of the committed message. The distinguishing advantage is at least $1 - 1/k$, where k is the cardinality of the message space of Com. □

A delay channel Del is a resource which takes as input a message from a domain at the first interface and, after a fixed time t, outputs the message at the second interface. Such a resource would be of substantial interest, as it would for example allow to implement fair exchange between two parties.

Corollary 2. *There exists no protocol* $\boldsymbol{\pi} = (\pi_1, \pi_2)$ *such that* $C \sqsubseteq^{\boldsymbol{\pi}}$ Del.

Proof. The following distinguisher has advantage 1 in distinguishing Del from DelγDel for any γ. It inputs a message and checks whether the message is output after time t. □

D Modeling indifferentiability

In this section we give a simple explanation of the theory of indifferentiability [11], as a special case of our abstraction notion. Indifferentiability is used today as a key tool for proving the soundness of hash function constructions.

By $R \equiv S$ we denote the information-theoretic equivalence of resources R and S [8] (i.e., R and S have the identical behavior).

Definition 21. For a 1-interface resource R, let $\overline{R}$ denote the 2-interface resource that behaves (identically) at both interfaces as (the same instantiation of) R.

For example, if R is a uniform random function, then $\overline{R}$ is a uniform random function accessible (identically) at two interfaces. Let $\perp$ is the special converter that blocks access to an interface.

Definition 22. A resource T is *out-bound* at interface i if $\perp^i \alpha^i T \equiv \perp^i T$ for all α, i.e., if no converter α can have an effect at the other interfaces.

In the plain form of indifferentiability, one considers single resources of the type $\overline{R}$ which are out-bound (at both interfaces), where the first interface is modeled as honest and the second interface is modeled as dishonest. This allows to model a resource providing public randomness which is also accessible to the adversary. Examples of such public randomness resources are a public random string required for implementing a hash function, and a public random oracle (see below).

In the described setting, one can show that the four conditions of (5) are reduced to the single condition, given in [11]:

Definition 23. S is *reducible to* R *in the sense of indifferentiability* if

$$\pi R \approx S\sigma$$

for some converters π and σ.

This definition captures both the information-theoretic and the computational setting.

We now prove an impossibility result.

Assume that for resource R the entire randomness can efficiently be read out. Formally, there exists an efficient converter α which in a first phase interacts only at its inner interface and, in a second phase, only interacts at its outside interface, such that

$$\overline{R}\alpha \equiv \overline{R},$$

i.e., α can reproduce the behavior of R after having read out its randomness. Let $H(R)$ be the entropy of R, i.e., the entropy of the random variable passed in α between the first and the second phase.

Theorem 5. *If one can (efficiently) extract all the randomness from R and one can (efficiently) extract (considerably) more than $H(R)$ entropy from S, then $\overline{S}$ is not reducible to $\overline{R}$ in the sense of indifferentiability.*

Proof. We need to prove that there are no π and σ such that

$$\pi\overline{R} \approx \overline{S}\sigma.$$

Let α be the converter that extracts the randomness from R, as described above. Then

$$\pi\overline{R}\alpha\pi \equiv \overline{\pi R}$$

is a symmetric resource. In contrast,

$$\overline{S}\sigma\alpha\pi$$

is not symmetric and one can easily detect this by reading out sufficiently much entropy at the left interface of $\overline{S}\sigma\alpha\pi$ and testing for equality with the corresponding output at the right interface. For $\pi\overline{R}\alpha\pi$ equality will hold while for $\overline{S}\sigma\alpha\pi$ it will not (with high probability), hence this gives a distinguisher. □

A (public) random oracle is a resource with two interfaces which provides at both interfaces access to the same random function. A (public) random string is a resource with two interfaces which provides at both interfaces access to the same uniform random string of fixed (i.e. small) length.

The following corollary corresponds to the impossibility result of [6, 11], by considering the hash function as the converter π which at its inside interface reads a (public) hash function parameter and at its outside interface answers hashing queries.

Corollary 3. *A (public) random oracle is not reducible (in the sense of indifferentiability) to a (public) finite random string.*

E The Alice-Bob-Eve setting with honest Alice and Bob

A standard cryptographic setup consists of two honest parties, Alice and Bob, connected by a certain communication resource (e.g., an insecure channel) that may be partially controlled by an adversary, Eve. The $2^3 = 8$ conditions of Theorem 2 can be shown to reduce to the following two conditions, phrased here for singleton specifications (no filters). (See also [12] for a more detailed discussion of the two conditions.)

The first condition models *availability* and states what must be achieved if no adversary is present.

$$\pi_1^A\pi_2^B\perp^E R \;\approx\; \perp^E S.$$

For example, if R is an insecure channel in parallel to a secret key, S is an authenticated channel, and π_1 and π_2 are the protocol engines that append a MAC and check a MAC, respectively, then the above condition states that if Eve is not present, the message must be delivered.

The second condition models *security* and states if Eve is present, anything she could do in the real setting she could also do in the ideal setting:

$$\pi_1^A\pi_2^B R \;\approx\; \sigma^E S.$$

F Encryption and leakable channels

In this section we briefly explain how one can model what encryption achieves, in a context where Alice, Bob, and Eve could all be potentially dishonest. This leads to making the coercibility of encryption explicit.

We consider resources with interface set $\mathcal{I} = \{A, B, E\}$. The real resource specification R consists of a secret key and an authenticated channel from A to B, which is potentially accessible to E, modeled by a button for E which (potentially) releases the message or parts of it to E. The filters ϕ_A and ϕ_B are trivial (i.e., $\mathbf{1}$), and ϕ_E hides the button.

The ideal resource specification S consists of a secure channel from A to B, which is leakable in the sense that both A and B have a button they can press, in which case the message is (potentially) released to E. The filters ψ_A and ψ_B hide the corresponding buttons.[28]

This feature is necessary to achieve the strong abstraction notion, $R_{\boldsymbol{\phi}} \sqsubseteq^{\boldsymbol{\pi}} S_{\boldsymbol{\psi}}$, i.e., local simulatability. This models the fact that Alice and Bob are coercible since they potentially have the possibility to leak the message.

[28] Formally, also E has a button, hidden by ψ_E, but not discussed further here.

Note that it is crucial that the buttons are filtered since otherwise, if the leaking feature were guaranteed, then an encryption scheme could be used as a commitment scheme (from A to E) which, as we saw, is impossible to realize from plain communication. In other words, the filtered resource specification models that the message could potentially leak, but that this is not guaranteed.

Actually, S also consist of a secret key with leakage buttons. This is necessary for the simulation to work and makes explicit that in encryption, sender and receiver become potentially committed to the key they used (i.e., could possibly be coerced to leak it).

We point out that the term coercibility can be used to interpret the feature of the specification S_{ψ} that A and B have the leakage buttons. In our framework, it does not correspond to a new security notion.

In view of the above it seems desirable to realize an unleakable channel, i.e., with no leakage buttons for A and B. Unfortunately this can be shown to be impossible. The proof is similar in spirit to the impossibility proof for commitments.

Theorem 6. *An unleakable secure channel cannot be realized from an authenticated channel and a secret key.*