Simulation-Based Security with Inexhaustible Interactive Turing Machines

Ralf Küsters
Institut für Informatik
Christian-Albrechts-Universität zu Kiel
24098 Kiel, Germany
kuesters@ti.informatik.uni-kiel.de

Abstract

Recently, there has been much interest in extending models for simulation-based security in such a way that the runtime of protocols may depend on the length of their input. Finding such extensions has turned out to be a non-trivial task. In this work, we propose a simple, yet expressive general computational model for systems of Interactive Turing Machines (ITMs) where the runtime of the ITMs may be polynomial per activation and may depend on the length of the input received. One distinguishing feature of our model is that the systems of ITMs that we consider involve a generic mechanism for addressing dynamically generated copies of ITMs. We study properties of such systems and, in particular, show that systems satisfying a certain acyclicity condition run in polynomial time. Based on our general computational model, we state different notions of simulation-based security in a uniform and concise way, study their relationships, and prove a general composition theorem for composing a polynomial number of copies of protocols, where the polynomial is determined by the environment. The simplicity of our model is demonstrated by the fact that many of our results can be proved by mere equational reasoning based on a few equational principles on systems.

1. Introduction

In the simulation-based security paradigm the security of protocols is defined in such a way that security is preserved even if the protocols are used as components of an arbitrary (polynomially bounded) distributed system. This strong composability property allows the modular design and analysis of protocols. The main idea behind simulation-based security is that the security of a protocol is defined in terms of an ideal protocol (also called, ideal functionality). A real protocol securely realizes the ideal protocol if every attack on the real protocol can be translated into an "equivalent" attack on the ideal protocol, where equivalence is specified based on an environment trying to distinguish the real attack from the ideal one.

Several related models for simulation-based security have been proposed [5, 19, 4, 3, 18, 10] (see [10] for a comparison of the models). In these models, systems of Interactive Turing Machines (ITMs) are considered. However, in the various models the ITMs can have different forms: rather standard ITMs, as used in [5], probabilistic I/O automata [4], and process calculus expressions [18, 10]. Depending on the kind of entities running in a system (environment, real/ideal adversary, simulator, real/ideal protocol) and the order of quantification over these entities, different security notions for simulation-based security are obtained, including strong [11, 10] (see also [5]), black-box [19], (dummy) universal [5, 19], and reactive simulatability [19].

The mentioned models have in common that the *total runtime* of the ITMs, i.e., the runtime summed over all activations, is bounded by a polynomial in the security parameter alone and may not depend on the length of the input that the ITMs receive from other ITMs. (We call these ITMs *exhaustible* in the following.) This is a mainly technically motivated restriction which guarantees that a system of ITMs runs in polynomial time in the security parameter. However, as explained below, it significantly limits the expressivity of the models and in some cases results in unintuitive behavior.

Recently, there has therefore been much interest in developing models for simulation-based security involving ITMs whose runtime may depend on the length of the input received from other ITMs. Canetti [7] and Hofheinz et al. [14] were the first to propose and study models for simulation-based security with such ITMs. Developing such models is a non-trivial task. As already pointed out in [4, 7, 14], naïve extensions of the existing models do not work: In a system of ITMs (whose runtime may depend on the length of their input), two ITMs can send messages back and forth among each other. Hence, such a system would not terminate, let alone perform a polynomially bounded computation, which is, however, required to guarantee the security of cryptographic primitives. Canetti [7] and Hofheinz et al. [14] have pointed out that globally bounding the runtime of an otherwise possibly non-terminating system by a polynomial, i.e., stopping the system after a polynomial bound has been reached, also does not yield a reasonable computational model for simulation-based security: an environment could (artificially) distinguish a real attack from an ideal one by measuring the overall number of steps taken in the different attacks.

Contribution of this Paper. In this paper, we propose a model for simulation-based security which extends and simplifies several aspects of previous models (see also the related work). More precisely, the main contributions of this work are twofold: First, we propose a simple, yet expressive general computational model for systems of what we call inexhaustible ITMs independent of the application to simulation-based security. A distinguishing feature of this model is a generic mechanism for addressing dynamically generated ITMs. Second, we demonstrate the flexibility and simplicity of our model by formalizing several notions of simulation-based security in it, along with a study of the relationships of the security notions and general composition theorems. In previous models, the formulations of the security notions were much more cumbersome or the security notions could not be formalized at all. Also, the composition theorems were more restricted in different respects. The simplicity of our model is reflected in the fact that many proofs can be carried out by mere equational reasoning on systems based on a few equational principles. Let us explain our general computational model and the application to simulation-based security in more detail.

The general computational model. The main building blocks of our computational model are the already mentioned *inexhaustible ITMs*. The runtime of these ITMs is only polynomially bounded per activation where the polynomial is in the length of the current input, the security parameter, and the size of the current configuration, i.e., the length of the current content written on the work tapes of the machine. This enables a machine to read every input and in every activation scan its entire current configuration. It also prevents machines from being exhausted by other machines sending useless messages; we note that in Canetti's model [7], ITMs can be exhausted. Inexhaustible ITMs have two main features that distinguish them from weakly polynomial machines [4, 14]: First, they may run in one of two modes ($\mathsf{CheckAddress}$ and $\mathsf{Compute}$). These modes are used within a generic mechanism for addressing copies of ITMs. This avoids to fix specific details of an addressing mechanism (such as session IDs) in the general computational model. Second, we distinguish between enriching and consuming input tapes of an ITM and require that the output produced by a single ITM and the size of its current configuration (i.e., the length of the content written on the work tapes of the machine) is bounded by a polynomial in the security parameter plus the length of the input that has been received on *enriching* tapes so far.

The systems of ITMs that we consider may contain an unbounded number of copies of ITMs. In a run of a system, ITMs may create new copies of ITMs by invoking other machines. In other words, the number of copies of ITMs is determined dynamically. As mentioned, using the two modes in which ITMs may run, we employ a generic mechanism for addressing copies of ITMs. In the application for simulation-based security, this mechanism allows us to model multi-party protocols and to talk about different sessions of a protocol (as needed in the composition theorems). We identify semantic and syntactic conditions on systems of ITMs which guarantee that these systems run in polynomial time. The syntactic condition is an acyclicity condition on the way ITMs are connected via the mentioned enriching tapes.

We prove several equational properties of systems in our general computational model which in the application for simulation-based security allows us to carry out many of the proofs by mere equational reasoning on systems. We also show that any (sub-)system can be simulated by a single ITM (the simulation is independent of the environment in which the system may run). In particular, this is true for those systems describing an unbounded number of sessions of a protocol. This is the core of the joint state theorem [9, 7].

While, as we will see, our general computational model forms a solid and flexible basis for studying different forms of simulation-based security, we believe that our model and the properties shown are of interest independent of the application to simulation-based security.

Simulation-based security. Based on our computational model, we state and investigate different notions of simulation-based security and prove general composition theorems. One important feature of our model is that the security notions can be stated in a uniform and concise way and that many proofs can be carried out based on the mentioned equational principles for (general) systems of ITMs.

More precisely, we consider two classes of systems for describing (real/ideal) protocols, while the latter class is only briefly discussed due to space limitations: IO-enriching protocol systems and IO-network-enriching protocol systems.

In IO-enriching protocol systems tapes which are part of the I/O interface of the (real/ideal) protocol may be enriching while those that are part of the network interface are consuming. This enables parties to produce output whose length is only polynomially bounded in the security parameter plus the length of the workload received on their I/O interface, such as messages to be encrypted, signed, or securely transmitted. For this class of protocol systems we formulate the security notions strong, black-box, (dummy) universal, and reactive simulatability and identify sufficient conditions under which these notions are equivalent. We also prove a general composition theorem for composing a polynomial number of copies of protocols where the polynomial can be determined by the environment (and/or superior protocols).

In IO-network-enriching protocol systems not only the tapes of the I/O interface but also of the network interface may be enriching, yielding a more general class of protocol systems, but with somewhat restricted security notions.

Drawbacks of models with exhaustible ITMs. Models for simulation-based security based on exhaustible ITMs have several drawbacks:

First, their expressivity is limited. For example, when specifying an ideal protocol for modeling encryption (see, e.g., [5, 2]) the number and length of messages that can be encrypted using this protocol has to be bounded by some fixed polynomial in the security parameter; the same is true for other ideal protocols, such as those for modeling signatures [5, 8, 6, 1, 2] and secure message transmission [19]. Such a fixed bound is quite artificial and also restricts the security guarantees. Inexhaustible ITMs overcome these problems. Another example that illustrates the limited expressivity in models with exhaustible ITMs is the following: A party, modeled as an ITM, running a protocol is not able to block useless messages. It first has to examine the incoming message to decide whether to drop or to further process the message. This task consumes resources, and hence, by swamping an ITM with useless messages, external parties, including the adversary and the environment, can exhaust the total runtime available to a party and force it to stop. A partial solution to this problem of blocking useless messages is the length function in the model by Backes, Pfitzmann, and Waidner [4]. A more general approach is the concept of guards introduced in [10]. However, inexhaustible ITMs supersede such constructions.

Second, models with exhaustible ITMs exhibit in some cases unintuitive and unexpected behavior. For example, almost identical protocols may not be simulatable w.r.t. black-box simulatability, while they are with universal simulatability. More concretely, consider an ideal and real protocol which are identical except that on the network interface the ideal protocol sends the bit-wise complement of messages the real protocol would send. While the real protocol realizes the ideal protocol w.r.t. universal simulatability, this is, in general, not the case w.r.t. black-box simulatability. The main reason for this peculiarity is that if the runtime of ITMs is polynomially bounded in the security parameter, then in general it is not possible to plug an entity, say $\mathcal{D}$, between two other entities, say $\mathcal{Q}_1$ and $\mathcal{Q}_2$, such that $\mathcal{D}$ can be chosen independently of at least one of the entities $\mathcal{Q}_1$ or $\mathcal{Q}_2$, and such that $\mathcal{D}$ forwards messages between $\mathcal{Q}_1$ and $\mathcal{Q}_2$ (this property is called FORWARDER property in [10]): $\mathcal{D}$ can be exhausted by these entities, i.e., the runtime available to $\mathcal{D}$ might not suffice to forward all messages between $\mathcal{Q}_1$ and $\mathcal{Q}_2$. As shown in [10], this property has a great impact on the relationships between the different security notions. When using ITMs where all tapes are enriching, the FORWARDER property can be satisfied. Another unexpected behavior is that when considering systems where the number of ITMs in the system is unbounded and is determined by the environment, then such a system cannot be simulated by a single ITM. For example, one cannot simulate an unbounded number of copies of protocols within one ITM. This is, however, what is required by the joint state theorem as stated in [7, 9]. In our model, every (sub-)system can be simulated by a single ITM.

Related Work. As mentioned above, Canetti [7] and Hofheinz et al. [14] were the first to study models for simulation-based security where the runtime of ITMs may depend on the length of their input. Let us discuss the main differences to the present work.

The results proved by Hofheinz et al. are most closely related to the results presented in this paper for IO-network-enriching protocol systems. These protocol systems are similar in expressivity to the polynomially shaped weakly polynomial collections considered in [14]. However, the way the security notions are defined is quite different from the definitions presented here. Hofheinz et al. do not consider IO-enriching protocol systems. The study of these systems is the core of the present paper, and as shown here, these systems allow to formulate all simulation-based security notions considered in the literature in a concise and uniform way, while, for instance, strong and black-box simulatability have not been investigated by Hofheinz et al. Also, the computational model employed in [14] does not explicitly allow to talk about systems with an unbounded number of dynamically generated ITMs (such systems would have to be simulated within a fixed and finite number of ITMs). Hence, without further extending the model by Hofheinz et al., a composition theorem for composing a polynomial number of machines can not be stated (let alone proved) in their setting.

The model by Canetti [7] has been evolving over time and is still subject to change. Canetti's model and the one proposed here are orthogonal. On the one hand, Canetti's model has the following main features that our model does not have: ITMs may generate the *code* of machines they invoke, the depth of invocations of "subroutine ITMs" may be linear in the security parameter and the auxiliary input given to the system, and a description of a system involves a control function which oversees whether or not an ITM is allowed to communicate with another ITM. On the other hand, our model is simpler, and yet, more expressive and flexible in the following respects: The *total* runtime of the ITMs that Canetti employs is bounded by a polynomial in the security parameter and the length of the input received on I/O tapes (minus the runtime provided to "subroutine ITMs"). In particular, the runtime of these ITMs may not be polynomial per activation, and in fact, these ITMs can be exhausted by other ITMs sending useless messages. This limits the expressivity of the model in the sense that certain protocols can not be formulated, e.g., protocols that, without consuming resources, simply ignore messages of unexpected format. Also, the exhaustion of ITM leads to much more involved constructions for the security notions and proofs. By using inexhaustible ITMs, we avoid these problems in our model. To guarantee that systems as defined by Canetti run in polynomial time, the length of the output of ITMs when invoking new ITMs must be strictly decreasing compared to the length of the input received. The number of ITMs that may be invoked by a machine is also restricted in a certain way. We do not have these restrictions in our model. Another difference between the two models is that Canetti explicitly defines as part of his computational model how session and party IDs are used to address specific copies of protocols. In the present work we instead have developed a more general mechanism for this purpose and do not fix details of the addressing mechanism in the general computational model. We also note that our composition theorems are more flexible in the way we allow protocols to connect to subprotocols. We finally mention that in the version of [7] from January 2005, Canetti considers ITMs whose runtime may depend on the input received on all tapes (he called such ITMs A-PPT). In our terminology, these are ITMs where all of the tapes are enriching. Canetti formulated different security notions using these ITMs. However, the proofs for establishing the relationships between these notions were flawed.

Structure of the Paper. In Section 2, we present our general computational model, including the definition of (inexhaustible) ITMs and systems of such ITMs, with important properties presented in Section 3. Based on the general computation model, we introduce the mentioned security notions for IO-enriching protocol systems in Section 4, with their relationships studied in Section 5. The composition theorems for this class of protocol systems are presented in Section 6. IO-network-enriching protocol systems are investigated in Section 7. We conclude in Section 8. We refer the reader to [16] for a full version of this paper.

Notation and Basic Terminology. For a bit string $a \in \{0,1\}^*$ we denote by $|a|$ the length of a. Following [7], a function $f : \{1\}^* \times \{0,1\}^* \to \mathbb{R}_{\geq 0}$ is *negligible* if for every polynomial p and q there exists k_0 such that for all $k > k_0$ and all bit strings $a \in \bigcup_{k' \leq q(k)} \{0,1\}^{k'}$ we have that $f(1^k, a) \leq \frac{1}{p(k)}$.

2. The General Computational Model

In this section, we define our general computational model independent of the application to simulation-based security. We introduce single interactive Turing machines and systems of such machines, define runs of systems, and state basic properties. We also introduce further notation and terminology, used in subsequent sections.

2.1. Inexhaustible Interactive Turing Machines

We first introduce the syntax of (inexhaustible) interactive Turing machines and then the way these machines perform their computations.

Syntax. An *(inexhaustible) interactive Turing machine (ITM, for short)* M is a probabilistic Turing machine with the following tapes and a polynomial q associated with it where q will be used as a bound in computations of M: a read-only tape on which the mode the ITM M is supposed to run is written (the *mode tape*)—the possible modes are $\mathsf{CheckAddress}$ and $\mathsf{Compute}$ (see below)—, a read-only tape on which the security parameter is written (the *security parameter tape*), a write-only tape (the *address decision tape*, used in mode $\mathsf{CheckAddress}$), zero or more *input* and *output tapes*, and *work tapes*. The input and output tapes have names and, in addition, input tapes have an attribute whose possible values are $\mathsf{consuming}$ or $\mathsf{enriching}$ (see below). We require that different tapes of M have different names. We allow M to randomly choose transitions. Alternatively, one could equip M with a random tape (see, e.g., [12]).

The set of input and output tapes of M is denoted by $\mathcal{T}(M)$, the set of input tapes by $\mathcal{T}_{in}(M)$, and the set of output tapes by $\mathcal{T}_{out}(M)$.

The names of input and output tapes will determine how ITMs are connected in a system of ITMs: If an ITM sends a message on an output tape named c, then only (a copy of) an ITM with an input tape named c can receive this message. We will use input tapes with attribute $\mathsf{enriching}$ (enriching input tapes, for short) to bound both the length of the output that may be produced by an ITM and the size of its current configuration.

Tapes named start and $\mathsf{decision}$ will serve a particular purpose. We require that only input tapes can be named start and only output tapes can be named $\mathsf{decision}$. We will later use start to provide a system with external input and to trigger an ITM if no other ITM was triggered. An ITM is triggered by another ITM if the latter sends a message to the former. An ITM with an input tape named start will be called *master ITM*. On tapes named decision the final output of a system of ITMs will be written.

An ITM M runs in one of two modes, CheckAddress or Compute. The mode in which M is supposed to run is written on the mode tape of M.

Computation. We describe the computation of an ITM M in mode CheckAddress and Compute, respectively. Informally speaking, in mode CheckAddress an ITM M checks whether the incoming message is in fact addressed to it. Typically, this mode is used for the following purpose: In a system of ITMs there may be several copies of M (belonging to different parties in a multi-party protocol or to different copies of a protocol). To address the different copies one can prefix messages with identifiers (for example, session identifiers (SIDs) and/or party identifiers (PIDs)). Now, in mode CheckAddress, M checks whether the incoming message is prefixed with the expected identifier, and either accepts or rejects that message. This allows to establish an unbounded number of *virtual channels* between ITMs. In mode Compute, the ITM actually processes an incoming message and possibly writes output on one of the output tapes, i.e., sends a message to another ITM. More formally, the computation in the two modes is defined as follows:

Mode CheckAddress: When M is activated in mode CheckAddress, it is the case that CheckAddress is written on the mode tape of M, the security parameter η is written on the security parameter tape, and one message, say m, is written on one of the input tapes, say c (the other input tapes and the output tapes are empty—or otherwise will be emptied before M starts to run—and the content on the work tapes represent the current configuration of M). We require that (i) at the end of the activation M has written accept or reject on the address decision tape; accordingly, we write $M(\mathsf{CheckAddress}, \eta, c, m) = \mathsf{accept}$ and $M(\mathsf{CheckAddress}, \eta, c, m) = \mathsf{reject}$, respectively, (ii) the computation performed by M in this mode is deterministic, i.e., is independent of internal coin tosses, and (iii) the number of transitions taken in the activation is bounded by $q(n)$ where q is the polynomial associated with M and n is the security parameter plus the length of the content of the input and work tapes at the beginning of the activation.

Mode Compute: To specify the computation in mode Compute, let l denote the length of the security parameter plus the accumulated length of all inputs written on *enriching* input tapes of M in mode Compute so far (i.e., the sum of the lengths of inputs written on enriching input tapes in the current activation in mode Compute and all previous activations in mode Compute).

When M is activated in mode Compute, it is the case that Compute is written on the mode tape of M, the security parameter η is written on the security parameter tape, and one message, say m, is written on one of the input tapes, say c (the other input tapes and the output tapes are empty—or otherwise will be emptied before M starts to run—and the content on the work tapes represent the current configuration of M). We require that the computation in every activation of M satisfies the following conditions: (i) Similar to other models [4, 7, 10], at the end of the activation, M has written *at most one* message on one of its output tapes (i.e., only one message can be sent to another ITM at a time), (ii) the number of transitions taken in the activation is bounded by $q(n)$ where q is the polynomial associated with M and n is the security parameter tape plus the length of the content of the input and work tapes at the beginning of the activation, (iii) the sum of the lengths of all outputs written on output tapes so far by M (in all activations) is bounded by $q(l)$, (iv) at the end of the current activation, the length of the content of the work tapes is bounded by $q(l)$.

We emphasize that in mode CheckAddress and Compute, M can not be exhausted: Whenever M is activated in one of the two modes, M is able to "scan" its complete current configuration, including the incoming message. Requirements (iii) and (iv) in mode Compute bound the length of the output that can be produced and the size of the internal configuration. This will be used to guarantee that a system of ITMs runs in polynomial time. Note that the bounds in (iii) and (iv) may depend on the length of the input given to the machine on enriching input tapes. When modeling protocols, this will enable parties to produce output which is only polynomially bounded in the security parameter plus the length of the input received on the I/O interface (see Section 4 and 7), such as messages to be encrypted, signed, or transmitted.

Of course, inexhaustible ITMs can simulate exhaustible ITMs (which might be useful for modeling denial-of-service attacks) since inexhaustible ITMs can count the number of steps performed so far and halt if a certain bound has been reached.

2.2. Systems of ITMs

A *system* of ITMs can be built according to the following grammar where M ranges over (descriptions of) ITMs:

$$\mathcal{S} ::= M \mid (\mathcal{S} \,||\, \mathcal{S}) \mid !\mathcal{S}.$$

We require that the set of names of input tapes of different occurrences of ITMs in a system $\mathcal{S}$ are disjoint. This implies that in $\mathcal{S}$ only at most one ITM may be a master ITM, i.e., may have start as input tape. For example, if $\mathcal{S} = M_1 \,||\, M_2 \,||\, !\, M_3$, then the above restriction says that $\mathcal{T}_{in}(M_i) \cap \mathcal{T}_{in}(M_j) = \emptyset$ for every $i \neq j$.

Intuitively, $\mathcal{S}_1 \,||\, \mathcal{S}_2$ stands for the parallel composition of the systems $\mathcal{S}_1$ and $\mathcal{S}_2$, and $!\,\mathcal{S}$ stands for the parallel composition of an unbounded number of copies of (machines in) the system $\mathcal{S}$, where the actual number of copies is determined by the environment, i.e., external or internal machines invoking (machines of) $\mathcal{S}$. Following the common terminology of process calculus [13, 17], we call '!' the *bang operator*.

We say that an ITM M occurs in the scope of a bang in $\mathcal{S}$ if $\mathcal{S}$ contains a subexpression of the form $!\,\mathcal{S}'$ such that M occurs in $\mathcal{S}'$. It will be clear from the semantics of systems, i.e., the way a system of ITMs runs, that every system $\mathcal{S}$ can equivalently be written as $\mathcal{S} = M_1 \,||\, \cdots \,||\, M_k \,||\, !\, M'_1 \,||\, \cdots \,||\, !\, M'_{k'}$ where the M_i's and M'_i's are ITMs.

We will mainly be concerned with what we call well-formed systems. These systems are guaranteed to run in polynomial time (see Section 2.3) and they satisfy a certain acyclicity condition in the way ITMs are connected via enriching tapes. To define well-formed systems, we associate a graph with a system.

A system $\mathcal{S}$ induces a graph $G_\mathcal{S}$ which is defined as follows: The nodes of $G_\mathcal{S}$ are the ITMs occurring in $\mathcal{S}$. If M_1 and M_2 are two nodes in $G_\mathcal{S}$, then there is an edge from M_1 to M_2 in $G_\mathcal{S}$ if M_1 has an output tape c and M_2 has an *enriching* input tape c. For example, the graph $G_\mathcal{S}$ of $\mathcal{S} = M_1 \,||\, M_2 \,||\, !\, M_3$ has three nodes, M_1, M_2, M_3, and there is an edge from M_i to M_j if M_i has an output tape c and M_j an enriching input tape c.

Definition 1 *We call a system $\mathcal{S}$* well-formed *if $G_\mathcal{S}$ is acyclic and the master ITM (if any) occurring in $\mathcal{S}$ is not in the scope of a bang.*

2.3. Running a System

We now define how a system $\mathcal{S}$ runs given a security parameter η and a bit string a as external input. We denote such systems by $\mathcal{S}(1^\eta, a)$.

Informally speaking, in a run of $\mathcal{S}(1^\eta, a)$ at every time only one ITM is active and all other ITMs wait for new input. The active machine may write at most one message on one of its output tapes, say c. This message is then delivered to another ITM (which has an input tape named c). The previously active machine goes into a wait state and the receiver of the message is activated, resulting, after some internal computation, into a new output which is sent to another ITM, and so on. The first ITM to be activated in a run is the master ITM. It gets a as external input (on tape start). A run stops if the master ITM, after being activated, does not produce output or output was written on an output tape named $\mathsf{decision}$. If a message is sent on an output tape c but no previously activated ITM is willing to accept the message, then a new ITM (with input tape named c) might be created. If this is not possible, the master ITM will be triggered. More formally, a run of $\mathcal{S}(1^\eta, a)$ is defined as follows:

The current (global) configuration of a system in a run is described by a tuple (A, P) where A is a sequence of configurations of ITMs, the sequence of *(previously) activated machines*, and P is a system. The ITMs occurring in P are called *passive*. (In what follows, we often do not distinguish between an ITM and its current configuration.) We emphasize that the machines in A are not the ones that are currently active, i.e., currently performing some computation—only one of these machines was just active. The machines in A are rather those machines that were active at some point in the run so far. If a message is output without a machine in A willing to accept this message (this is tested by running the machines in mode $\mathsf{CheckAddress}$ starting with the first machine in A), it is tested if there is an ITM in P that would accept the message. If so, this machine will be copied from P to A. Also, it will be removed from P if it is not in the scope of a bang, and otherwise, it will stay in P. The intuition is that there is an unbounded supply of those ITMs in P that are in the scope of an ITM.

A run ρ of a system $\mathcal{S}(1^\eta, a)$ is a sequence of tuples of the form (A, P). The initial configuration is (A_0, P_0) where A_0 is the empty sequence—no machine has been activated yet—, and $P_0 = \mathcal{S}$. Roughly speaking, one gets from one configuration (A, P) to the next configuration (A', P') by one machine (either among A or a new machine obtained from P) reading a message from its input tape, thereby updating its current configuration, and possibly writing a message on one of its output tapes (which is then read in the next step by another ITM).

The first step in a run is to read the external input a, which is provided on tape start. Since initially A_0 is empty, it is checked whether P_0 contains a master ITM, i.e., an ITM with input tape start (recall that the master ITM is uniquely determined in $\mathcal{S}$). If this is not the case, the run stops. Otherwise, if there is a master ITM, say M, and M accepts a in mode $\mathsf{CheckAddress}$, i.e., $M(\mathsf{CheckAddress}, \eta, \mathsf{start}, a) = \mathsf{accept}$, then a (written on start) is processed by M in mode $\mathsf{Compute}$ and M (more precisely, the current configuration of M) is moved to A_0, and removed from P_0 if it is not in the scope of a bang. If M did not produce output, the run stops. If M produced output, say m was written on tape c, then in the next step this output is read by another ITM yielding a successor configuration.

More precisely, to define a successor configuration of a configuration, assume that the current configuration is (A, P) where A is the sequence of configurations $M_1, \ldots, M_n$ and P is some system, and that in the previous step the message m was written by some machine on an output tape c. (As explained above, after the first step of the run, we have that $A = M_1$ where M_1 is the current configuration of the master ITM and P coincides with P_0, except that possibly the master ITM might have been removed depending on whether it was in the scope of a bang in P.). We now describe how the successor configuration of (A, P) is obtained. We distinguish three cases:

1. There exists a machine M_i in A (where i is chosen to be minimal) with an input tape named c which accepts m on c, i.e., $M_i(\mathsf{CheckAddress}, \eta, c, m) = \mathsf{accept}$. Then, M_i is activated in mode Compute to process the input m on tape c. Now, A is updated with the new configuration of M_i (note that in this new configuration a new output message may be written on one of the output tapes); P remains unchanged.
2. No machine in A with an input tape named c accepts m on tape c. But there is a passive machine M (in P) with an input tape named c such that c is an *enriching* tape of M and M accepts m on tape c. We activate M in mode Compute to process m on tape c and add the new configuration of M at the end of A. (This new configuration may contain a new output message on one of the output tapes.) If M is not in the scope of a bang, then M is removed from P.
3. If neither 1. nor 2. is satisfied, the configuration does not change.

If in one step no output is produced (in 1. by M_i, in 2. by M), then in the next step the empty input ε is read from start, i.e., the master ITM is triggered.

A run immediately stops if the master ITM after being activated has not produced output or some machine wrote output on a tape named decision—this output is the overall output of the system.

We emphasize that a copy of an ITM can only be generated if a message is sent to an ITM via its enriching input tape (see 2. above). For simulation-based security, this requirement is not an essential restriction, but it is important to guarantee that well-formed systems run in polynomial time (see below).

Definition 2 *Let p be a polynomial p and ρ be a run of $\mathcal{S}(1^\eta, a)$. Then, ρ is p-*bounded *if the accumulated length of all outputs written on output tapes during the run is $\leq p(\eta + |a|)$. A system $\mathcal{S}$ is p-*bounded *if for all security parameters η and external inputs a all runs of $\mathcal{S}(1^\eta, a)$ are p-bounded. A system $\mathcal{S}$ is* (polynomially) bounded *if there exists a polynomial p such that $\mathcal{S}$ is p-bounded.*

We can prove that the length of every run of a bounded system $\mathcal{S}(1^\eta, a)$, the number of activated ITMs in such a run, the size of the configurations in a run, and the overall number of transitions taken by ITMs in a run of $\mathcal{S}(1^\eta, a)$ can be bounded by a polynomial in $\eta + |a|$. As a result one obtains:

Proposition 3 *Every bounded system can be simulated by a single ITM.*

We call a system $\mathcal{S}$ *almost p-bounded* if the probability $\mathsf{Prob}[\text{run of } \mathcal{S}(1^\eta, a) \text{ is not } p\text{-bounded}]$ as a function of η and a is negligible. For such systems, Proposition 3 also holds, except that a simulated run may deviate from a run in the original system with negligible probability.

We note that not all systems are (almost) bounded. For example, consider the system $\mathcal{S} = M_1 \,||\, M_2$ where M_1 and M_2 are connected via enriching tapes in both directions and one of the two machines is the master ITM. Then, M_1 and M_2 could send messages back and forth forever. Another example is the system $\mathcal{S} = !\, M$ where M is a master ITM. If M in mode CheckAddress only accepts a message in its first activation and in mode Compute always produces some fixed output, then after every activation of M a new copy of M will be generated and the run of the system does not terminate.

Note that the systems in the examples are not well-formed. We can prove:

Theorem 4 *Well-formed systems are bounded.*

We note that if new ITMs could be generated when invoked not only via enriching but also via consuming input tapes, then well-formed systems would not necessarily be bounded (see [16] for an example). If, however, we restricted ourselves to the simpler case where the number of copies of ITMs is bounded by a fixed polynomial in the security parameter (rather than determined by invoking machines), Theorem 4 would still hold. All other results (with appropriate reformulations) proved in this paper would also carry over to this simpler setting.

For bounded systems $\mathcal{S}$, we denote by

$$\mathsf{Prob}[\mathcal{S}(1^\eta, a) \rightsquigarrow 1]$$

the probability that a run of $\mathcal{S}(1^\eta, a)$ returns 1, i.e., 1 is written on decision. This definition can be extended to almost bounded systems. Basically, one only considers bounded runs and ignores all others.

2.4. Further Notation and Terminology

To state properties of systems and to apply our general computational model to simulation-based security, we now introduce some more notation and terminology.

The set of tapes of a system $\mathcal{S}$ is the set of all tapes of ITMs occurring in $\mathcal{S}$. We call a tape of $\mathcal{S}$ *internal* if it occurs both as an input tape of an ITM in $\mathcal{S}$ and an output tape of (another) ITM in $\mathcal{S}$. Otherwise, a tape is called *external*. We will distinguish between the I/O- and network interface of a system. We therfore partition the set of external tapes of $\mathcal{S}$ into two types: *network* and *I/O* tapes. A network (I/O) tape can be an input or output tape depending on whether it only occurs as an input or an output tape in $\mathcal{S}$.

We will write $\mathcal{P} \mid \mathcal{Q}$ instead of $\mathcal{P} \mid\mid \mathcal{Q}$ to denote the parallel composition of $\mathcal{P}$ and $\mathcal{Q}$ where the set of internal tapes of $\mathcal{P}$ and $\mathcal{Q}$ are disjoint (this can always be achieved by renaming internal tapes). The intuition is that $\mathcal{P}$ and $\mathcal{Q}$ are different systems (e.g., a protocol and its environment) which communicate via their external tapes; they should not interfere on their internal tapes.

Two systems $\mathcal{P}$ and $\mathcal{Q}$ are *compatible* if they have the same set of external tapes. They are *I/O-compatible* if they have the same set of I/O tapes and disjoint sets of network tapes. A system $\mathcal{Q}$ is *connectible* for $\mathcal{P}$ if each common external tape of $\mathcal{P}$ and $\mathcal{Q}$ has the same type in both (network or I/O) and complementary directions (input or output). A system $\mathcal{A}$ is *adversarially connectible* for $\mathcal{P}$ if $\mathcal{A}$ is connectible for $\mathcal{P}$ and the set of external tapes of $\mathcal{A}$ is disjoint from the set of I/O tapes of $\mathcal{P}$. Thus, an adversary can only connect on the network tapes of a protocol. Similarly, $\mathcal{E}$ is environmentally connectible for $\mathcal{P}$ if it can only connect on the I/O tapes of $\mathcal{P}$. Given a set $\mathbf{B}$ of systems, we denote by $\mathrm{Con}_{\mathbf{B}}(\mathcal{Q})$ ($\mathrm{Adv}_{\mathbf{B}}(\mathcal{Q})$/$\mathrm{Env}_{\mathbf{B}}(\mathcal{Q})$) the set of all systems in $\mathbf{B}$ that are (adversarially/environmentally) connectible for $\mathcal{Q}$. We denote by $\mathrm{Sim}^{\mathcal{P}}_{\mathbf{B}}(\mathcal{Q})$ the set of all systems $\mathcal{S}$ in $\mathbf{B}$ such that $\mathcal{S}$ is adversarially connectible for $\mathcal{Q}$ and $\mathcal{S} \mid \mathcal{Q}$ is compatible with $\mathcal{P}$.

Two almost bounded systems $\mathcal{P}$ and $\mathcal{Q}$ are *equivalent* or *indistinguishable* ($\mathcal{P} \equiv \mathcal{Q}$) iff the function $f(1^\eta, a) = |\mathsf{Prob}[\mathcal{P}(1^\eta, a) \rightsquigarrow 1] - \mathsf{Prob}[\mathcal{Q}(1^\eta, a) \rightsquigarrow 1]|$ is negligible (in the sense defined at the end of Section 1).

We will later consider what we call a *dummy ITM $\mathcal{D}$* which simply forwards messages between entities: The dummy ITM has for all of its input tapes a corresponding output tape and all input tapes are enriching. The concrete set of input and output tapes that $\mathcal{D}$ has depends on the entities between which $\mathcal{D}$ is put. The dummy ITM accepts all messages on input tapes in mode $\mathsf{CheckAddress}$ and in mode $\mathsf{Compute}$ it simply copies a message received on an input tape to the corresponding output tape. Note that this is possible since all input tapes are enriching. We also emphasize that, except for the set of input and output tapes, $\mathcal{D}$ does not depend on the entities between which it is put.

3. Properties of Systems

In this section, we summarize some useful properties of systems.

The following lemma says that the dummy ITM can be plugged between two systems without changing the behavior of the overall system. In particular, using this dummy ITM the FORWARDER property mentioned in the introduction can be satisfied.

Lemma 5 *Let $\mathcal{P}$ and $\mathcal{Q}$ be two systems such that $\mathcal{P}$ is connectible for $\mathcal{Q}$ and $\mathcal{P} \mid \mathcal{Q}$ is (almost) bounded. Then, we have that the system $\mathcal{P} \mid \mathcal{D} \mid \mathcal{Q}$ is (almost) bounded and $\mathcal{P} \mid \mathcal{Q} \equiv \mathcal{P} \mid \mathcal{D} \mid \mathcal{Q}'$, where $\mathcal{D}$ is the dummy ITM (which only depends on the tapes connecting $\mathcal{P}$ and $\mathcal{Q}$, not on the systems themselves) and $\mathcal{Q}'$ is obtained from $\mathcal{Q}$ by renaming some tapes according to $\mathcal{D}$.*

We now show that every well-formed system within a more complex system can be replaced by a single ITM. This is the core of the joint state theorem as stated in [9, 7]. In what follows, we say that an input tape c is enriching in a system $\mathcal{Q}$ if there is an ITM M in $\mathcal{Q}$ such that c is an enriching input tape of M.

Lemma 6 *Let $\mathcal{Q}_1$ and $\mathcal{Q}_2$ be well-formed systems such that $\mathcal{Q}_1$ is connectible for $\mathcal{Q}_2$ and $\mathcal{Q}_1 \mid \mathcal{Q}_2$ is (almost) bounded. Then, there exists an ITM M compatible with $\mathcal{Q}_2$ such that a tape c of M is enriching iff c is enriching in $\mathcal{Q}_2$ and $\mathcal{Q}_1 \mid \mathcal{Q}_2 \equiv \mathcal{Q}_1 \mid M$. Moreover, the construction of M only depends on $\mathcal{Q}_2$ and in mode $\mathsf{CheckAddress}$ M accepts every message.*

The following lemma shows how the parallel composition of systems can be combined into one system with only consuming external tapes. This is used for moving entities (such as adversarial systems) into an environmental system.

Lemma 7 *Let $\mathcal{Q}_1, \mathcal{Q}_2, \mathcal{Q}_3$ be systems such that $\mathcal{Q}_3$ does not contain a master ITM, $\mathcal{Q}_2$ is connectible for $\mathcal{Q}_3$, $\mathcal{Q}_1$ is connectible for $\mathcal{Q}_2 \mid \mathcal{Q}_3$, and $\mathcal{Q}_1 \mid \mathcal{Q}_2 \mid \mathcal{Q}_3$ is well-formed. Then, there exists a system $\mathcal{Q}$ compatible with $\mathcal{Q}_1 \mid \mathcal{Q}_2$ such that $\mathcal{Q}_1 \mid \mathcal{Q}_2 \mid \mathcal{Q}_3 \equiv \mathcal{Q} \mid \mathcal{Q}_3$ and all external tapes of $\mathcal{Q}$ are consuming, except for start, which may be enriching (if it occurs in $\mathcal{Q}$).*

If in the above lemma, every external tape of $\mathcal{Q}_1 \mid \mathcal{Q}_2$ is consuming, then we can simply set $\mathcal{Q} = \mathcal{Q}_1 \mid \mathcal{Q}_2$. Otherwise, $\mathcal{Q}$ is an ITM which simulates $\mathcal{Q}_1 \mid \mathcal{Q}_2$ (in the environment $\mathcal{Q}_3$). In what follows, we denote $\mathcal{Q}$ as constructed in the proof of Lemma 7 by $[\mathcal{Q}_1 \mid \mathcal{Q}_2]_{\mathcal{Q}_3}$. The next lemma will allow us to "open" $[\mathcal{Q}_1 \mid \mathcal{Q}_2]_{\mathcal{Q}_3}$, i.e., replace $[\mathcal{Q}_1 \mid \mathcal{Q}_2]_{\mathcal{Q}_3}$ by $\mathcal{Q}_1 \mid \mathcal{Q}_2$, in a context different from $\mathcal{Q}_3$.

Lemma 8 *Let $\mathcal{Q}_1$ and $\mathcal{Q}_2$ be two systems which do not contain a master ITM, are well-formed and compatible, and satisfy the following condition: $\mathcal{E} \mid \mathcal{Q}_1 \equiv \mathcal{E} \mid \mathcal{Q}_2$ for every $\mathcal{E} \in Con_{\mathbf{E}}(\mathcal{Q}_1)$. (Note that $\mathcal{E} \mid \mathcal{Q}_1$ and $\mathcal{E} \mid \mathcal{Q}_2$ are well-formed.) Then for every system $\mathcal{E}_1$ connectible for $\mathcal{Q}_1$ and every system $\mathcal{E}_2$ connectible for $\mathcal{E}_1 \mid \mathcal{Q}_1$ such that $\mathcal{E}_2 \mid \mathcal{E}_1 \mid \mathcal{Q}_1$ is well-formed, we have that $[\mathcal{E}_2 \mid \mathcal{E}_1]_{\mathcal{Q}_1} \mid \mathcal{Q}_2 \equiv \mathcal{E}_2 \mid \mathcal{E}_1 \mid \mathcal{Q}_2$ and $\mathcal{E}_2 \mid \mathcal{E}_1 \mid \mathcal{Q}_2$ is almost bounded.*

4. Notions of Simulation-Based Security

In this section, we define the notions of simulation-based security mentioned in the introduction.

We first need to define protocol, adversarial, and environmental systems to specify the corresponding classes of entities. Here we define what we call IO-enriching protocol systems (or simply protocol systems) and IO-network-enriching adversarial systems (or simply adversarial systems). In this and the following two sections, we will study simulation-based security w.r.t. these classes of protocol and adversarial systems. In Section 7, different classes of protocol and adversarial systems will be considered. The definition of the environmental systems will stay the same in both settings.

An *(IO-enriching) protocol system* $\mathcal{P}$ is a well-formed system such that i) no tape in $\mathcal{P}$ is named start or decision, ii) all network tapes of $\mathcal{P}$ are consuming (I/O-tapes may be enriching), and iii) for every ITM M occurring in $\mathcal{P}$ such that M is not in the scope of a bang, we require that M accepts every incoming message in mode CheckAddress. We denote the set of protocol systems by $\mathbf{P}$. The motivation behind condition iii) is that if M does not occur in the scope of a bang, then in every run of $\mathcal{P}$ (in some environment) there will be at most one copy of M. Hence, there is no reason to address different copies of M, and therefore, in mode CheckAddress, M should accept every incoming message. This condition will be used in the proof of the composition theorem (Theorem 13 and Corollary 14).

An *(IO-network-enriching) adversarial system* $\mathcal{A}$ is a well-formed system such that no tape in $\mathcal{A}$ is named start or decision. We denote the set of adversarial systems by $\mathbf{A}$ or by $\mathbf{S}$. Note that we allow all external tapes of $\mathcal{A}$ to be enriching.

An *environmental system* $\mathcal{E}$ is a well-formed system such that all external tapes are consuming, except for start which may be enriching. We denote the set of environmental systems by $\mathbf{E}$. Note that $\mathcal{E}$ may contain start and decision. In particular, $\mathcal{E}$ may contain a master ITM (while protocol and adversarial systems may not). This choice is justified by results shown in [10] and corresponds to the choice made in other models (see, e.g., [5, 7]).

The security notions can now be defined in a concise and simple way. Note that from the definition of the different entities (in particular, the restrictions regarding what tapes may be enriching), it follows easily that all systems in the following definition, except for $\mathcal{E}\,|\,\mathcal{A}\,|\,\mathcal{S}\,|\,\mathcal{F}$, are well-formed, and hence, bounded.

Definition 9 *Let $\mathcal{P}$ and $\mathcal{F}$ be I/O-compatible protocol systems, the real and ideal protocol, respectively.*

Strong Simulatability (SS). $\mathcal{P} \leq^{SS} \mathcal{F}$ *iff* $\exists\, \mathcal{S}{\in}Sim^{\mathcal{P}}_{\mathbf{S}}(\mathcal{F})\ \forall\, \mathcal{E}{\in}Con_{\mathbf{E}}(\mathcal{P})$: $\mathcal{E}\,|\,\mathcal{P} \equiv \mathcal{E}\,|\,\mathcal{S}\,|\,\mathcal{F}$.

Black-box Simulatability (BB). $\mathcal{P} \leq^{BB} \mathcal{F}$ *iff* $\exists\mathcal{S}{\in}\ Sim^{\mathcal{P}}_{\mathbf{S}}(\mathcal{F})\ \ \forall\mathcal{A}{\in}Adv_{\mathbf{A}}(\mathcal{P})\ \ \forall\mathcal{E}{\in}Env_{\mathbf{E}}(\mathcal{A}\,|\,\mathcal{P})$: $\mathcal{E}\,|\,\mathcal{A}\,|\,\mathcal{P} \equiv \mathcal{E}\,|\,\mathcal{A}\,|\,\mathcal{S}\,|\,\mathcal{F}$ *and* $\mathcal{E}\,|\,\mathcal{A}\,|\,\mathcal{S}\,|\,\mathcal{F}$ *is almost bounded.*

Universal Simulatability/Composability (UC). $\mathcal{P}{\leq}^{UC}\mathcal{F}$ *iff* $\forall\mathcal{A}{\in}Adv_{\mathbf{A}}(\mathcal{P})\ \exists\mathcal{I}{\in}Sim^{\mathcal{A}\,|\,\mathcal{P}}_{\mathbf{S}}(\mathcal{F})\ \forall\mathcal{E}{\in}Env_{\mathbf{E}}(\mathcal{A}\,|\,\mathcal{P})$: $\mathcal{E}\,|\,\mathcal{A}\,|\,\mathcal{P} \equiv \mathcal{E}\,|\,\mathcal{I}\,|\,\mathcal{F}$.

Dummy Version of UC (UCdummy). $\mathcal{P} \leq^{UCdum} \mathcal{F}$ *iff* $\exists\mathcal{I}{\in}Sim^{\mathcal{D}\,|\,\mathcal{P}}_{\mathbf{S}}(\mathcal{F})\ \ \forall\mathcal{E}{\in}Env_{\mathbf{E}}(\mathcal{D}\,|\,\mathcal{P})$: $\mathcal{E}\,|\,\mathcal{D}\,|\,\mathcal{P} \equiv \mathcal{E}\,|\,\mathcal{I}\,|\,\mathcal{F}$.

Reactive Simulatability (RS). $\mathcal{P} \leq^{RS} \mathcal{F}$ *iff* $\forall\mathcal{A}{\in}Adv_{\mathbf{A}}(\mathcal{P})\ \ \forall\mathcal{E}{\in}Env_{\mathbf{E}}(\mathcal{A}\,|\,\mathcal{P})\ \ \exists\mathcal{I}{\in}Sim^{\mathcal{A}\,|\,\mathcal{P}}_{\mathbf{S}}(\mathcal{F})$: $\mathcal{E}\,|\,\mathcal{A}\,|\,\mathcal{P} \equiv \mathcal{E}\,|\,\mathcal{I}\,|\,\mathcal{F}$.

Using the property of dummy ITMs (Lemma 5), it is easy to see that all security notions introduced above are reflexive (modulo renaming of tapes), i.e., every protocol can be simulated by itself. Also, unlike in previous models, the above security notions do not exhibit the unintuitive properties anymore mentioned in the introduction: a real protocol in fact realizes almost identical ideal protocols. (Recall the example from the introduction where an ideal protocol coincides with the ideal protocol except that the ideal protocol outputs the bit-wise complement of messages the real protocol outputs on the network interface.)

5. Relationships Between Notions of Simulation-Based Security

We study the relationships between the different security notions. In a nutshell, we have two classes of unconditionally equivalent notions: i) strong, black-box, and dummy universal simulatability, and ii) universal and reactive simulatability. All notions are equivalent if the ideal protocol is what we call generous.

To define generous protocols, we need the following notion: Given a security parameter η and a polynomial p, we say that a non-negative integer n is *polynomially at least as big as* a non-negative integer i w.r.t. η and p if $p(\eta+n) \geq i$.

Now, roughly speaking, an (ideal) protocol is generous if the length of the output it writes on network tapes is polynomially at least as big as the length of the input it receives on I/O tapes. In other words, a generous protocol gives at least as much computation power to a simulator as it receives on its I/O interface. If this property is not satisfied for a given ideal protocol, it is often possible to have the ideal protocol output dummy messages without changing the desired security properties of the protocol (see, e.g., the functionality for signatures in [6]). If the ideal protocols are formulated in a "non-interactive way", i.e., the simulator hardly interacts with the functionality (see, e.g., the new formulation of signatures and encryption in [7]), then in order to make these ideal protocols generous one could, for example, modify the ideal protocol in such a way that it initially expects to receive the overall length of messages it is supposed to handle on the I/O interface (per party) and have the ideal protocol forward this information to the simulator in an initial phase. Such an ideal protocol would still be more flexible than an ideal protocol with an a priori bound on the number and length of messages it can handle.

Formally, generous protocols are defined as follows:

Definition 10 *We call a protocol system $\mathcal{F}$* generous *if there exists a polynomial p such that for every $\mathcal{E} \in Con_{\mathbf{E}}(\mathcal{F})$, η, a, and in every run of $(\mathcal{E} \,|\, \mathcal{F})(\eta, a)$, whenever $\mathcal{F}$ sends a message on an external tape, then the lengths of the output written so far by $\mathcal{F}$ on network tapes is polynomially at least as big as the length of the input received so far by $\mathcal{F}$ on enriching I/O tapes w.r.t. η and p.*

The following theorem summarizes the relationships between the security notions.

Theorem 11 *Let $\mathcal{P}$ and $\mathcal{F}$ be I/O compatible protocol systems. We have that:*

1. *$\mathcal{P} \leq^{SS} \mathcal{F}$ iff $\mathcal{P} \leq^{BB} \mathcal{F}$ iff $\mathcal{P} \leq^{UCdum} \mathcal{F}$.*
2. *$\mathcal{P} \leq^{UC} \mathcal{F}$ iff $\mathcal{P} \leq^{RS} \mathcal{F}$.*
3. *If $\mathcal{F}$ is generous, then: $\mathcal{P} \leq^{SS} \mathcal{F}$ iff $\mathcal{P} \leq^{BB} \mathcal{F}$ iff $\mathcal{P} \leq^{UCdum} \mathcal{F}$ iff $\mathcal{P} \leq^{UC} \mathcal{F}$ iff $\mathcal{P} \leq^{RS} \mathcal{F}$.*

Most of the above equivalences can be proved by equational reasoning using the equational principles established in Section 3. The flavor of these proofs is similar to the proof of the composition theorem for a constant number of copies of protocols (see the proof of Theorem 12). We note that for the equivalence of universal and reactive simulatability we use that the environment gets auxiliary input, i.e., is non-uniform. As shown in [15], the two notions are not equivalent in the uniform case; this is also true if, in case of reactive simulatability, the auxiliary input provided to the environment is chosen before the ideal adversary.

6. Composition Theorems

We first state a composition theorem for composing a constant number of protocols and present the proof, which is based on the equational principles established in Section 5. We then extend this theorem to an unbounded number of copies of protocols.

Theorem 12 *Let $\mathcal{P}_1, \ldots, \mathcal{P}_k, \mathcal{F}_1, \ldots, \mathcal{F}_k$ be protocol systems such that $\mathcal{P}_1 \,|\, \cdots \,|\, \mathcal{P}_k$ and $\mathcal{F}_1 \,|\, \cdots \,|\, \mathcal{F}_k$ are well-formed and for every j the following conditions are satisfied:*

1. *$\mathcal{P}_j$ is environmentally connectible for $\mathcal{P}_{j+1} \,|\, \cdots \,|\, \mathcal{P}_k$.*
2. *$\mathcal{F}_j$ is environmentally connectible for $\mathcal{F}_{j+1} \,|\, \cdots \,|\, \mathcal{F}_k$.*
3. *$\mathcal{P}_j$ and $\mathcal{F}_j$ are I/O-compatible.*
4. *$\mathcal{P}_j \leq^{SS} \mathcal{F}_j$.*

Then, $\mathcal{P}_1 \,|\, \cdots \,|\, \mathcal{P}_k \leq^{SS} \mathcal{F}_1 \,|\, \cdots \,|\, \mathcal{F}_k$.

PROOF. We prove the theorem for $k = 2$. For $k > 2$ the statement follows by induction on k.

Let $\mathcal{E} \in \mathrm{Con}_{\mathbf{E}}(\mathcal{P}_1 \,|\, \mathcal{P}_2)$. Since $\mathcal{P}_1 \leq^{SS} \mathcal{F}_1$ and $\mathcal{P}_2 \leq^{SS} \mathcal{F}_2$ we have

1. $\exists\ \mathcal{S}_1 {\in} \mathrm{Sim}_{\mathbf{S}}^{\mathcal{P}_1}(\mathcal{F}_1)\ \forall\ \mathcal{E}' {\in} \mathrm{Con}_{\mathbf{E}}(\mathcal{P}_1)$: $\mathcal{E}' \,|\, \mathcal{P}_1 \equiv \mathcal{E}' \,|\, \mathcal{S}_1 \,|\, \mathcal{F}_1$, and
2. $\exists\ \mathcal{S}_2 {\in} \mathrm{Sim}_{\mathbf{S}}^{\mathcal{P}_2}(\mathcal{F}_2)\ \forall\ \mathcal{E}' {\in} \mathrm{Con}_{\mathbf{E}}(\mathcal{P}_2)$: $\mathcal{E}' \,|\, \mathcal{P}_2 \equiv \mathcal{E}' \,|\, \mathcal{S}_2 \,|\, \mathcal{F}_2$.

Define $\mathcal{S} = \mathcal{S}_1 \,|\, \mathcal{S}_2$. Because the set of network tapes of $\mathcal{S}_1$ and $\mathcal{S}_2$ are disjoint, it easily follows that $\mathcal{S}$ is well-formed; more precisely, $\mathcal{S} \in \mathrm{Sim}_{\mathbf{S}}^{\mathcal{P}_1 \,|\, \mathcal{P}_2}(\mathcal{F}_1 \,|\, \mathcal{F}_2)$. Now, we obtain

$$\begin{array}{rcll}
\mathcal{E} \,|\, \mathcal{P}_2 \,|\, \mathcal{P}_1 & \equiv & [\mathcal{E} \,|\, \mathcal{P}_2]_{\mathcal{P}_1} \,|\, \mathcal{P}_1 & \text{(Lemma 7)} \\
& \equiv & [\mathcal{E} \,|\, \mathcal{P}_2]_{\mathcal{P}_1} \,|\, \mathcal{S}_1 \,|\, \mathcal{F}_1 & \text{(1.)} \\
& \equiv & \mathcal{E} \,|\, \mathcal{P}_2 \,|\, \mathcal{S}_1 \,|\, \mathcal{F}_1 & \text{(1., Lemma 8)} \\
& \equiv & \mathcal{E} \,|\, \mathcal{S}_1 \,|\, \mathcal{F}_1 \,|\, \mathcal{P}_2 & \\
& \equiv & [\mathcal{E} \,|\, \mathcal{S}_1 \,|\, \mathcal{F}_1]_{\mathcal{P}_2} \,|\, \mathcal{P}_2 & \text{(Lemma 7)} \\
& \equiv & [\mathcal{E} \,|\, \mathcal{S}_1 \,|\, \mathcal{F}_1]_{\mathcal{P}_2} \,|\, \mathcal{S}_2 \,|\, \mathcal{F}_2 & \text{(2.)} \\
& \equiv & \mathcal{E} \,|\, \mathcal{S}_1 \,|\, \mathcal{F}_1 \,|\, \mathcal{S}_2 \,|\, \mathcal{F}_2 & \text{(2., Lemma 8)} \\
& \equiv & \mathcal{E} \,|\, \mathcal{S}_1 \,|\, \mathcal{S}_2 \,|\, \mathcal{F}_1 \,|\, \mathcal{F}_2 & \\
& \equiv & \mathcal{E} \,|\, \mathcal{S} \,|\, \mathcal{F}_1 \,|\, \mathcal{F}_2 & \text{(Definition of } \mathcal{S}\text{)}
\end{array}$$

□

Next we present a general composition theorem for composing a polynomial number of copies of protocols where the polynomial is determined by the environment. To address the different copies of a protocol, we use the mode CheckAddress of ITMs combined with session identifiers (SIDs)

More precisely, we turn a system $\mathcal{Q}$ into its session version $\underline{\mathcal{Q}}$, which allows us to address different copies of (ITMs occurring in) $\mathcal{Q}$ by a particular SID. We first define the session version of a single ITM.

The *session version* $\underline{M}$ of an ITM M is obtained from M as follows: Basically, $\underline{M}$ simulates M except that all messages received have to be prefixed by a particular SID (i.e., in mode CheckAddress the ITM $\underline{M}$ will reject all messages not prefixed by the particular SID) and all messages sent out are prefixed by this SID. The SID $\underline{M}$ will use is the one with which $\underline{M}$ is first activated (hence, in the first activation, $\underline{M}$ will accept the incoming message in mode CheckAddress and then store the SID).

Now, the *session version* $\underline{Q}$ of a system $\mathcal{Q}$ is obtained from $\mathcal{Q}$ by replacing every ITM occurring in $\mathcal{Q}$ by its session version.

The following theorem says that if a real protocol securely realizes an ideal protocol, then an unbounded number of copies of the real protocol securely realize an unbounded number of copies of ideal protocol.

Theorem 13 *Let $\mathcal{P}, \mathcal{F}$ be protocol systems such that $\mathcal{P}$ and $\mathcal{F}$ are I/O-compatible and $\mathcal{P} \leq^{SS} \mathcal{F}$. Then, $!\underline{\mathcal{P}} \leq^{SS} !\underline{\mathcal{F}}$.*

We remark that in the above composition theorem, the session versions and the SIDs are simply used as a means to address certain (ITMs belonging to) copies of protocols. A protocol itself is not and does not need to be aware of the SID used to address ITMs belonging to it, and the specific addressing mechanism used.

As an immediate consequence of Theorem 12 and 13 we obtain the following corollary.

Corollary 14 *Let $\mathcal{P}_1, \mathcal{P}_2, \mathcal{F}_1, \mathcal{F}_2$ be protocol systems such that the systems $\mathcal{P}_1 \mid !\underline{\mathcal{P}}_2$ and $\mathcal{F}_1 \mid !\underline{\mathcal{F}}_2$ are well-formed and the following conditions are satisfied for $j \in \{1,2\}$:*

1. *$\mathcal{P}_1$ is environmentally connectible for $!\underline{\mathcal{P}}_2$.*
2. *$\mathcal{F}_1$ is environmentally connectible for $!\underline{\mathcal{F}}_2$.*
3. *$\mathcal{P}_j$ and $\mathcal{F}_j$ are I/O compatible.*
4. *$\mathcal{P}_j \leq^{SS} \mathcal{F}_j$.*

Then, $\mathcal{P}_1 \mid !\underline{\mathcal{P}}_2 \leq^{SS} \mathcal{F}_1 \mid !\underline{\mathcal{F}}_2$. If $\mathcal{P}_1$ and $\mathcal{F}_1$ coincide up to the names of the network tapes (to ensure that $\mathcal{P}_1$ and $\mathcal{F}_1$ are I/O compatible), we do not need to require 4. for $j = 1$.

In this corollary, for the case that $\mathcal{P}_1$ and $\mathcal{F}_1$ coincide up to the names of the network tapes, we use that $\mathcal{P}_1 \leq^{SS} \mathcal{F}_1$: one can choose the simulator to be the dummy ITM. In this setting, the corollary says that if (an unbounded number of copies of) an ideal protocol $\mathcal{F}_2$ is used as a component in a more complex system $\mathcal{P}_1$ ($\mathcal{F}_1$), then it can be replaced by its realization $\mathcal{P}_2$. Clearly, by iteratively applying Theorem 12 and 13, one can construct much more complex systems than those described in the above corollary.

Using the equivalences between the security notions stated in Section 5, the above composition theorems immediately carry over to the other security notions considered in this paper. (Note that if the ideal protocols $\mathcal{F}$, $\mathcal{F}_1, \dots, \mathcal{F}_k$ are generous, then so are $!\underline{\mathcal{F}}$ and $\mathcal{F}_1 \mid \cdots \mid \underline{\mathcal{F}}_k$.)

7. IO-Network-Enriching Protocol Systems

In this section, we briefly discuss how our general computational model can be applied to a different class of protocol systems, called IO-network-enriching protocol systems.

In IO-network-enriching protocol systems not only I/O tapes but also network tapes may be enriching. The class of IO-network-enriching protocol systems is quite similar in terms of expressivity to the class of polynomially shaped weakly polynomial collections defined in [14].

The security notions universal and reactive simulatability for IO-network-enriching protocol systems can be defined just as in the case of IO-enriching protocol systems (see Definition 9). However, to ensure that the systems $\mathcal{E} \mid \mathcal{A} \mid \mathcal{P}$ and $\mathcal{E} \mid \mathcal{I} \mid \mathcal{P}$ are well-formed, we need to restrict the class of adversarial systems (the definition of environmental systems remains unchanged). Note that with the current definition of (IO-network-enriching) adversarial systems the systems $\mathcal{A} \mid \mathcal{P}$ and $\mathcal{I} \mid \mathcal{F}$ might not be well-formed anymore if $\mathcal{P}$ and $\mathcal{F}$ may be IO-network-enriching protocol systems: $\mathcal{A}$ ($\mathcal{I}$) may connect to $\mathcal{P}$ ($\mathcal{F}$) by enriching network tapes, and vice versa. One therefore has to restrict adversarial systems to be IO-enriching, i.e., network tapes have to be consuming.

As in the case of IO-enriching protocols, it is not hard to show that the notions universal and reactive simulatability as defined above are equivalent (where, as in the case of IO-enriching protocol systems, we use that the environment is non-uniform). Also, a composition theorem similar to Theorem 12 can be proved.

Comparing IO-Network-Enriching and IO-Enriching Protocol Systems. The obvious advantage of IO-network-enriching protocol systems compared to IO-enriching protocol systems is that the runtime of such systems may depend on the length of the input received on network tapes *and* I/O tapes (rather than only on I/O tapes). Hence, IO-network-enriching protocol systems can forward arbitrarily long messages from the adversary. However, this can be mimiced by IO-enriching protocol systems since additional resources for forwarding messages coming from network tapes can be supplied by the environment. Hence, the additional feature of IO-network-enriching protocol systems does not seem to be essential. In fact, IO-enriching protocol systems and the security notions defined for them in this paper appear to be the more favorable and useful setting for several reasons:

As demonstrated in this paper, IO-enriching protocol systems allow for natural and simple definitions of all five security notions: strong, black-box, dummy universal, universal, and reactive simulatability. It remains to be investigated whether for the former three notions—which are often prefered over universal and reactive simulatability as they typically greatly simply proofs—equally natural definitions exist also for IO-network-enriching protocol systems. We note that in the model by Hofheinz et al. [14] dummy universal simulatability can be formulated for a class of protocols similar to IO-network-enriching protocols. However, the definitions are much more complex than those presented here for IO-enriching protocol systems. (Strong and black-box simulatability have not been defined in their setting.)

The notion of universal simulatability for IO-network-enriching protocols as defined here is problematic if the ideal adversary is invoked often by the ideal protocol (e.g., to supply ciphertexts) since the ideal adversary could run out of resources. (Recall that the tapes with which the ideal adversary connects to the ideal protocol are consuming.) Hence, certain natural ideal protocols are not realizable. In the setting for IO-enriching protocols, where all tapes of the ideal adversary may be enriching, such problems do not occur.

8. Conclusion

We have proposed an expressive general computational model for systems containing an unbounded number of inexhaustible ITMs and involving a generic addressing mechanism for copies of ITMs. This model extends and simplifies certain aspects of previous models. Based on this model, we demonstrated that several security notions for different classes of protocol systems can be formulated in a simple and uniform way and that, unlike in previous models, these security notions exhibit intuitive properties. We also proved general composition theorems. Many of the proofs could be carried out by mere equational reasoning based on a few equational principles on systems of ITMs.

Almost all models for simulation-based security (including our model) are sequential in the sense that at any time in a run at most one machine is active (an exception is the model in [18]). While, in order to concentrate on cryptographic issues, this is a good abstraction of distributed systems, the computation in real distributed systems is concurrent, i.e., many machines can be active at the same time. It would be interesting to see in how far the model presented here, including the security notions considered, could be extended to a real concurrent model.

Acknowledgments. We would like to thank Ran Canetti for many interesting discussions on models for simulation-based security. We also thank Michael Backes, Anupam Datta, John Mitchell, and the anonymous reviewers for helpful discussions and comments.

References

[1] M. Backes and D. Hofheinz. How to Break and Repair a Universally Composable Signature Functionality. In *ISC 2004*, volume 3225 of *Lecture Notes in Computer Science*, pages 61–72. Springer, 2004.

[2] M. Backes, B. Pfitzmann, and M. Waidner. A composable cryptographic library with nested operations. In *CCS 2003*, pages 220–230. ACM, 2003.

[3] M. Backes, B. Pfitzmann, and M. Waidner. A General Composition Theorem for Secure Reactive Systems. In *TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 336–354. Springer, 2004.

[4] M. Backes, B. Pfitzmann, and M. Waidner. Secure Asynchronous Reactive Systems. Technical Report 082, Cryptology ePrint Archive, 2004.

[5] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *FOCS 2001*, pages 136–145. IEEE Computer Society, 2001.

[6] R. Canetti. Universally Composable Signature, Certification, and Authentication. In *CSFW 2004*, pages 219–233. IEEE Computer Society, 2004.

[7] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. Technical report, Cryptology ePrint Archive, December 2005. Online available at `http://eprint.iacr.org/2000/067.ps`.

[8] R. Canetti and H. Krawczyk. Universally Composable Notions of Key Exchange and Secure Channels. In *EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 337–351. Springer, 2002.

[9] R. Canetti and T. Rabin. Universal Composition with Joint State. In *CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 265–281. Springer, 2003.

[10] A. Datta, R. Küsters, J. Mitchell, and A. Ramanathan. On the Relationships Between Notions of Simulation-Based Security. In *TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 476–494. Springer-Verlag, 2005.

[11] A. Datta, R. Küsters, J. Mitchell, A. Ramanathan, and V. Shmatikov. Unifying Equivalence-Based Definitions of Protocol Security. In *WITS 2004*, 2004.

[12] O. Goldreich. *Foundations of Cryptography*, volume 1. Cambridge Press, 2001.

[13] C. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[14] D. Hofheinz, J. Müller-Quade, and D. Unruh. Polynomial Runtime in Simulatability Definitions. In *CSFW 2005*, pages 156–169. IEEE Computer Society, 2005.

[15] D. Hofheinz and D. Unruh. Comparing two notions of simulatability. In *TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 86–103. Springer-Verlag, 2005.

[16] R. Küsters. Simulation-Based Security with Inexhaustible Interactive Turing Machines. To appear in the Cryptology ePrint Archive, 2006. Also available from http://www.ti.informatik.uni-kiel.de/~kuesters/.

[17] R. Milner. *A Calculus of Communicating Systems*. Springer-Verlag, 1980.

[18] J. Mitchell, A. Ramanathan, A. Scedrov, and V. Teague. A probabilistic polynomial-time calculus for analysis of cryptographic protocols (preliminary report). In *17th Annual Conference on the Mathematical Foundations of Programming Semantics*, 2001.

[19] B. Pfitzmann and M. Waidner. A Model for Asynchronous Reactive Systems and its Application to Secure Message Transmission. In *IEEE Symposium on Security and Privacy*, pages 184–201. IEEE Computer Society Press, 2001.