Technical Report: Limits of the BRSIM/UC Soundness of Dolev-Yao Models with Hashes*

Michael Backes[1], Birgit Pfitzmann[2], Michael Waidner[2]

[1] Saarland University
[2] IBM Zurich Research Lab

September 21, 2008

Abstract. Automated tools such as model checkers and theorem provers for the analysis of security protocols typically abstract from cryptography by Dolev-Yao models, i.e., abstract term algebras replace the real cryptographic operations. Recently it was shown that in essence this approach is cryptographically sound for certain operations like signing and encryption. The strongest results show this in the sense of blackbox reactive simulatability (BRSIM)/UC with only small changes to both Dolev-Yao models and natural implementations. This notion essentially means the preservation of arbitrary security properties under active attacks in arbitrary protocol environments.
We show that it is impossible to extend the strong BRSIM/UC results to usual Dolev-Yao models of hash functions in the general case. These models treat hash functions as free operators of the term algebra. This result does not depend on any restriction of the real hash function; even probabilistic hashing is covered. In contrast, we show that these models are sound in the same strict sense in the random oracle model of cryptography. For the standard model of cryptography, we also discuss several conceivable restrictions and extensions to the Dolev-Yao models and classify them into possible and impossible cases in the strong BRSIM/UC sense.

1 Introduction

Tools for proving security protocols typically abstract from cryptography by deterministic operations on abstract terms and simple cancellation rules. An example term is $\mathsf{E}_{pke_w}(\mathsf{hash}(\mathsf{sign}_{sks_u}(m, N_1), N_2))$, where m denotes a payload message and N_1, N_2 two nonces, i.e., representations of fresh random numbers. We wrote the keys as indices only for readability; formally they are normal operands in the term. A typical cancellation rule is $\mathsf{D}_{ske}(\mathsf{E}_{pke}(m)) = m$ for corresponding keys. The proof tools handle these terms symbolically, i.e., they never evaluate them to bitstrings. In other words, the tools perform abstract algebraic manipulations on trees consisting of operators and base messages, using only the cancellation rules, the message-construction rules of a particular protocol, and abstract models of networks and adversaries. Such abstractions, although different in details, are collectively called Dolev-Yao models after their first authors [2].

It is not obvious that a proof in a Dolev-Yao model implies security with respect to real cryptographic definitions. Recently, this long-standing gap was essentially closed by proving that an almost normal Dolev-Yao model of several important cryptographic system types can be implemented with real cryptographic systems secure according to standard cryptographic definitions in a way that offers blackbox reactive simulatability [3]. We abbreviate blackbox reactive simulatability by BRSIM in the following. This security (in other words soundness) notion essentially means that one system, here the cryptographic realization, can be plugged into arbitrary protocols instead of another system, here the Dolev-Yao model, without any noticeable difference [4–6]. Essentially the same notion is also called UC for its universal composition properties [7].[1] In other words, this result shows that the Dolev-Yao model as such can serve as an ideal functionality that is correctly implemented by a real functionality given by actual cryptographic systems.

* An earlier version of this work appeared in [1].

[1] Recent revisions of the long version of [7] also contain an explicit blackbox version of UC, which is proven to be equivalent to UC. A similar equivalence was first shown in the long version of [4] for universal and blackbox synchronous reactive simulatability.

Extensions of this BRSIM/UC result to more cryptographic primitives were presented in [8–10] and used in protocol proofs [11–16]. General theorems on property preservation through the BRSIM notion imply that the same Dolev-Yao model and realization also fulfill some other soundness notions [4, 17–19, 17, 20, 21], and further soundness results specific to this Dolev-Yao model and realization were proved in [22]. Stronger links of this Dolev-Yao model to conventional Dolev-Yao type systems were provided in [23], and an integration into the Isabelle theorem prover in [24]. Earlier results on relating Dolev-Yao models and real cryptography considered passive attacks only [25–27]. Later papers [28–30] consider to what extent restrictions to weaker security properties, such as integrity only, and/or less general protocol classes, e.g., a specific class of key exchange protocols, allow simplifications compared with [3]. [2] Since computational soundness has become a highly active line of research, we exemplarily list further recent results in this area without going into further details [36–39].

No prior paper relating Dolev-Yao models and cryptography considers hashing or one-way functions although they are important operators in automated proof tools based on Dolev-Yao models, e.g., [40–43]. (A very recent report does, but only under passive attacks [44].) The standard model is that hash is a free operator in the term algebra, i.e., there is no inverse operator, nor any other cancellation rule with operators like E and D. Only a party who knows or guesses a potentially hashed term t can test whether hashing t equals a given hash term h. The goal of our paper is to close this gap, and to study how the soundness results in the sense of BRSIM/UC can be extended when hash or one-way functions are added to a Dolev-Yao model and its cryptographic implementation. In the following, we only speak of hash functions since the standard Dolev-Yao model for the two classes is the same.

1.1 Our Contributions

It turns out that proving BRSIM/UC soundness for Dolev-Yao models with hash functions is impossible in a general way. Note that the question is not whether a hash function is a good and generally usable cryptographic primitive by itself, but only whether its idealization as a free operator in a term algebra, or a similar plausible idealization, is sound in this strong and pluggable sense. Prior work showed that certain (classes of) ideal functionalities are not BRSIM/UC-securely realizable, e.g., for bit commitments [45], coin tossing, zero knowledge, and oblivious transfer [7], classes of secure multi-party computation [46] and certain game-based definitions [47]. However, none of these works investigated a Dolev-Yao model. Impossibility of BRSIM/UC soundness of a Dolev-Yao model with XOR was shown in [48]. For our case of hashes, the reasons for impossibility and thus the proofs are quite different. Furthermore, the proofs in [48] are reduction proofs, essentially saying that if an idealization of XOR and other cryptographic operations is soundly implementable in the sense of BRSIM/UC, it can be used to compute cryptographic algorithms and is therefore not intuitively Dolev-Yao. In contrast, we obtain absolute impossibility results. We achieve this by making stronger definitions on what makes an ideal functionality of hashing and other cryptographic operations a Dolev-Yao model.

It is important to note that there is so far no rigorous definition of "any Dolev-Yao model" in the literature that is independent of specific underlying system models such as CSP, π-calculus, IO automata, or strand spaces. For positive results, this is not a problem. However, an impossibility result that only holds for one such model would not be very convincing. (In particular, the closest model to build on would be that from [3], and due to syntax idiosynchrasies many people find it hard to transfer basic ideas from that model to others.) Hence, instead of proving impossibility for one specific Dolev-Yao model, we will only make certain assumptions on the Dolev-Yao model; we believe they are fulfilled by all such models existing so far. Essentially we only assume that the hash functions are abstracted as free operators as informally explained above, that they are applicable to arbitrary terms, and that the model contains some other typical operators and base types.

One reason (but not the only one) for the impossibility in the general case is that hash functions, at least those with a one-way property, are by nature committing, i.e., if one first gets $h = \mathsf{hash}(m)$ and later m one can validate whether indeed $h = \mathsf{hash}(m)$. It is well known that such a commitment property

[2] There is also work on formulating syntactic calculi for dealing with probabilism and polynomial-time considerations and encoding them into proof tools, in particular [31–35]. This is orthogonal to the work of justifying Dolev-Yao models, which offer a higher level of abstraction and thus much simpler proofs where applicable, so that proofs of larger systems can be automated.

often causes problems in proofs of BRSIM/UC: If the simulator has to simulate a bitstring for h before knowing m, then whatever it picks will most likely not match m. Thus the simulation fails if m is later revealed. In some cases, the commitment problem can be circumvented by using non-standard models of cryptography, e.g., the random oracle model [49] or the common random string model, cf. [45]. Indeed we can show BRSIM/UC for the standard Dolev-Yao model of hashes if the cryptographic realization of the hash function is treated as a random oracle.

We can also consider probabilistic hashing [50, 51]. First, it is not an alternative for justifying the Dolev-Yao abstraction of hashing by a free operator in the sense of BRSIM/UC: Instead one needs hash and verify operators that cancel, similar to authentication. (Even for the purely passive case this extension is made in [44].) Moreover, hashing the same message must produce a fresh term each time, similar to standard models of nonces, or the Dolev-Yao style model of probabilistic encryption in [3]. This freshness is needed because it is distinguishable in the real system whether a message was hashed twice, or whether an existing hash value was forwarded. Hence the implementation would be incorrect in the sense of BRSIM/UC if this were not possible in the ideal, Dolev-Yao style system. Even then, however, our impossibility results hold; we sketch this extension below. Roughly, this is so because probabilistic hashing mainly improves the secrecy of the hashed message; however, the imperfect secrecy offered by deterministic hash functions is not an argument in our counterexamples.

For the standard model of cryptography, the next question is whether certain restrictions on the use of hash functions enable a BRSIM/UC soundness result. One option is to restrict the types of terms that can be hashed, in particular to forbid the hashing of payloads. By "payload" we mean an application message, e.g., an email text or the amount and currency of a payment in a payment protocol that uses the cryptographic functionality. Technically, payloads play a special role (as m in the example of the commitment problem shows) because they are known outside the cryptographic system and thus can typically not be modified by the simulator, in contrast to nonces, keys, etc. As to practical usage, this restriction is serious but not unreasonable; e.g., key exchange protocols typically do not use general payloads, and indeed [30] relies on their absence. However, we still obtain an impossibility result if excluding payloads is the only restriction on hashable terms. The basic idea in that case is that by constructing large enough terms, the users of the cryptographic system can simulate payloads. Another conceivable restriction is therefore on the size of hashable terms. Again this restriction is serious (e.g., general hash chains and trees are now excluded) but not unreasonable because many protocols only use rather small terms. In fact, for the restriction to hashing single nonces, we obtain a positive result, but so far not for any other type of restriction to small terms. This result may seem surprising: A typical counterargument is "how can this be true if we allow real hash functions that leak parts of the message?" The point is that nonces are system-internal, and thus leaking bits about them is not harmful by itself as long as the application-level properties of the Dolev-Yao model remain fulfilled, i.e., essentially that the adversary cannot guess nonces of which it has only seen hash values. Clearly this argument does not hold for payloads, and it also does not hold for keys, because leaking key bits would harm later key usage. An example of protocols that use hashing only for single nonces are protocols that use hashing only for one-time signatures.

Another restriction is to give up the ideal secrecy property of the hash functions, i.e., to give at least the ideal adversary an operator that inverts hash. The technical motivation is that this clearly prevents the commitment problem. On the application side, this restriction excludes all protocols where one-wayness is the core property for which a hash function is used. However, there are protocols where shortening of messages with collision resistance is the core property desired; here such a model could be used. Note that in a Dolev-Yao model we can have collision freeness, i.e., no equality between hashes of different terms, without secrecy. The realization of such an operator by a real cryptographic function would of course still have to be collision-resistant and thus one-way if it is sufficiently shortening. Anyway, we still obtain an impossibility result in this case: No shortening hash function exists that enables a secure realization of a Dolev-Yao model with hashes even without secrecy. If we combine giving up secrecy with not hashing payloads and only terms of constant size, then we obtain a positive result.

Of course the restrictions that we considered are not the only conceivable ones; in particular it may be interesting to find other positive cases in the standard model of cryptography than the two that we prove. Furthermore, we do not exclude that even the standard Dolev-Yao model of hashes may be sound with respect to weaker soundness definitions.

2 Summary of Reactive Simulatability/UC, also with Random Oracle

As our results are for the security definitions of BRSIM/UC, we first briefly review this notion. BRSIM/UC is used for comparing an ideal and a real system with respect to security [4, 5, 7]. We believe that our following results are independent of the small differences between the definition styles and therefore write "BRSIM/UC" and similar term pairs like "ideal system/functionality". For the actual results, we have to use a specific formalism, and we use that from [5]. Here one speaks of ideal and real systems (the functionalities and protocols of UC). The ideal system is often called TH for "trusted host", see Figure 1, and the protocol machines of the real system are often called M_u, where u is a user index. The ideal or real system interacts with arbitrary so-called honest users, often collectively denoted by a machine H; this corresponds to potential protocols or human users to whom the functionality is offered. Furthermore, the ideal or real system interacts with an adversary, often denoted by A, who is often given more power than the honest users; in particular in real systems A typically controls the network. A and H can interact; this corresponds to known- and chosen-message attacks etc.

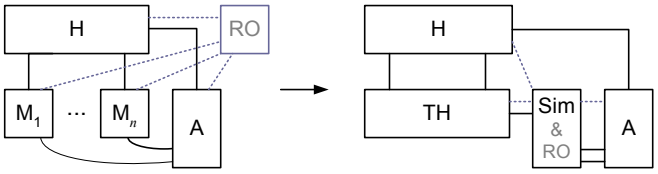

Fig. 1. Overview of blackbox reactive simulatability with a real system on the left and an ideal system on the right, and a potential random oracle. The views of H must be indistinguishable.

Reactive simulatability between the real and ideal system essentially means that for every attack on the real system there exists an equivalent attack on the ideal system. More specifically, blackbox reactive simulatability (BRSIM) states that there exists a simulator Sim that can use an arbitrary real adversary as a blackbox, such that arbitrary honest users cannot distinguish whether they interact with the real system and the real adversary, or with the ideal system and the simulator with its blackbox. Indistinguishability, here applied to the two families of views of the honest users, is the well-known notion from [52]. We always assume that all parties are polynomial-time. The original formulation of BRSIM as sketched above would allow the simulator to also modify the interaction between A and H; however, all known positive BRSIM proofs work as in Figure 1. The reason is similar to why all known positive RSIM proofs are BRSIM proofs: It is hard to make concrete use of the communication of two unknown machines A and H just as of the program text of the unknown machine A. We therefore show impossibility for this usual, stronger notion of BRSIM. Moreover, the blackbox version of UC also does not allow the simulator access to the corresponding interaction (called adversary and environment there), i.e., it corresponds to Figure 1 immediately.

A formal representation of random oracles in the UC notation was given in [53]. This can be used one-to-one in the BRSIM terminology. In a real system, each machine has distinguished connections for querying the random oracle RO, which is the usual stateful machine from [49] that generates a random string as "hash value" for each message m when it is first queried about m. In the ideal system with a simulator, these distinguished connections connect to the simulator, i.e., the simulator learns every query to the random oracle and can give arbitrary answers. This is also shown in Figure 1.[3]

[3] Alternatively, one could use a correct random oracle also in the ideal system and only allow the simulator to eavesdrop the queries of some or all parties. However, this weakens the power of the simulator considerably, and most of our impossibility proofs for the standard model of cryptography would hold for this model with only minor changes. Thus the strength of the simulator means a certain weakness in our positive results, but the need for this is another indicator that the impossibility problems are strong.

3 Informal Overview of the Impossibility Proofs

In this section, we present our results as proof sketches with a minimum amount of notation. Later we define more notation and precise assumptions, and then extend the proof sketches to full proofs. A reader with a specific Dolev-Yao model in mind and not interested in the generalization to "arbitrary Dolev-Yao-like models" should be able to see already in this section that the results could be instantiated for that model.

In the following, messages may occur in several representations, which we distinguish by superscripts. We write terms in the Dolev-Yao sense without superscript, e.g., $h := \mathsf{hash}(m)$ for a hash term. The real cryptographic versions get a superscript $^{\mathsf{r}}$, e.g., $h^{\mathsf{r}} := \mathsf{hash}^{\mathsf{r}}(m^{\mathsf{r}})$ for the corresponding real bitstring, computed by applying a real hash function $\mathsf{hash}^{\mathsf{r}}$ to the real representation m^{r} of m. The users and adversaries may concretely address the terms/bitstrings in yet another way when interacting with the real or ideal functionality (hopefully indistinguishably, hence we need only one notation); we write these representations with superscripts ^{u} for honest user u and $^{\mathsf{a}}$ for the adversary. (Using the actual terms as these representations is a special case.) In the figures we write H_u for the actual user u, which is a part of the global H in Figure 1.

3.1 Scenarios with Payloads

Our first scenario in Figure 2 demonstrates that the real hash function $\mathsf{hash}^{\mathsf{r}}$ must at least be collision-resistant in order to offer a sound implementation of a Dolev-Yao model with hashes and payloads. This is not surprising, but we need the collision resistance in the next proofs. The proof idea is that otherwise

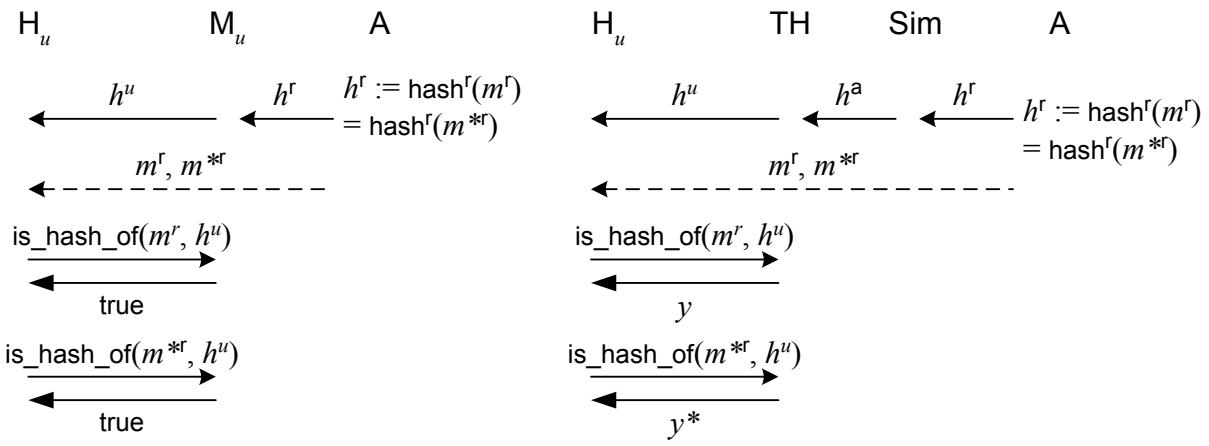

Fig. 2. Counterexample with payload hashing for not collision-resistant hash function.

the adversary can find two colliding payloads m^{r} and $m^{*\mathsf{r}}$ and send their hash h^{r} to an honest party u. It also exchanges m^{r} and $m^{*\mathsf{r}}$ secretly with the user u outside the system. Recall that such outside exchanges are allowed for chosen-message attacks etc. and that the notion of BRSIM/UC considered here does not allow the simulator to learn or change them, i.e., it must produce an overall indistinguishable view for H and A; we denote these outside exchanges by dashed arrows in the interaction figures.

Then user u uses the ideal or real functionality to check whether the message received through the system is the hash of both payloads. In the real system (on the left in all our interaction figures), the answer is true both times by the choice of h^{r}. However, the ideal collision freeness of the ideal system (on the right in all our interaction figures) does not allow this; hence the ideal and the real system are distinguishable.

The major challenge in formalizing this proof sketch is in the treatment of payloads, because most Dolev-Yao models do not put the actual payloads into the terms. Below we make precise assumptions on this treatment and define the ideal collision freeness, and then turn this sketch into a proof.

Our second scenario in Figure 3 demonstrates that even with a collision-resistant function hash^r, a sound implementation of a Dolev-Yao model with hashes and payloads is impossible if the ideal Dolev-Yao functionality offers ideal secrecy. By ideal secrecy we mean that an adversary who obtains the hash of an otherwise unknown term cannot do better than comparing this hash with self-made hashes of guessed terms. The scenario is that an honest party u selects a random payload m^r of length $2k$ (where k is the security parameter), sends it to the adversary outside the system, and sends the hash of this payload to the adversary through the ideal or real system. From the real system, the adversary gets the real hash $h^r := \mathsf{hash}^r(m^r)$, tests whether this is indeed the correct hash value of the payload, and tells the result to u outside the system. By the ideal secrecy, the simulator Sim for the ideal system cannot find out more about m^r than excluding polynomially many guesses. Using the collision resistance of the real hash function, we show that Sim can consequently only guess h^r with negligible probability, and thus cannot simulate this scenario correctly.

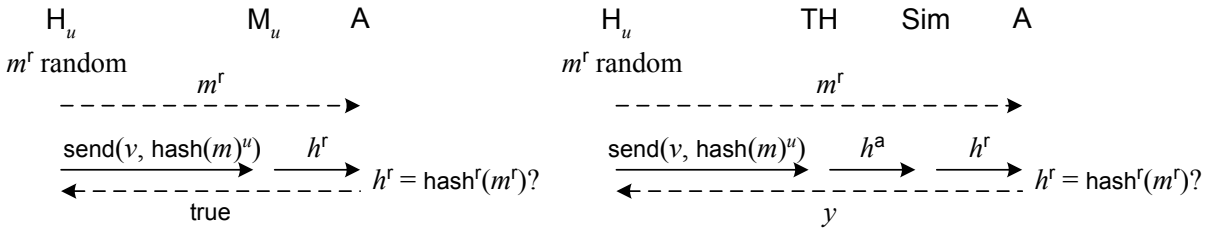

Fig. 3. Counterexample with payload hashing and ideal secrecy for collision-resistant hash function.

The technical difficulties with the full proof for this scenario lie in an appropriate formulation of the ideal secrecy, independent of one specific Dolev-Yao model.

Our third scenario in Figure 4 demonstrates that omitting the ideal secrecy requirement does not help as long as the real hash function is shortening as well as collision-resistant, and thus one-way. Here the real adversary agrees on a random payload m^r with an honest user u outside the system, and sends its hash h^r to u through the system. The user tests, using the ideal or real functionality, whether the obtained message is indeed a hash of m^r. In the real system, the output is clearly true. The best way for the simulator to cause the same output from the ideal functionality would be to send the term $h = \mathsf{hash}(m)$ via TH in the first step. However, this would require guessing m^r and thus breaking the one-way property of hash^r.

The difficulty with the full proof for this scenario, besides the question of payload representations as in the first scenario, is that $\mathsf{hash}(m)$ is not the only term that causes the output true. For instance, $\mathsf{D}(\mathsf{E}(\mathsf{hash}(m)))$ or $\mathsf{hash}(\mathsf{D}(\mathsf{E}(m)))$ for en- and decryption operators E and D are other such terms. We therefore have to be careful in how we can argue that every successful strategy for the simulator really leads to a successful algorithm that extracts m^r and thus breaks the one-way property.

3.2 Scenarios without Payloads

After showing that payloads and hashes in Dolev-Yao models lead to comprehensive impossibility results for secure realizations in the sense of BRSIM/UC, we consider restricted Dolev-Yao models without payloads. However, as long as this is the only restriction, we still prove impossibility. The basic proof idea is to let the users and the adversary simulate payloads by encoding them into the structure of long terms. Concretely, we use a list of $2k$ nonces and encode a payload as a bit vector $\vec{b}$ that selects a sublist of these nonces. Instead of nonces, any other type can be used of which one can generate $2k$ instances that are ideally different, e.g., keys.

With a scenario similar to Figure 2, only adding the random choice of the nonces for the encoding, we show that a hash function must be collision-resistant on these bit vectors in order to offer a BRSIM/UC-sound implementation of a Dolev-Yao model with hashes and lists of nonces. Then with a scenario similar to Figure 3, we show that if ideal secrecy is offered no sound implementation exists at all. With a scenario similar to Figure 4, we show that even without ideal secrecy, no sufficiently shortening hash

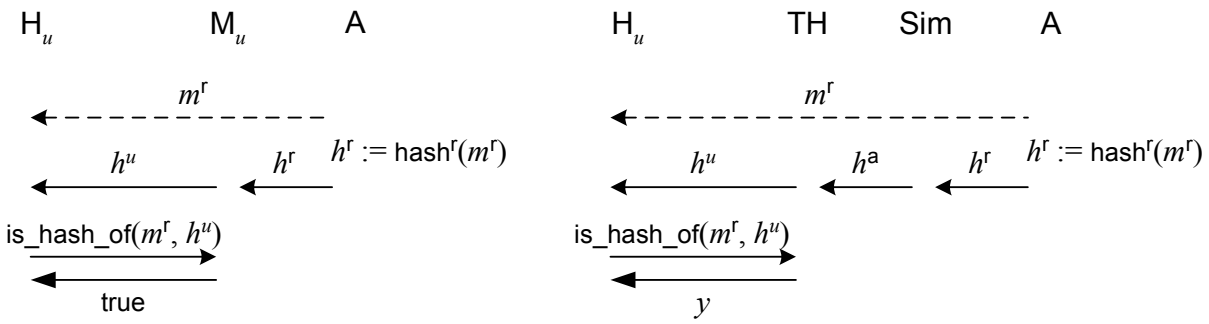

Fig. 4. Counterexample with payload hashing for shortening hash function.

function, in particular one whose output length depends only on the security parameter, yields a sound implementation.

3.3 Probabilistic Hash Functions

Finally, we sketch why the scenarios show impossibility also for probabilistic hashing: The real hash function hash^r can simply be probabilistic now. On the user side, checking whether a received term or message is the hash of a known message is already written $\mathsf{is\_hash\_of}$; this is suitable also to express the new verification procedure. On the adversary side in Figure 3, the test "$h^r = \mathsf{hash}^r(m^r)$?" must be replaced by a call to the real verification algorithm. Everything else remains the same. For readability, we do not make these extensions in the following formal part, but stick to the standard free hash operator.

4 Assumptions on Dolev-Yao Models for our Impossibility Results

As explained in the introduction, we would like to work out the impossibility proofs sketched in Section 3 not only for one specific Dolev-Yao model (then we could simply use an arbitrary definition from the literature and would not need our own definitions), but for all of them. However, "all Dolev-Yao models" is not a notion that anyone tried to formalize before. Hence we now characterize Dolev-Yao models and their realizations by weak rigorous requirements. In other words, we define common properties that are fulfilled by all standard Dolev-Yao models, and hopefully also by all other conceivable variants that might count as Dolev-Yao models. This makes our impossibility results as strong as possible.

4.1 Minimum Assumptions on a Dolev-Yao Model with Hashes

In this section we describe the functionaliy that we assume every Dolev-Yao system with hashes offers. We start with the basic notions of terms, including a hash operator. Recall that hash is essentially a free operator in the term algebra of typical Dolev-Yao models. However, we do not define this strong freeness, but only a weaker property, ideal collision freeness (both because this makes our results stronger, and because not all Dolev-Yao models are actually defined as initial models of an equational specification).

Definition 1 (Terms of a Dolev-Yao Model with Hashes). *We require that we can derive definitions of the following concepts from a Dolev-Yao model with hashes:*

a. A set Terms denoting the overall set of valid terms. We speak of atoms *and* operators *denoting the potential leaves and inner nodes, respectively, of the terms considered as trees. The terms, atoms and operators may be typed. There is an equivalence relation "$\equiv$" on Terms. We call $(Terms, \equiv)$ the term algebra.*[4]

b. A unary operator hash*, which fulfils* ideal collision freeness*, i.e.,* $\mathsf{hash}(t) \equiv \mathsf{hash}(t') \Rightarrow t \equiv t'$ *for all* $t, t' \in Terms$.

[4] Clearly syntactic term equality "=" implies equivalence. Typically "$\equiv$" is constructed from cancellation rules.

c. *A set Hashable_Terms ⊆ Terms of the terms that are valid operands of the operator* hash*. We speak of a model with* unrestricted hashing *if Hashable_Terms = Terms.*
d. *A list operator (possibly implemented by repeated pairing in the original syntax). Two lists are equivalent iff all their corresponding elements are.*

◇

Next we define some minimum actions that the users and the adversary can carry out on the terms, and the results of these actions. In our context, this is the basis for showing that our impossibility scenarios are at least executable in every Dolev-Yao model (which probably nobody doubted).

While our notion of term representations t^u for individual users is certainly more general than notions that may be familiar to some readers, and thus can only strengthen our impossibility results, let us briefly motivate how it relates to such notions: An important concept in Dolev-Yao models is that of terms t constructible for some user u or the adversary (by applying operators and cancellation rules to previously known messages); however, the syntax for this concept varies considerably. Some high-level representations simply use t itself in the protocol representations (e.g., "$\mathsf{hash}(m)$" even when someone who does not know m forwards this term). More detailed representations, e.g., in CSP or π-calculus, typically use the concepts of variables inherent to these calculi, usually by matching received messages with a pattern describing the expected message format, and then using the pattern variables in subsequent message constructions. The syntax explicitly made for BRSIM/UC of the Dolev-Yao-style model in [3] uses local variables called handles and explicit parsing of received messages. The syntax from all these models can easily be mapped to that in our following definition.

We do not need a full definition of how a user acquires term representations. However, we define that terms can be sent and that the ideal adversary controls the network as usual in Dolev-Yao models. Furthermore we define that users can hash terms and compare whether one term is the hash of another term. In some, but not all, Dolev-Yao models this comparison can be made by using a general equality operator corresponding to the term equivalence $\equiv$.

Definition 2 (Actions on a Dolev-Yao Model with Hashes). *Users and the ideal adversary can make at least the following inputs into the ideal functionality of a Dolev-Yao model with hashes, with the described results.*

a. *If an honest user u inputs* $\mathsf{send}(v, t^u)$ *for a term representation t^u, this leads to an output* $\mathsf{receive}(u, v, t^{\mathsf{a}})$ *for the adversary.*
b. *If the adversary inputs* $\mathsf{send}(u, v, t^{\mathsf{a}})$ *for a term representation t^{a}, this leads to an output* $\mathsf{receive}(u, t^v)$ *for user v (i.e., the adversary impersonates u).*
c. *If a user u (honest or the adversary represented by $u = \mathsf{a}$) has a term representation t^u, then it also has a representation for the term* $\mathsf{hash}(t)$*. (Typically this is something like the string "*$\mathsf{hash}(t^u)$*".)*
d. *An input* $\mathsf{is\_hash\_of}(t^u, h^u)$ *by a user u (honest or* a*) leads to a Boolean output y for user u with $y =$* true *iff $h \equiv \mathsf{hash}(t)$.*

◇

4.2 Payload Assumptions

All Dolev-Yao models in real proof tools have at least payload messages, nonces, and keys as atoms. However, as payloads are particularly problematic in simulations and some protocol classes do not need general payloads, we define Dolev-Yao models with and without them. A payload m models an application message, i.e., its cryptographic realization m^{r} can be an arbitrary bitstring; examples are emails, payment messages, and digital pictures. In this sense, our scenarios in Section 3 are perfectly natural: The users and the adversary select payloads as arbitrary bitstrings. However, the internal representation of payloads in the terms in Dolev-Yao proof tools is usually a constant supply of payload names or a nonce-like construction of fresh names. We therefore assume that the full ideal functionality maintains a translation table between the real payloads that occur in a system execution and their internal representations.[5]

[5] As an additional motivation for this assumption, recall that we want to compare the Dolev-Yao model and its cryptographic realization in the sense of BRSIM/UC. Thus they must offer the same syntactic user interfaces,

Definition 3 (Payloads in Dolev-Yao Models in the BRSIM/UC Setting). *A* Dolev-Yao model with payloads *allows us to derive a type (subset)* payload *in the set Terms. In every execution, every occurring payload term has a fixed realization m^{r}, and $m^{\mathsf{r}} = m'^{\mathsf{r}}$ implies $m \equiv m'$. The range of payload realizations m^{r} is at least $\{0,1\}^{2k}$. A real payload m^{r} can always be used as an input representation m^u by every user.* ◇

We now consider how secret a hashed term is in a Dolev-Yao model when the adversary learns its hash. We only need this in our second scenario, where we want to show that an adversary receiving a (representation of) a term $t = \mathsf{hash}(m)$ containing a payload m cannot get significant information about the real payload m^{r} and thus its real hash. In normal Dolev-Yao models, the hash operator is free, and thus there is no inverse operator that the adversary can use to extract m, nor a sequence of such operators. In addition, in many Dolev-Yao models one would represent the initial situation where the adversary does not know m by not giving the adversary any representation of m, thus excluding any possibility that the adversary guesses m. With such a strong assumption the impossibility proof would be easy. However, we allow the more realistic case that the adversary might guess payloads (as, e.g., in [3]). Furthermore, we only make the minimum assumption that payloads are secret in hash terms except for this guessing.

Definition 4 (Ideal Secrecy of New Payloads). *A Dolev-Yao model with hashes offers* ideal secrecy of new payloads *iff the following holds: If user u inputs $\mathsf{send}(v, h^u)$ where $h = \mathsf{hash}(m)$ for a newly chosen payload m^{r} (i.e., one that was not input to the Dolev-Yao model before), then the ideal adversary, from its output $\mathsf{receive}(u, v, h^{\mathsf{a}})$ and without further involvement of the user H, cannot obtain more information about m^{r} than by learning for x bitstrings m'^{r} whether $m'^{\mathsf{r}} = m^{\mathsf{r}}$ (in addition to its a-priori information), if it interacts at most x times with the ideal system and thus in particular if it runs in time x.* ◇

Finally, we define the weak freeness property of Dolev-Yao hashes that we need for the third scenario. Essentially this is that without knowing a payload m or its real representation m^{r} one cannot construct a term equivalent to $\mathsf{hash}(m)$. Like other definitions of "knowledge" in cryptography, this is done by defining that the capability to construct such a term implies the capability to find out m^{r}. This reduction is done constructively by an extractor algorithm.

Definition 5 (Minimum Non-Constructibility of Unknown Payload Hashes). *A Dolev-Yao model with hashes offers* minimum non-constructibility of unknown payload hashes *if there exists a polynomial-time algorithm* Ext*, called extractor, such that the following holds: If the ideal adversary (for simplicity at the system start) makes a sequence of inputs and then sends a term t to an honest user such that $t \equiv \mathsf{hash}(m)$ for a payload m, then the extractor, given the transcript of the ideal adversary's in- and outputs, outputs m^{r}.* ◇

For Dolev-Yao models with well-defined and constructible normalizations of terms, the extractor is essentially this normalization: It constructs t and the relation of payload terms and their representations from the transcript (typically the transcript is simply of the form "$\mathsf{send}(v, t^{\mathsf{a}})$" where payloads in t^{a} are in their real representation) and normalizes t; the result is $\mathsf{hash}(m)$, from which m^{r} can be looked up. This clearly holds for typical Dolev-Yao models that only have constructors and destructors like encryption and decryption. It gets more complex in Dolev-Yao models with algebraic operations like XOR; however, specifically XOR is known not to be realizable in BRSIM/UC [48].

4.3 Nonce List Assumptions

For the case without payloads, our scenarios use lists of nonces. We therefore define what we assume about nonces (lists are already in Definition 1). The first assumption is extremely simple and normal, except that some basic Dolev-Yao models only allow a fixed number of nonces, while we need at least $2k$ (as does every Dolev-Yao model suitable for arguing about an unbounded number of sessions).

i.e., in- and output formats. This holds for all definition variants of BRSIM/UC. In particular, in Figure 1 this is the interface between TH or $\mathsf{M}_1, \ldots, \mathsf{M}_n$, respectively, and the entirety of honest users H. In [7], it is the input and output formats of the ideal and real functionality. Syntactically different user interfaces would either simply prevent the same users from using alternatively the real or the ideal system, or lead to trivial distinguishability.

Definition 6 (Nonces in Dolev-Yao Models). *A* Dolev-Yao model with nonce lists *allows us to derive a type (subset)* nonce *in the set Terms. Every participant can use, or explicitly generate, at least* $2k$ *new nonces (we do not need a fixed syntax for this generation); such nonces are pairwise not equivalent.* ◇

The next definition extends the ideal secrecy of hashed terms, which we earlier defined only for new payloads, to new lists of nonces. More precisely, we define that an ideal hash term does not divulge which of the many potential sublists of a list of nonces was hashed. (We make these weak special assumptions to strengthen the impossibility results, and to avoid complex considerations about prior knowledge in the general case.)

Definition 7 (Ideal Secrecy of New Nonce Lists). *A Dolev-Yao model with hashes offers* ideal secrecy of new nonce lists *iff the following holds: Let user* u *generate* $2k$ *new nonces* $\vec{n} = (n_1, \ldots, n_{2k})$ *and potentially send them to the adversary, select a random bit vector* $\vec{b} = (b_1, \ldots, b_{2k}) \xleftarrow{R} \{0,1\}^{2k}$*, and input* $\mathsf{send}(v, (\mathsf{hash}(\vec{b} \odot \vec{n}))^u)$ *where* $\vec{b} \odot \vec{n}$ *denotes the sublist consisting of the nonces* n_i *with* $b_i = 1$*. Then the ideal adversary, from its output* $\mathsf{receive}(u, v, h^{\mathsf{a}})$ *and without further interaction with the user* u*, cannot obtain more information about* $\vec{b}$ *than by learning for* x *bit vectors* $\vec{b'}$ *whether* $\vec{b'} = \vec{b}$*, if it interacts at most* x *times with the ideal system and thus in particular if it runs in time* x*.* ◇

Finally, we define that the ideal adversary cannot construct a hash over a sublist of nonces without knowing which sublist of the nonces it is using.

Definition 8 (Minimum Non-Constructibility of Unknown Nonce-list Hashes). *A Dolev-Yao model with hashes offers* minimum non-constructibility of unknown nonce-list hashes *if there exists a polynomial-time algorithm* Ext*, called extractor, such that the following holds: If the ideal adversary (for simplicity at the system start) makes a sequence of inputs and then sends a list* $\vec{n}$ *of* $2k$ *pairwise different nonces and a term* h *to an honest user such that* $h \equiv \mathsf{hash}(\vec{b} \odot \vec{n})$*, then the extractor, given the transcript of the ideal adversary's in- and outputs, outputs* $\vec{b}$*.* ◇

Again, the existence of such an extractor is clear for Dolev-Yao models with a normalization algorithm because the term $\mathsf{hash}(\vec{b} \odot \vec{n})$ cannot be further reduced and is thus the normal form of every equivalent term. Given the overall list of nonces $\vec{n}$, which the ideal adversary sent separately, the selection of nonces in this term and thus $\vec{b}$ can be read off.

4.4 Minimum Assumptions on a Cryptographic Realization

A general characteristics of real systems is that they are distributed. This means that each participant u has its own machine, here called M_u, and the machines are only connected by channels that offer well-defined possibilities for observations and manipulations by a real adversary. Specifically for the realization of Dolev-Yao models with hashes, we make the following (natural) minimum assumptions in the standard model of cryptography: Real channels are insecure; the input to send a term t leads to actual sending of a bitstring t^{r}; and hash terms are realized by applying a fixed (hash) function to the realization of the contained terms.

Definition 9 (Realization of a Dolev-Yao Model with Hashes). *In a* realization of a Dolev-Yao model with hashes in the standard model of cryptography*, an input* $\mathsf{send}(v, t^u)$ *to a machine* M_u *releases a bitstring* t^{r} *to the real adversary, such that within one execution of the system* $t \equiv t' \Rightarrow t^{\mathsf{r}} = t'^{\mathsf{r}}$ *for all terms* t, t'*. There must be a deterministic, polynomial-time function* $\mathsf{hash}^{\mathsf{r}}$ *such that* $(\mathsf{hash}(t))^{\mathsf{r}} = \mathsf{hash}^{\mathsf{r}}(t^{\mathsf{r}})$ *for all* $t \in$ *Hashable_Terms. An input* $\mathsf{is\_hash\_of}(t^u, h^u)$ *to a machine* M_u *leads to the output* true *iff* $\mathsf{hash}^{\mathsf{r}}(t^{\mathsf{r}}) = h^{\mathsf{r}}$*.*

For nonces, there must be a probabilistic polynomial-time algorithm $\mathsf{G_n}$ *that is used to generate* n^{r} *when it is needed for a new nonce* n*, and* $2k$ *executions of* $\mathsf{G_n}$ *must yield pairwise different results* n^{r}_1*,* $\ldots$*,* n^{r}_{2k} *with overwhelming probability.* ◇

In realizations with type tagging we can consider an original cryptographic hash function together with the type tag as $\mathsf{hash}^{\mathsf{r}}$. Note that we made no assumptions on the cryptographic properties of $\mathsf{hash}^{\mathsf{r}}$ and

only a weak one on G_n; we will show that neither "good" nor "bad" realizations lead to soundness in the sense of BRSIM/UC. [6]

5 Details of the Impossibility Proofs

We now present the missing details for the impossibility proof sketches in Section 3, using the definitions from Section 4.

5.1 Unsoundness of Dolev-Yao Models with Payloads

The first scenario from Section 3.1 becomes the following lemma. Its proof is contained in the long version [54].

Lemma 1. *(Collision Resistance of the Real Hash Function) If a Dolev-Yao model with hashes and payloads (Definitions 1 to 3) has a realization in the standard model of cryptography (Definition 9) that is secure in the sense of BRSIM/UC, then the hash function* hash^r *in this realization is collision-resistant. For simplicity we define here that a collision for security parameter* k *consists of two messages of length* $2k$*.* □

The second scenario from Section 3.1 together with this lemma gives us the following theorem.

Theorem 1. *(Unsoundness of Dolev-Yao Models with Hashes and Ideal Secrecy of New Payloads) No Dolev-Yao model with hashes and ideal secrecy of new payloads (Definitions 1 to 4) has a realization in the standard model of cryptography (Definition 9) that is secure in the sense of BRSIM/UC.* □

Proof. Assume that a Dolev-Yao model and a realization as specified in the theorem exist. By Lemma 1, the hash function hash^r in the realization must be collision-resistant. Then $\delta(k) := \max_{h^\mathsf{r} \in \{0,1\}^*}(\mathsf{Pr}[\mathsf{hash}^\mathsf{r}(m^\mathsf{r}) = h^\mathsf{r} :: m^\mathsf{r} \xleftarrow{R} \{0,1\}^{2k}])$ is negligible (as a function of k), because otherwise two random messages of length $2k$ are a collision with not negligible probability. We elaborate our second scenario in Figure 3: The user H_u chooses the payload as $m^\mathsf{r} \xleftarrow{R} \{0,1\}^{2k}$. By Definitions 2 and 3, the adversary and the user can indeed act as described in Section 3.1, and by Definition 9, the output for A in the real system is $\mathsf{hash}^\mathsf{r}(m^\mathsf{r})$. In the ideal system, the simulator Sim gets an output $\mathsf{receive}(u, v, h^\mathsf{a})$ from the Dolev-Yao model TH and has to produce a string h^r for the adversary. For indistinguishability, this string must fulfill $h^\mathsf{r} = \mathsf{hash}^\mathsf{r}(m^\mathsf{r})$ with overwhelming probability.

Definition 4 is applicable and implies that Sim (which acts as the ideal adversary here), with x calls to TH, cannot obtain more information about m^r than by learning for x bitstrings m'^r whether $m'^\mathsf{r} = m^\mathsf{r}$. As m^r is uniformly random, the probability that Sim hits $m'^\mathsf{r} = m^\mathsf{r}$ in this process is $x/2^{2k}$, where x is polynomial because Sim is polynomial-time. Thus this probability is negligible. In the other case, the optimal choice of h^r for Sim is the most likely hash value over the remaining $2^{2k} - x$ possible payloads. The probability that this value is correct is at most $\delta(k)2^{2k}/(2^{2k} - x)$. This is negligible because x is polynomial. ■

Next we consider the case without secrecy of hashed terms, but with the additional assumption that the output length of the real hash function depends only on the security parameter, not on the input length. (Weaker definitions of significantly shortening hash functions would also suffice.) The third scenario from Section 3.1 together with Lemma 1 gives us the following theorem.

Theorem 2. *(Unsoundness of Dolev-Yao Models with Hashes and Payloads without Secrecy) No Dolev-Yao model with hashes and payloads, even without ideal secrecy, but with minimum non-constructibility of unknown payload hashes, (Definitions 1 to 3 and 5) has a realization in the standard model of cryptography (Definition 9) that is secure in the sense of BRSIM/UC and where the real hash function is shortening. For simplicity, we require that the range of a shortening hash function is* $\{0,1\}^k$*.* □

[6] In computational considerations about hash^r we allow hash^r to depend on the security parameter k, which is fixed in each system execution. To allow collision resistance in the sense of the typical cryptographic definition, it should even depend on a key pk chosen at the beginning of each system execution; our proofs could easily be adapted to this case.

Proof. Assume that a Dolev-Yao model and a realization as specified in the theorem exist. By Lemma 1, the hash function $\mathsf{hash}^{\mathsf{r}}$ in the realization must be collision-resistant. As $\mathsf{hash}^{\mathsf{r}}$ is also shortening, it is one-way. (Otherwise the following algorithm finds a collision with not negligible probability: Select a random payload $m^{\mathsf{r}} \xleftarrow{\mathcal{R}} \{0,1\}^{2k}$, use the assumed inversion algorithm $\mathsf{A}_{\mathsf{owf}}$ to find a preimage m'^{r} of $\mathsf{hash}^{\mathsf{r}}(m^{\mathsf{r}})$, and output m^{r} and m'^{r} if they are unequal. This holds because all payloads, except less than 2^k and thus negligibly many, collide with another one. If $\mathsf{A}_{\mathsf{owf}}$ succeeds for such a payload m^{r}, then with probability at least $1/2$ we have $m'^{\mathsf{r}} \neq m^{\mathsf{r}}$.)

We now elaborate our third scenario in Figure 4. By Definitions 2 and 3, the adversary and the user can indeed act as described in Section 3.1, and by Definition 9 the output for H_u in the real system is indeed true. By the assumption of the proof, the simulator can achieve the same result in the ideal system with overwhelming probability. Hence it makes an input $\mathsf{send}(v, u, h^{\mathsf{a}})$ where, by Definition 2, $h \equiv \mathsf{hash}(m)$ for the term m that is realized as m^{r}. Definition 5 is applicable to our scenario and essentially states that Sim, which acts as the ideal adversary, must know m^{r} for this. More precisely, we use the postulated extractor Ext to extract m^{r} from the transcript of Sim whenever Sim is successful. This gives us an inversion algorithm $\mathsf{A}_{\mathsf{owf}}$ for the function $\mathsf{hash}^{\mathsf{r}}$ that succeeds with not negligible probability, in contradiction to the one-wayness of $\mathsf{hash}^{\mathsf{r}}$. Concretely, $\mathsf{A}_{\mathsf{owf}}$ is the combination of TH, Sim and Ext. This is indeed a non-interactive algorithm as required in the definition of one-wayness: Sim initially gets one input h^{r} and TH has no input so far. Then Sim and TH interact with each other, but interaction with H or A would be distinguishable from the real system. ∎

5.2 Unsoundness without Payloads

Now we work out the scenarios for restricted Dolev-Yao models without payloads. As sketched in Section 3.2, we proceed similar to the scenarios with payloads, letting the users and the adversary replace payloads by bit vectors that select sublists of nonces. For this, we first define collision resistance and one-wayness with respect to the bit vectors.

Definition 10 (Bit-vector Collision Resistance and One-Wayness). *Let a Dolev-Yao model with hashes and a realization in the standard model of cryptography with the hash function* $\mathsf{hash}^{\mathsf{r}}$ *be given (Definitions 1, 2, and 9). We say that* $\mathsf{hash}^{\mathsf{r}}$ *is* bit-vector collision-resistant *if every polynomial-time adversary can only find a list* $\vec{n}^{\mathsf{r}} = (n_1^{\mathsf{r}}, \ldots, n_{2k}^{\mathsf{r}})$ *of* $2k$ *pairwise different real nonces and bit vectors* $\vec{b} \neq \vec{b^*} \in \{0,1\}^{2k}$ *with* $\mathsf{hash}^{\mathsf{r}}(\vec{b} \odot \vec{n}^{\mathsf{r}}) = \mathsf{hash}^{\mathsf{r}}(\vec{b^*} \odot \vec{n}^{\mathsf{r}})$ *with negligible probability, where* $\vec{b} \odot \vec{n}^{\mathsf{r}}$ *for a bit vector* $\vec{b} = b_1, \ldots, b_{2k}$ *denotes the sublist consisting of the nonces* n_i^{r} *with* $b_i = 1$.

We say that $\mathsf{hash}^{\mathsf{r}}$ *is* bit-vector one-way *if every polynomial-time algorithm* $\mathsf{A}_{\mathsf{owf}}$*, on input* $h^{\mathsf{r}} := \mathsf{hash}^{\mathsf{r}}(\vec{b} \odot \vec{n}^{\mathsf{r}})$ *for random* $\vec{b} \xleftarrow{\mathcal{R}} \{0,1\}^{2k}$ *and real nonces generated with* G_{n}*, can only output a bit vector* $\vec{b^*} \in \{0,1\}^{2k}$ *with* $h^{\mathsf{r}} = \mathsf{hash}^{\mathsf{r}}(\vec{b^*} \odot \vec{n}^{\mathsf{r}})$ *with negligible probability.* ◇

Lemma 2. *(Bit-vector Collision Resistance of the Real Hash Function) If a Dolev-Yao model with hashes and nonces (Definitions 1, 2, and 6) where Hashable_Terms contains at least all lists of up to* $2k$ *nonces has a realization in the standard model of cryptography (Definition 9) that is secure in the sense of BRSIM/UC, then the hash function* $\mathsf{hash}^{\mathsf{r}}$ *in this realization is bit-vector collision-resistant.* □

The proofs of this lemma and the two following theorems are postponed to the long version [54].

Theorem 3. *(Unsoundness of Dolev-Yao Models with Hashes and Ideal Secrecy of New Nonce Lists) No Dolev-Yao model with hashes and ideal secrecy of new nonce lists (Definitions 1, 2, 6, and 7) has a realization in the standard model of cryptography (Definition 9) that is secure in the sense of BRSIM/UC.* □

Theorem 4. *(Unsoundness of Dolev-Yao Models with Hashes and Nonce Lists without Secrecy) Let a Dolev-Yao model with hashes be given whose set Hashable_Terms contains at least all lists of up to* $2k$ *nonces, and where minimum non-constructibility of unknown nonce-list hashes holds (Definitions 1, 2, 6, and 8). Then no cryptographic implementation in the standard model of cryptography (Definition 9) with a shortening real hash function is sound in the sense of BRSIM/UC.* □

6 Soundness Results

In this section we show that Dolev-Yao-style hashes can be proven sound in the random oracle model, and under specific restrictions on the usage of hash functions or their properties in the ideal system even in the standard model of cryptography.

6.1 Soundness of Dolev-Yao Models with Hashes in the Random Oracle Setting

The first soundness result states that normal Dolev-Yao models without specific restrictions can be proven sound in the random oracle model. As an overall result for an operator-rich Dolev-Yao model with hashes, this requires an underlying Dolev-Yao model with the other usual cryptographic operators and a realization secure in the sense of BRSIM/UC. Hence we have to use that of [3]. However, what happens specifically with the hashes can be explained well without specific notation from [3]. We sketch this in this section, leaving more details to the long version of this paper [54].

The Dolev-Yao functionality is that of a free hash operator with unrestricted hashing and with ideal secrecy in the typical sense that the adversary, upon learning a hash value, has no deconstruction operator or other ways to obtain information about the contained term except by the input $\mathsf{is\_hash\_of}$ for comparing a message and a potential hash. The only additional power that the ideal adversary gets compared with honest users is to make hashes with unknown preimages, i.e., terms that could be written $\mathsf{hash}(?)$. The preimages will remain unknown forever; that this works in the realization is a consequence of the random oracle model.

In the cryptographic realization, the operator hash is essentially realized by the random oracle. The only addition is that the bitstrings are typed, i.e., the realization $\mathsf{hash}(t)^{\mathsf{r}}$ of a hash term is the pair $(\text{`}\mathsf{hash}\text{'}, \mathsf{RO}(t^{\mathsf{r}}))$ where 'hash' is a fixed string and RO abbreviates the result of a (stateful) random oracle call.

Our security claim is that this realization is as secure as this ideal Dolev-Yao-style system in the sense of BRSIM/UC in the random oracle model; see Section 2. The proof of the theorem can be found in the long version [54].

Theorem 5. *(Soundness of a Dolev-Yao Model with Hashes in the Random Oracle Model) A Dolev-Yao model with unrestricted hashing and secrecy of hashed terms can be securely implemented by a canonical cryptographic realization in the sense of BRSIM/UC in the random oracle model. Both the Dolev-Yao model and cryptographic realization are defined in detail in the long version [54].* □

6.2 Soundness Results in the Standard Model

Finally, we briefly present two restricted but still practically useful types of Dolev-Yao models with hashes that have secure realizations even in the standard model of cryptography. Both models allow the ideal adversary to construct hash terms with unknown preimages, i.e., terms $\mathsf{hash}(?)$. In contrast to the model in Section 6.1, the adversary can later provide a preimage for such a term. Both realizations require a collision-resistant hash function; in the first case the hash function must also be one-way.

The first type of Dolev-Yao model gives up the ideal secrecy, and can then work with a significant class of hashable terms. By the size of a term t we mean the number of nodes in the tree representation of t.

Theorem 6. *(Soundness Without Payloads or Secrecy for Constant-Sized Terms) A Dolev-Yao model without secrecy of hashed terms where terms in Hashable_Terms do not contain payloads and are at most of a constant size l can be securely realized in the sense of BRSIM/UC with arbitrary collision-resistant, one-way hash functions in the standard model of cryptography.* □

We believe that this theorem can be extended to terms that contain payloads, but only together with fresh nonces that remain secret, but the overhead of such a condition that must be defined over an overall system execution does not seem justified because the case is of limited usefulness: Such a nonce cannot be sent over an insecure channel, and thus unless a secret channel is available no other party can do anything with such a hash term, such as test if for correctness.

The second theorem offers the ideal secrecy of typical Dolev-Yao models of hashing, but only individual nonces can be hashed, as for instance in one-time signatures. Recall from the introduction that while this may seem surprising given that real hash functions (even probabilistic ones) do not offer perfect secrecy, it is correct because nonces are system-internal objects whose Dolev-Yao abstraction essentially only requires freshness and unguessability of the nonces as a whole.

Theorem 7. *(Soundness With Secrecy for Nonce Hashing) A Dolev-Yao model with secrecy of hashed terms and where the set Hashable_Terms contains only individual nonces can be securely realized in the sense of BRSIM/UC with arbitrary collision-resistant hash functions in the standard model of cryptography.* □

Similar to the random oracle case, for the precise model we rely on the existing Dolev-Yao model of [3] and extend it with hashes. The detailed models and sketches of both proofs are given in the long version [54].

7 Conclusion

We have investigated whether Dolev-Yao models with hashes or one-way functions can be realized in the sense of BRSIM/UC, i.e., such that the Dolev-Yao model is regarded as an ideal functionality that is securely implemented by its realization. We have shown that this is not possible for the standard type of such Dolev-Yao models where hashing is a free operator. This impossibility result holds for all polynomial-time computable functions in the role of the real hash function.

We then considered restrictions and extensions of the Dolev-Yao model or its ideal properties that have a potential to simplify simulations. For these, we obtained additional BRSIM/UC impossibility results: First, modeling probabilistic hashing makes no difference. Secondly, it does not help if no payloads can be hashed, but only cryptographic terms (in fact, lists of nonces are sufficient). Thirdly, even if we give up the ideal secrecy property of hashes (retaining the ideal collision freeness so that the model is still reasonable), we obtain BRSIM/UC impossibility for all realizations of the hash operator by polynomial-time computable functions whose output length is independent of the input length, and thus for all typical real hash functions. This is the first impossibility proof for a Dolev-Yao model that does not assume any ideal secrecy property.

On the positive side, we showed that a BRSIM/UC-sound realization of standard Dolev-Yao hashes is possible in the random oracle model. we also obtain BRSIM/UC soundness in the standard model of cryptography for two cases: One includes ideal secrecy, but only allows hashing of single nonces, e.g., for the use in one-time signatures. The other gives up ideal secrecy, but allows hashing of arbitrary cryptographic terms, i.e., terms without payloads, up to an arbitrary but constant size.

Acknowledgments. We thank Martín Abadi, Anupam Datta, Ante Derek, Cathy Meadows, John Mitchell, and Andre Scedrov for interesting discussions.

References

1. Michael Backes, Birgit Pfitzmann, and Michael Waidner. Limits of the reactive simulatability/UC of Dolev-Yao models with hashes. In *Proceedings of 11th European Symposium on Research in Computer Security(ESORICS)*, volume 4189 of *Lecture Notes in Computer Science*, pages 404–423. Springer, 2006.
2. Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
3. Michael Backes, Birgit Pfitzmann, and Michael Waidner. A composable cryptographic library with nested operations. In *Proc. 10th ACM CCS*, pages 220–230, 2003.
4. Birgit Pfitzmann and Michael Waidner. Composition and integrity preservation of secure reactive systems. In *Proc. 7th ACM CCS*, pages 245–254, 2000.
5. Birgit Pfitzmann and Michael Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *Proc. 22nd IEEE Symp. on Security & Privacy*, pages 184–200, 2001. More model details in IACR ePrint Report 2004/082 with M. Backes.
6. Michael Backes, Birgit Pfitzmann, and Michael Waidner. The reactive simulatability framework for asynchronous systems. *Information and Computation*, pages 1685–1720, 2007.

7. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. 42nd IEEE FOCS*, pages 136–145, 2001.
8. Michael Backes, Birgit Pfitzmann, and Michael Waidner. Symmetric authentication within a simulatable cryptographic library. In *Proc. 8th ESORICS*, volume 2808 of *LNCS*, pages 271–290. Springer, 2003.
9. Michael Backes and Birgit Pfitzmann. Symmetric encryption in a simulatable Dolev-Yao style cryptographic library. In *Proc. 17th IEEE CSFW*, pages 204–218, 2004.
10. Michael Backes, Birgit Pfitzmann, and Andre Scedrov. Key-dependent message security under active attacks - BRSIM/UC-soundness of symbolic encryption with key cycles. In *Proceedings of 20th IEEE Computer Security Foundation Symposium (CSF)*, 2007. Preprint on IACR ePrint 2005/421.
11. Michael Backes and Birgit Pfitzmann. A cryptographically sound security proof of the Needham-Schroeder-Lowe public-key protocol. In *Proc. 23rd Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 1–12, 2003.
12. Michael Backes. A cryptographically sound dolev-yao style security proof of the Otway-Rees protocol. In *Proceedings of 9th European Symposium on Research in Computer Security (ESORICS)*, volume 3193 of *Lecture Notes in Computer Science*, pages 89–108. Springer, 2004.
13. Michael Backes and Markus Duermuth. A cryptographically sound Dolev-Yao style security proof of an electronic payment system. In *Proceedings of 18th IEEE Computer Security Foundations Workshop (CSFW)*, pages 78–93, 2005.
14. Michael Backes and Birgit Pfitzmann. On the cryptographic key secrecy of the strengthened Yahalom protocol. In *Proceedings of 21st IFIP International Information Security Conference (SEC)*, pages 233–245, 2006.
15. Michael Backes, Sebastian Moedersheim, Birgit Pfitzmann, and Luca Vigano. Symbolic and cryptographic analysis of the secure WS-ReliableMessaging Scenario. In *Proceedings of Foundations of Software Science and Computational Structures (FOSSACS)*, volume 3921 of *Lecture Notes in Computer Science*, pages 428–445. Springer, 2006.
16. Michael Backes, Iliano Cervesato, Aaron D. Jaggard, Andre Scedrov, and Joe-Kai Tsay. Cryptographically sound security proofs for basic and public-key kerberos. In *Proceedings of 11th European Symposium on Research in Computer Security(ESORICS)*, volume 4189 of *Lecture Notes in Computer Science*, pages 362–383. Springer, 2006. Preprint on IACR ePrint 2006/219.
17. Michael Backes and Birgit Pfitzmann. Computational probabilistic non-interference. In *Proceedings of 7th European Symposium on Research in Computer Security (ESORICS)*, volume 2502 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2002.
18. Michael Backes and Christian Jacobi. Cryptographically sound and machine-assisted verification of security protocols. In *Proc. 20th Annual Symp. on Theoretical Aspects of Computer Science (STACS)*, volume 2607 of *LNCS*, pages 675–686. Springer, 2003.
19. Michael Backes, Birgit Pfitzmann, Michael Steiner, and Michael Waidner. Polynomial fairness and liveness. In *Proceedings of 15th IEEE Computer Security Foundations Workshop (CSFW)*, pages 160–174, 2002.
20. Michael Backes and Birgit Pfitzmann. Intransitive non-interference for cryptographic purposes. In *Proc. 24th IEEE Symp. on Security & Privacy*, pages 140–152, 2003.
21. Michael Backes. Quantifying probabilistic information flow in computational reactive systems. In *Proceedings of 10th European Symposium on Research in Computer Security (ESORICS)*, volume 3679 of *Lecture Notes in Computer Science*, pages 336–354. Springer, 2005.
22. Michael Backes and Birgit Pfitzmann. Relating symbolic and cryptographic secrecy. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2(2):109–123, 2005.
23. Peeter Laud. Secrecy types for a simulatable cryptographic library. In *Proc. 12th ACM CCS*, pages 26–35, 2005.
24. Christoph Sprenger, Michael Backes, David Basin, Birgit Pfitzmann, and Michael Waidner. Cryptographically sound theorem proving. In *Proc. 19th IEEE CSFW*, pages 153–166, 2006.
25. Martin Abadi and Phillip Rogaway. Reconciling two views of cryptography: The computational soundness of formal encryption. In *Proc. 1st IFIP TCS*, volume 1872 of *LNCS*, pages 3–22. Springer, 2000.
26. Martín Abadi and Jan Jürjens. Formal eavesdropping and its computational interpretation. In *Proc. 4th TACS*, pages 82–94, 2001.
27. Peeter Laud. Semantics and program analysis of computationally secure information flow. In *Proc. 10th European Symp. on Programming (ESOP)*, pages 77–91, 2001.
28. Daniele Micciancio and Bogdan Warinschi. Soundness of formal encryption in the presence of active adversaries. In *Proc. 1st Theory of Cryptography Conf. (TCC)*, volume 2951 of *LNCS*, pages 133–151. Springer, 2004.
29. Peeter Laud. Symmetric encryption in automatic analyses for confidentiality against active adversaries. In *Proc. 25th IEEE Symp. on Security & Privacy*, pages 71–85, 2004.

30. Ran Canetti and Jonathan Herzog. Universally composable symbolic analysis of mutual authentication and key exchange protocols. In *Proc. 3rd Theory of Cryptography Conf. (TCC)*, pages 380–403. Springer, 2006.
31. J. Mitchell, M. Mitchell, and A. Scedrov. A linguistic characterization of bounded oracle computation and probabilistic polynomial time. In *Proc. 39th FOCS*, pages 725–733, 1998.
32. John Mitchell, Mark Mitchell, Andre Scedrov, and Vanessa Teague. A probabilistic polynominal-time process calculus for analysis of cryptographic protocols. *ENTCS*, 47:1–31, 2001.
33. Russell Impagliazzo and Bruce M. Kapron. Logics for reasoning about cryptographic constructions. In *Proc. 44th IEEE FOCS*, pages 372–381, 2003.
34. Anupam Datta, Ante Derek, John Mitchell, Vitalij Shmatikov, and Matthieu Turuani. Probabilistic polynomial-time semantics for a protocol security logic. In *Proc. 32nd Intern. Colloquium on Automata, Languages and Programming (ICALP)*, volume 3580 of *LNCS*, pages 16–29. Springer, 2005.
35. Bruno Blanchet. A computationally sound mechanized prover for security protocols. In *Proc. 27th IEEE Symp. on Security & Privacy*, pages 140–154, 2006.
36. Michael Backes, Christian Jacobi, and Birgit Pfitzmann. Deriving cryptographically sound implementations using composition and formally verified bisimulation. In *Proceedings of 11th International Symposium on Formal Methods Europe (FME)*, volume 2391 of *Lecture Notes in Computer Science*, pages 310–329. Springer, 2002.
37. Michael Backes, Birgit Pfitzmann, and Michael Waidner. Low-level ideal signatures and general integrity idealization. In *Proceedings of 7th Information Security Conference (ISC)*, volume 3225 of *Lecture Notes in Computer Science*, pages 39–51. Springer, 2004.
38. Michael Backes, Markus Duermuth, Dennis Hofheinz, and Ralf Kuesters. Conditional reactive simulatability. In *Proceedings of 11th European Symposium on Research in Computer Security (ESORICS)*, volume 4189 of *Lecture Notes in Computer Science*, pages 424–443. Springer, 2006. Preprint on IACR ePrint 2006/132.
39. Michael Backes and Peeter Laud. Computationally sound secrecy proofs by mechanized flow analysis. In *Proceedings of 13th ACM Conference on Computer and Communications Security (CCS)*, pages 370–379, 2006.
40. Michael Merritt. *Cryptographic Protocols*. PhD thesis, Georgia Institute of Technology, 1983.
41. Lawrence Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Cryptology*, 6(1):85–128, 1998.
42. Bruno Blanchet. An efficient cryptographic protocol verifier based on Prolog rules. In *Proc. 14th IEEE CSFW*, pages 82–96, 2001.
43. David Basin, Sebastian Mödersheim, and Luca Viganò. OFMC: A symbolic model checker for security protocols. *Intern. Journal of Information Security*, 2004.
44. Flavio D. Garcia and Peter van Rossum. Sound computational interpretation of formal hashes. IACR Cryptology ePrint Archive 2006/014, January 2006.
45. Ran Canetti and Marc Fischlin. Universally composable commitments. In *Proc. CRYPTO 2001*, volume 2139 of *LNCS*, pages 19–40. Springer, 2001.
46. Ran Canetti and Hugo Krawczyk. Universally composable notions of key exchange and secure channels. In *Proc. EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 337–351. Springer, 2002.
47. Anupam Datta, Ante Derek, John Mitchell, Ajith Ramanathan, and Andre Scedrov. Games and the impossibility of realizable ideal functionality. In *Proc. 3rd Theory of Cryptography Conf. (TCC)*. Springer, 2006. To appear.
48. Michael Backes and Birgit Pfitzmann. Limits of the cryptographic realization of Dolev-Yao-style XOR. In *Proc. 10th ESORICS*, volume 3679 of *LNCS*, pages 178–196. Springer, 2005.
49. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. 1st ACM CCS*, pages 62–73, 1993.
50. Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *Proc. CRYPTO 97*, pages 455–469. Springer, 1997.
51. Ran Canetti, Daniele Micciancio, and Omer Reingold. Perfectly one-way probabilistic hash functions. In *Proc. 30th ACM STOC*, pages 131–140, 1998.
52. Andrew C. Yao. Theory and applications of trapdoor functions. In *Proc. 23rd FOCS*, pages 80–91, 1982.
53. Dennis Hofheinz and Jörn Müller-Quade. Universally composable commitments using random oracles. In *Proc. 1st Theory of Cryptography Conf. (TCC)*, volume 2951 of *LNCS*, pages 58–76. Springer, 2004.
54. Michael Backes, Birgit Pfitzmann, and Michael Waidner. Limits of the Reactive Simulatability/UC of Dolev-Yao models with hashes. IACR Cryptology ePrint Archive 2006/068, February 2006.