Foundations of Non-Malleable Hash and One-Way Functions

Alexandra Boldyreva[1], David Cash[1], Marc Fischlin[2], and Bogdan Warinschi[3]

[1] Georgia Institute of Technology, USA,
{aboldyre,cdc}@cc.gatech.edu,
[2] Darmstadt University of Technology, Germany,
marc.fischlin@gmail.com,
[3] University of Bristol, UK,
bogdan@cs.bris.ac.uk

Abstract. Non-malleability is an interesting and useful property which ensures that a cryptographic protocol preserves the independence of the underlying values: given for example an encryption $\mathcal{E}(m)$ of some unknown message m, it should be hard to transform this ciphertext into some encryption $\mathcal{E}(m^*)$ of a related message m^*. This notion has been studied extensively for primitives like encryption, commitments and zero-knowledge. Non-malleability of one-way functions and hash functions has surfaced as a crucial property in several recent results, but it has not undergone a comprehensive treatment so far. In this paper we initiate the study of such non-malleable functions. We start with the design of an appropriate security definition. We then show that non-malleability for hash and one-way functions can be achieved, via a theoretical construction that uses perfectly one-way hash functions and simulation-sound non-interactive zero-knowledge proofs of knowledge (NIZKPoK). We also discuss the complexity of non-malleable hash and one-way functions. Specifically, we show that such functions imply perfect one-wayness and we give a black-box based separation of non-malleable functions from one-way permutations (which our construction bypasses due to the "non-black-box" NIZKPoK based on trapdoor permutations). We exemplify the usefulness of our definition in cryptographic applications by showing that (some variant of) non-malleability is necessary and sufficient to securely replace one of the two random oracles in the IND-CCA encryption scheme by Bellare and Rogaway, and to improve the security of client-server puzzles.

1 Introduction

Motivation. Informally, non-malleability of some function f is a cryptographic property that asks that learning $f(x)$ for some x does not facilitate the task of generating some $f(x^*)$ so that x^* is related to x in some non-trivial way. This notion is especially useful when f is used to build higher-level multi-user protocols where non-malleability of the protocol itself is crucial (e.g., for voting or

auctioning). Non-malleability has been rather extensively studied for some cryptographic primitives. For example, both definitions as well as constructions from standard cryptographic assumptions are known for encryption, commitments and zero-knowledge [17, 5, 29, 16, 20, 14, 1, 15, 27, 28, 2]. Non-malleability in the case of other primitives, notably for one-way functions and for hash functions,[4] has only recently surfaced as a crucial property in several works [7, 8, 11, 19], which we discuss below.

For instance, plenty of cryptographic schemes are only proved secure in the random oracle (RO) model [4], where one assumes that a hash function behaves as a truly random function to which every party has access to. It is well-known that such proofs do not strictly guarantee security for instantiations with hash functions whose only design principles are based on one-wayness and/or collision-resistance, because random functions posses multiple properties the proofs may rely on. Hiding all partial information about pre-images, i.e. perfect one-wayness, is one of these properties, and has been studied in [9, 12]. Non-malleability is another example of such a property.

An illustrative example is the encryption scheme of Bellare and Rogaway [4], where a ciphertext of message M has the form $(f(r), G(r) \oplus M, H(r, M))$ for a trapdoor permutation f, hash functions G, H and random r. The scheme is known to be IND-CCA secure in the random oracle model. However, an instantiation of H with a malleable function for which given $H(r, M)$ it is possible to compute $H(r, M \oplus M')$, for some fixed M' known to the attacker, renders the scheme insecure: the attacker can recover M by submitting to the decryption oracle the valid ciphertext $(f(r), G(r) \oplus M \oplus M', H(r, M \oplus M'))$.

It was shown in [7] that a similar attack can be carried out against the popular OAEP encryption scheme whenever the instantiation of the underlying hash function is malleable. A subsequent work [8] showed that some form of non-malleability permits positive results about security of an alleviated version of the OAEP scheme in the standard model. However, it remains unclear if the approach to non-malleability in [8] expands beyond the OAEP example, and the work left open the construction of non-malleable primitives.

Another motivating example is the abstraction used to model hash functions in symbolic (Dolev-Yao) security analysis. In this setting it is *axiomatized* that an adversary can compute some hash only when it knows the underlying value. Clearly, malleable hash functions do not satisfy this axiom. Therefore, non-malleability for hash functions is necessary in order to ensure that symbolic analysis is (in general) sound with respect to the standard cryptographic model. Otherwise, real attacks that use malleability can not be captured/discovered in the more abstract symbolic model.

In a different vein, and from a more conceptual perspective, higher-level protocols could potentially benefit from non-malleable hash functions as a building block. A recent concrete example is the recommended use of such non-malleable hash functions in a human-computer interaction protocol for protecting local

[4] In the sequel we aggregate both one-way functions and hash functions under the term hash functions for simplicity.

storage [11]. There, access should be linked to the ability to answer human-solvable puzzles (similar to CAPTCHAs), but it should be infeasible for a machine to maul puzzles and redirect them under a different domain to other human beings.

We will also discuss a construction of a cryptographic puzzle from [25] designed to prevent DoS attacks, and show that malleability of the underlying hash function leads to insecure constructions.

Hence, non-malleability is a useful design principle that designers of new hash functions should keep in mind. At this point, however, it is not even clear what the exact requirements from a theoretical viewpoint are. Therefore, a first necessary step is to find a suitable definition which is (a) achievable, and (b) applicable. The next step would be to design practical hash functions and compression functions which are non-malleable, or which at least satisfy some weaker variant of non-malleability.

Contributions. In this paper we initiate the study of non-malleable hash functions. We start with the design of an appropriate security definition. Our definition uses the standard simulation paradigm, also employed in defining non-malleability for encryption and commitment schemes. It turns out however that a careless adjustment of definitions for other primitives yield definitions for non-malleable hash functions that cannot be realized. We therefore motivate and provide a meaningful variation of the definition which ensure that the notion is achievable and may be useful in applications.

Testifying to the difference to other cryptographic primitives, we note that for non-malleable encryption the original simulation-based definition of [17] was later shown to be equivalent to an indistinguishability-based definition [5]. For our case here, finding an equivalent indistinguishability-based definition for non-malleable hash functions appears to be far from trivial, and we leave the question as an interesting open problem.

We then show that our definition can be met. Our construction of a non-malleable hash function employs a perfectly one-way hash function (POWHF) [9, 12], i.e., a probabilistic hash function which hides all information about its pre-image. Notice that this form of secrecy in itself does not ensure non-malleability, so we make the function non-malleable by appending a simulation-sound non-interactive zero-knowledge proof of knowledge (NIZKPoK) [29, 14] of the hashed value.[5] Both primitives exist, for example, if trapdoor permutations exist.[6]

The construction we provide is probabilistic and does not achieve the desired level of efficiency for practical applications. We emphasize that our construction should be regarded as a feasibility result that shows that, in principle, non-

[5] Analogously to Canetti's terminology of perfectly one-way *hash* functions [9] we refer to our construction as a hash function since we require collision resistance, although it does not compress.

[6] We remark that the intuitively appealing approach of using non-malleable encryption or commitment schemes to directly construct non-malleable hashes does not work. One of the reasons is that the former primitives rely on secret randomness, whereas hash values need to be publicly verifiable given the pre-image.

malleable hash functions can be built from standard assumptions. We leave open the problem of finding a practical, deterministic solution. We note that our definition is general enough to allow such constructions.

Next, we investigate necessary cryptographic assumptions for building non-malleable functions. We provide two results. First we show that a non-malleable hash function needs to hide any information about the pre-image. This result justifies the use of POWHFs in our construction. Then we show (in the style of Impagliazzo-Rudich [24]) that black-box constructions of non-malleable *one-way* functions from one-way permutations are in fact impossible even if the collision-resistance requirement is dropped. To be more precise, we follow the approach of Hsiao and Reyzin [23] and show that no black-box security reduction is possible. Notice that our construction circumvents the impossibility result due to the use of a "non-black-box" NIZKPoK.

Finally, we study the applicability of our definition. We show that non-malleability is in fact sufficient for secure partial instantiation of the aforementioned encryption scheme of Bellare and Rogaway [4], i.e., that the scheme remains IND-CCA secure when H is replaced with a non-malleable hash function. Although G is still a random oracle, this partial instantiation helps to better understand the necessary properties of the primitives and also provides a better security heuristic.

We also sketch an application to the framework of cryptographic puzzles [25] as a defense against DoS attacks, where non-malleability surfaces as an important property. The usefulness of the definition has also been shown in [19], using a special case of a preliminary version of our definition to prove that HMAC [3] is a secure message authentication code, assuming that the compression function of the hash function is non-malleable. We expect further applications of non-malleable hash functions in other areas, and some of the techniques used in our proof here may be helpful for these scenarios.

Related Work. Independently of our work, Canetti and Dakdouk [10] and Pandey et al. [26] recently also suggested one-way functions with special properties related to, yet different from non-malleability, and Canetti and Varia [13] investigated non-malleable obfuscation. The work of Canetti and Dakdouk [10] introduces the notion of extractable perfect one-way functions where generating an image also guarantees that one knows a preimage. This should even hold if an adversary sees related images, a setting which somewhat resembles the one that we give for non-malleability. Yet, extractability in [10] is defined by requiring the existence of a knowledge extractor which generates a preimage from the adversary's view, including the other images. In contrast, the common approach to non-malleability (which we also adopt) is to deny the simulator access to the other images, in order to capture the idea that these images should not help. Hence the security definition from [10] is incomparable to ours. Moreover using the notion of [10] to show insecurity of candidate practical hashes seems difficult: arguing about the success of an attacker under their definition involves, in particular, showing that it is impossible to extract a pre-image when someone produces an image. In contrast, security as defined by our notion is easier to

refute. For example, the hash functions from [7] for which flipping a bit in the pre-image results in flipping a bit in the image are clearly insecure under our definition.

The work by Pandey et al. [26] defines adaptive one-way function families where inversion for an image under some key is still infeasible, even if one is allowed to obtain preimages under different keys. This notion is also related to non-malleability and turns out to be useful to design non-malleable protocols like commitments and zero-knowledge proofs. Unfortunately, this strong notion is not known to be realizable.

It is noteworthy that, analogously to our work here, both papers choose the Bellare-Rogaway encryption function as an important test case, and succeed in instantiating the second random oracle of the scheme. Together with the notion that we develop in this paper, these give three different alternatives for the requirements needed for this instantiation. Those works also show that the first random oracle could be instantiated in the standard model with a function which in addition to the notions they define is also pseudorandom. Unfortunately, no construction from standard assumptions that meets either one of the two resulting notions is known. In contrast, our single-oracle instantiation through a non-malleable hash function is possible under standard assumptions.

The work by Canetti and Varia [13] independently considers the notion of verifiable non-malleable obfuscation where an adversary, given an obfuscated circuit, tries to produce an (obfuscated) circuit which is functionally related. The adversary's success is measured against the success of a simulator given only an oracle implementing the original circuit functionality. Their notion of verifiable non-malleable obfuscators comes closest to our notion of non-malleable hash functions, and their construction for achieving a weaker notion of verifiable non-malleable obfuscation resembles our feasibility construction closely.

The two notions are, nonetheless, different in spirit. For obfuscators the adversary's task is to find something *functionally* related, whereas for non-malleable hash functions the adversary's task is to find a hash of a related pre-image, thus capturing relations about *specific* values like relations among the bits. There are further technical differences like the fact that the (achievable) notion of weakly verifiable non-malleable obfuscators does not support auxiliary information —as required for our encryption case, for example— making the two notions incomparable. More details are given in Section 3.

2 Preliminaries

Definition 1 (Hash Functions). *A hash function* $\mathcal{H} = (\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$ *consists of PPTAs for key generation, evaluation and verification, where*

- *PPTA* HK *for security parameter* 1^k *outputs a key* K *(which contains* 1^k *and implicitly defines a domain* D_K*),*
- *PPTA* H *for inputs* K *and* $x \in D_K$ *returns a value* $y \in \{0,1\}^*$*,*
- *PTA* HVf *on inputs* K, x, y *returns a decision bit.*

It is required that for any $K \xleftarrow{\$} \mathsf{HK}(1^k)$*, any* $x \in D_K$*, any* $y \xleftarrow{\$} \mathsf{H}(K, x)$*, algorithm* $\mathsf{HVf}(K, x, y)$ *outputs* 1*.*

Note that we consider a very general syntax, comprising the "classical" notions of one-way functions (with a public key) and of collision-resistant hash functions which compress the input to a shorter digest (see [22] for definitions). In our case the evaluation algorithm H may be probabilistic, as long the correctness of hash values is verifiable given the pre-image only (via HVf). Also, we do not demand the length of the output of the hash function to be smaller than that of the input. However, while we capture a large class of primitives, the generalized syntax may not preserve all properties of the special cases, e.g., if the evaluation algorithm is probabilistic, two independent parties hashing the same input will not necessarily get the same value.

We now recall the definitions of one-wayness and collision resistance. For one-wayness the definition that we give is more general than the standard one in that it considers specific input distributions $\mathcal{X}$ for the function, and also accounts for the possibility that the adversary may have some partial information about the pre-image (modeled through a probabilistic function hint):

Definition 2 (One-wayness and Collision-resistance). *A hash function* $\mathcal{H} = (\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$ *is called*

- one-way *(wrt* $\mathcal{X}$ *and* hint*) if for any PPTA* $\mathcal{A}$ *the probability that for* $K \xleftarrow{\$} \mathsf{HK}(1^k)$*,* $x \xleftarrow{\$} \mathcal{X}(1^k)$*,* $h_x \xleftarrow{\$} \mathsf{hint}(K, x)$*,* $y \xleftarrow{\$} \mathsf{H}(K, x)$ *and* $x^* \xleftarrow{\$} \mathcal{A}(K, y, h_x)$ *we have* $\mathsf{HVf}(K, x^*, y) = 1$*, is negligible.*
- collision-resistant *if for any PPTA* $\mathcal{A}$ *the probability for* $K \xleftarrow{\$} \mathsf{HK}(1^k)$*,* $(x, x', y) \xleftarrow{\$} \mathcal{A}(K)$ *that* $x \neq x'$ *but* $\mathsf{HVf}(K, x, y) = 1$ *and* $\mathsf{HVf}(K, x', y) = 1$*, is negligible.*

3 Non-Malleability of Hash and One-Way Functions

Our definition for hash functions follows the classical (simulation-based) approach for defining non-malleability [17]. Informally, our definition requires that for any adversary which, on input a hash value y, finds another value y^* such that the pre-images are related, there exists a simulator which does just as well *without ever seeing* y*.*

In the adversary's attack we consider a three-stage process. The adversary first selects a distribution $\mathcal{X}$ from which a secret input x is then sampled (and passes on some state information). In the second stage the algorithm sees a hash value y of this input x, and the adversary's goal is to create another hash value y^* (usually different from y). In the third stage the adversary is given x and now has to output a pre-image x^* to y^* which is "related" to x (we make the definition stronger by giving the challenge pre-image to the adversary). The simulator may also pick a distribution $\mathcal{X}$ according to which x is sampled, but then it needs to specify x^* directly from the key of the hash function only.

In the second stage the adversary (and consequently the simulator) also gets as input a "hint" h_x about the original pre-image x, to represent some a-priori information potentially gathered from other executions of other protocols in which x is used. In fact, such side information is often crucial for the deployment in applications, e.g., for the encryption example in Section 6. As in the case of non-malleable commitments and encryption, related pre-images are defined via a relation $R(x, x^*)$. This relation may also depend on the distribution $\mathcal{X}$ to catch significantly diverging choices of the adversary and the simulator and to possibly restrict the choices for $\mathcal{X}$, say, to require a certain min-entropy. However, unlike for other primitives, we do not measure the success of the adversary and the simulator for arbitrary relations R between x and x^*, but instead restrict the relations to a class $\mathcal{R}$ of admissible relations. We discuss this and other subtleties after the definition:

Definition 3 (NM-Hash). *A hash function $\mathcal{H} = (\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$ is called non-malleable (with respect to probabilistic function hint and relation class $\mathcal{R}$)*[7] *if for any PPTA $\mathcal{A} = (\mathcal{A}_d, \mathcal{A}_y, \mathcal{A}_x)$ there exists a PPTA $\mathcal{S} = (\mathcal{S}_d, \mathcal{S}_x)$ such that for every relation $R \in \mathcal{R}$ the difference*

$$\Pr\left[\mathbf{Exp}^{nmh\text{-}1}_{\mathcal{H},\mathcal{A}}(k) = 1\right] - \Pr\left[\mathbf{Exp}^{nmh\text{-}0}_{\mathcal{H},\mathcal{S}}(k) = 1\right] \quad \text{is negligible, where :}$$

Experiment $\mathbf{Exp}^{nmh\text{-}1}_{\mathcal{H},\mathcal{A}}(k)$	***Experiment*** $\mathbf{Exp}^{nmh\text{-}0}_{\mathcal{H},\mathcal{S}}(k)$
$K \xleftarrow{\$} \mathsf{HK}(1^k)$	$K \xleftarrow{\$} \mathsf{HK}(1^k)$
$(\mathcal{X}, st_d) \xleftarrow{\$} \mathcal{A}_d(K)$ *// for state st_d*	$(\mathcal{X}, st_d) \xleftarrow{\$} \mathcal{S}_d(K)$
$x \xleftarrow{\$} \mathcal{X}(1^k)$, $h_x \xleftarrow{\$} \mathsf{hint}(K, x)$	$x \xleftarrow{\$} \mathcal{X}(1^k)$, $h_x \xleftarrow{\$} \mathsf{hint}(K, x)$
$y \xleftarrow{\$} \mathsf{H}(K, x)$	
$(y^*, st_y) \xleftarrow{\$} \mathcal{A}_y(y, h_x, st_d)$	
$x^* \xleftarrow{\$} \mathcal{A}_x(x, st_y)$	$x^* \xleftarrow{\$} \mathcal{S}_x(h_x, st_d)$
Return 1 iff	*Return 1 iff*
$R(\mathcal{X}, x, x^*)$	$R(\mathcal{X}, x, x^*)$
$\wedge\, (x, y) \neq (x^*, y^*)$	
$\wedge\, \mathsf{HVf}(K, x^*, y^*) = 1$	

Remark 1. Our definition is parameterized by a class of relations $\mathcal{R}$. This is because for some relations the definition is simply not achievable, as in the case when the relation involves the hash of x instead of x itself. More specifically, consider the relation $R(x, x^*)$ which parses x^* as K, y and outputs $\mathsf{HVf}(K, x, y)$. Then, an adversary on input y, h_x, st_d may output $y^* \xleftarrow{\$} \mathsf{H}(K, (K, y))$ and then, given x, returns $x^* = (K, y)$. This adversary succeeds in experiment $\mathbf{Exp}^{\mathrm{nmh\text{-}1}}_{\mathcal{H},\mathcal{A}}(k)$ with probability 1. In contrast, any simulator is likely to fail, as long as the hash function does not have "weak" keys, i.e., keys for which the distribution of

[7] Throughout the paper all hint functions and relations are assumed to be efficient. We furthermore assume that the security parameter is given in unary to all algorithms as additional input (if not mentioned explicitly).

generated images is ttrivial (such that the simulator can guess y with sufficiently high probability).

We resolve this problem by requiring the definition to hold for a subset $\mathcal{R}$ of all relations. It is of course desirable to seek secure constructions with respect to very broad classes of relations (cf. our construction in Section 4) which are more handy for general deployment. At the same time, certain scenarios may only require non-malleability with respect to a small set of relations (cf. the application example discussed in Section 6). Our definition is general and permits easy tuning for the needs of a particular application or a class of applications.

Remark 2. For virtually all "interesting" functions $\mathcal{H}$ and relation classes $\mathcal{R}$ the definition is achievable only for adversaries and simulators that output descriptions of well-spread distributions $\mathcal{X}$ (i.e., with super-logarithmic min-entropy). For the construction in next section we also require hint to be a so-called uninvertible function [9] (for which finding the exact pre-image is infeasible). Note that uninvertibility is a weaker requirement than one-wayness, as it holds for example for constant functions. We prefer to keep the definition as general as possible, so we do not explicitly impose such restrictions on the adversary, simulator, and hint.

Remark 3. In our definition we demand that the simulator outputs x^* given K and h_x only. A weaker condition would be to have a simulator $\mathcal{S}_y(h_x, \mathrm{st}_d)$ first output y^*, like the adversary $\mathcal{A}_y$, and then $x^* \leftarrow \mathcal{S}_x(x, \mathrm{st}_y)$, before checking that $R(\mathcal{X}, x, x^*)$ and that $\mathsf{HVf}(K, x^*, y^*) = 1$. Since in this case the simulator in the second stage is also given x we call this a *weak simulator* and hash functions achieving this notion *weakly non-malleable*. This distinction resembles the notions of non-malleable commitments with respect to commitment and with respect to opening [16, 20]. Depending on the application scenario of non-malleable hash functions the stronger or weaker version might be required. As an example, the result about the Bellare-Rogaway encryption scheme uses the stronger definition above, and our construction in the next section achieves this stronger notion, which obviously implies the weaker one.

Remark 4. Similarly to the previous variation one can let the adversary only output a hash value y^*, and omit the step where it later also has to give x^*. The simulator's task, too, is then to only output a hash value. Then one defines meaningful relations through existential quantifications ("... if there exists a pre-image x^* such that $R(x, x^*)$ holds"). This is essentially the approach taken by Canetti and Varia [13] for (weakly) verifiable non-malleable obfuscators.

On the one hand the "hash-only" approach above facilitates the adversary's task if it does not need to know a specific pre-image. On the other hand, it also simplifies the simulator's task. As an example the adversary in our definition may decide upon a *specific* x^* satisfying the relation, after seeing x. Security against such an attack cannot be captured by the above notion of relaxed simulators, whereas the simulator in our defintion also needs to find an appropriate x^*. This particular example demonstrates that our approach and the definition for (weakly) verifiable non-malleable obfuscators in [13] are incomparable. Further

differences between the notions are the lack of auxiliary information and the dependency of the simulator on the relation in the definition of Canetti and Varia [13]. In addition, the feasibility results presented later in our paper and the solutions in [13] are for incomparable classes of relations.

REMARK 5. Note that we only demand that $(x, y) \neq (x^*, y^*)$ for the adversary's choice (instead of demanding $x \neq x^*$ or $y \neq y^*$ instead), yielding a stronger definition, especially when the randomized hash function has multiple images for some input. Again, the particular need depends on the application and our solution meets this stronger requirement.

REMARK 6. In the case of non-malleable encryption the original simulation-based definition of [17] was later shown to be equivalent to an indistinguishability-based definition [5]. The superficial similarity between our definition of non-malleable hash functions and the one of non-malleable encryption suggests that this may be possible here as well. Surprisingly, straightforward attempts to define non-malleability of hash functions through indistinguishability do not seem to yield an equivalent definition. We discuss this issue in the full version [6] in more detail (because of lack of space), and leave it as an interesting open problem to find a suitable indistinguishability-based definition for non-malleable hash functions.

REMARK 7. The usual security notions for hash functions include one-wayness and collision-resistance. However, neither property is known to follow from Definition 3. Consider a *constant* function H which is clearly not one-way nor collision-resistant. But the function is weakly non-malleable as a simulator can simulate $\mathcal{A}$ in a black-box way by handing the adversary the constant value. We keep these rather orthogonal security properties separate, as some applications may require one but not the others.

REMARK 8. Some applications (like the HMAC example in [19]) require a multi-valued version of the definition in which the adversary can adaptively generate several distributions and receive the images (with side information) before deciding upon y^*. One can easily extend our definition accordingly, letting $\mathcal{A}_d$ loop several times, in each round i generating a distribution $\mathcal{X}_i$ and receiving y_i and h_{x_i} at the beginning of the next round and before outputting an image y^*. In general, it is possible to extend our construction to this case using stronger, adaptive versions of POWHFs and NIZKPoKs. See Remark 1 after Theorem 1.

4 Constructing Non-Malleable Hash Functions

In this section we give feasibility results via constructions for non-malleable hash functions. The main ingredient of our constructions is a perfectly one-way hash function (POWHF) [9, 12], which hides all information about the pre-image but which may still be malleable [7]. To ensure non-malleability we tag the hash value with a simulation-sound non-interactive zero-knowledge proof of knowledge of the pre-image. We first recall the definitions of these two primitives.

For POWHFs we slightly adapt the definition from [9, 12] to our setting. Originally, POWHFs have been defined to have a specific input distribution $\mathcal{X}$

(like the uniform distribution in [12, 18]). Here we let the adversary choose the input distribution adaptively, and merely demand that this distribution $\mathcal{X}$ satisfies a certain efficient predicate $P_{\mathsf{pow}}(\mathcal{X})$; this is analogous to the non-malleability experiment in which the adversary chooses $\mathcal{X}$ and the relation R takes $\mathcal{X}$ as additional input. We call the side information here aux (as opposed to hint for non-malleability) in order to distinguish between the two primitives. In fact, in our construction aux uses hint as a subroutine but generates additional output.

Definition 4 (POWHF). *A hash function $\mathcal{P} = (\mathsf{POWK}, \mathsf{POW}, \mathsf{POWVf})$ is called a perfectly one-way hash function (with respect to predicate P_{pow} and probabilistic function aux) if it is collision resistant, and if for any PPTA $\mathcal{B} = (\mathcal{B}_d, \mathcal{B}_b)$, where $\mathcal{B}_b$ has binary output, the following random variables are computationally indistinguishable:*

$K \xleftarrow{\$} \mathsf{POWK}(1^k)$; $x \xleftarrow{\$} \mathcal{X}(1^k)$	$K \xleftarrow{\$} \mathsf{POWK}(1^k)$
$a_x \xleftarrow{\$} \mathsf{aux}(K, x)$; $y \xleftarrow{\$} \mathsf{POW}(K, x)$	$(\mathcal{X}, st_d) \xleftarrow{\$} \mathcal{B}_d(K)$
$b \xleftarrow{\$} \mathcal{B}_b(y, a_x, st_d)$	$x \xleftarrow{\$} \mathcal{X}(1^k)$, $x' \xleftarrow{\$} \mathcal{X}(1^k)$
return (K, x, b) *if* $P_{pow}(\mathcal{X}) = 1$	$a_x \xleftarrow{\$} \mathsf{aux}(K, x)$; $y' \xleftarrow{\$} \mathsf{POW}(K, x')$
else $\perp$	$b \xleftarrow{\$} \mathcal{B}_b(y', a_x, st_d)$
	return (K, x, b) *if* $P_{pow}(\mathcal{X}) = 1$
	else $\perp$

Remark 1. As pointed out in [9, 12] the definition only makes sense if aux is an uninvertible function of the input (such that finding the pre-image x from a_x is infeasible) and $\mathcal{B}_x$ only outputs descriptions of well-spread distributions (with super-logarithmic min-entropy). Otherwise the notion is impossible to achieve. For generality, we do not restrict $\mathcal{X}$ and aux explicitly here.

Remark 2. Perfectly one-way hash functions (in the sense above) can be constructed from any one-way permutation [12, 18] (for the uniform input distribution), any regular collision-resistant hash function [12] (for any distribution with fixed, super-logarithmic min-entropy), or under the decisional Diffie-Hellman assumption [9] (for the uniform distribution). Usually these general constructions are not known to be secure assuming arbirtrary functions aux, yet for the particular function aux required by the application they can often be adapted accordingly. A concrete example is given in Section 6, in our discussion of the Bellare-Rogaway encryption scheme.

On the choice of the relation class. Recall that the definition of non-malleability is parametrized by a class of relations. As explained earlier in the paper, no non-malleable hash function for an arbitrary class exists (see Remark 1 after Definition 3). In the sequel, we exhibit a class of relations for which we show how to construct non-malleable hash functions, and then present our provably secure construction.

Specifically, we consider the class of relations $\mathcal{R}^{\mathsf{rinfo}}_{\mathrm{pred}}$, parameterized by an optional function rinfo and which consists of all relations of the form $R(x, x^*) =$

$P(x,\ P^*(\mathsf{rinfo}(x), x^*))$, for all efficient predicates P, P^*.[8] The function $\mathsf{rinfo}(x)$ may be empty or consist of a small fraction of bits of x (e.g., up to logarithmically many), and should be interpreted as the information about x that may be used in evaluating the relation R. It is important that rinfo is an univertible function, as otherwise, if one could recover x from $\mathsf{rinfo}(x)$, then $\mathcal{R}_{\mathrm{pred}}^{\mathsf{rinfo}}$ would comprise all efficient relations, $R(x, x^*) = P^*(x, x^*)$, and non-malleability with respect to this class, again, would not be achievable.

As an example consider the empty function rinfo such that $\mathcal{R}_{\mathrm{pred}}$ consists of all relations $R(x, x^*) = P(x, P^*(x^*))$. This class of relations allows to check for instance that individual bits of x and x^* are complement of each other, i.e., if π_j denotes the projection onto the j-th bit then one sets $P^*(x^*) = \pi_j(x^*)$ and lets $P(x, P^*(x^*))$ output 1 if $\pi_j(x) \neq \pi_j(x^*)$. This example has also been used by Boldyreva and Fischlin [7] to show the necessity of non-malleability for OAEP, and to give an example of a perfectly one-way hash function that is malleable in the sense that flipping the first bit of an image produces a hash of the pre-image whose first bit is also flipped.

In the examples above rinfo has been the empty function. Of course, using non-trivial functions rinfo allows for additional relations and enriches the class $\mathcal{R}_{\mathrm{pred}}^{\mathsf{rinfo}}$. Consider for example a hash function H that is malleable in the sense that an adversary, given $H(K, r\|m)$ for random $r \in \{0,1\}^k$, can compute $H(K, r\|m')$ for some $m' \neq m$. One way to capture that the two pre-images coincide on the first k bits is to set $\mathsf{rinfo}(r||m) = r$ and to set $P^*(r, x^*) = 1$ if and only if r is the prefix of x^*. Since rinfo should be univertible, the function should rather return only a fraction of r, though. Similarly, one can see that the class $\mathcal{R}_{\mathrm{pred}}^{\mathsf{rinfo}}$ "captures" relations like $R(x, x^*) = 1$ iff $x \oplus x^* = \delta$ for some constant δ, and many other useful relations.

Finally, we note that each relation from the class also checks that the chosen input distribution $\mathcal{X}$ "complies" with the eligible distributions from the underlying POWHF. That is, each relation also checks that the predicate $P_{\mathsf{pow}}(\mathcal{X})$ of the POWHF is satisfied. The full relation $R(\mathcal{X}, x, x^*)$ then evaluates to 1 iff $P(x, P^*(\mathsf{rinfo}(x), x^*)) = 1$ and $P_{\mathsf{pow}}(\mathcal{X}) = 1$. More formally, for any predicate P_{pow} and uninvertible function rinfo we define the class of relations:

$$\mathcal{R}_{\mathrm{pred}}^{\mathsf{rinfo},P_{\mathsf{pow}}} = \left\{ R : \begin{array}{l} \text{there exist efficient (probabilistic) predicates } P, P^* \\ \text{such that } R(\mathcal{X}, x, x^*) = P(x, P^*(\mathsf{rinfo}(x), x^*)) \wedge P_{\mathsf{pow}}(\mathcal{X}) \end{array} \right\}.$$

Our construction also uses a simulation-sound zero-knowledge proof of knowledge $\Pi = (\mathsf{CRS}, \mathsf{P}, \mathsf{V})$ for the NP-relation R_{pow} defined by:

$$R_{\mathsf{pow}} = \{(K_{\mathsf{pow}}||y_{\mathsf{pow}}, x||r) : \mathsf{POW}(K_{\mathsf{pow}}, x; r) = y_{\mathsf{pow}}\}.$$

which essentially says that one "knows" a pre-image of a hash value. Simulation-sound NIZK proofs of knowledge for such relations can be derived from trapdoor permutations [29, 14]. We recall the definition of the former in the full version.

THE CONSTRUCTION AND ITS SECURITY. The following theorem captures the security of our construction.

[8] Where we neglect the distribution $\mathcal{X}$ as part of the relation's input for the moment.

Theorem 1. *Let* $\mathcal{P} = (\mathsf{POWK}, \mathsf{POW}, \mathsf{POWVf})$ *be a perfectly one-way hash function with respect to* P_{pow} *and* aux*, where* $\mathsf{aux} = (\mathsf{hint}, \mathsf{rinfo})$ *for probabilistic functions* hint *and* rinfo*. Let* $\Pi = (\mathsf{CRS}, \mathsf{P}, \mathsf{V})$ *be a simulation-sound non-interactive zero-knowledge proof of knowledge for relation* R_{pow}*. Then the following hash function* $\mathcal{H} = (\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$ *is non-malleable with respect to* hint *and* $\mathcal{R}_{pred}^{\mathsf{rinfo}, P_{\mathsf{pow}}}$*:*

- *PPTA* HK *on input* 1^k *samples* $K_{\mathsf{pow}} \xleftarrow{\$} \mathsf{POWK}(1^k)$ *and* $\mathrm{crs} \xleftarrow{\$} \mathsf{CRS}(1^k)$ *and outputs* $K = (K_{\mathsf{pow}}, \mathrm{crs})$*. The associated domain* D_K *is given by* $D_{K_{\mathsf{pow}}}$*.*
- *PPTA* H *on input* K *and* $x \in D_K$ *computes* $y_{\mathsf{pow}} \leftarrow \mathsf{POW}(K_{\mathsf{pow}}, x; r)$ *for random* $r \xleftarrow{\$} RND_{K_{\mathsf{pow}}}$ *as well as* $\pi \xleftarrow{\$} \mathsf{P}(\mathrm{crs}, K_{\mathsf{pow}} || y_{\mathsf{pow}}, x || r)$*. It outputs* $y = (y_{\mathsf{pow}}, \pi)$*.*
- *PTA* HVf *for inputs* $K = (K_{\mathsf{pow}}, \mathrm{crs})$*,* x *and* $y = (y_{\mathsf{pow}}, \pi)$ *outputs* 1 *if and only if*
  $\mathsf{POWVf}(K_{\mathsf{pow}}, x, y_{\mathsf{pow}}) = 1$ *and* $\mathsf{V}(\mathrm{crs}, K_{\mathsf{pow}} || y_{\mathsf{pow}}, \pi) = 1$*.*

In addition, $\mathcal{H}$ *is collision-resistant.*

Due to space limitations we provide the detailed proof in the full version of the paper [6].

Remark 1. The malleability adversary has access to essentially two different sources of partial information about x: $\mathsf{hint}(x)$ which it receives explicitly as input, and $\mathsf{rinfo}(x)$ which it can use indirectly through the relation R. This motivates the requirement that $\mathcal{P}$ be perfectly one-way with respect to partial information $\mathsf{aux} = (\mathsf{hint}, \mathsf{rinfo})$.

Remark 2. As mentioned after the definition of non-malleable hash functions, some applications (like the one about HMAC [19]) may require a stronger notion in which the adversary can adaptively generate distributions and receives the images, before deciding upon y^*. Our construction above can be extended to this case, assuming that the POWHF obeys a corresponding "adaptiveness" property and that the zero-knowledge proof of knowledge is multiple simulation-sound and multiple zero-knowledge. Such adaptively-secure POWHFs (for uniform distributions) can be built from one-way permutations [18] and suitable zero-knowledge proofs exist, assuming trapdoor permutations [29, 14].

5 On the Complexity of Non-Malleable Functions

In this section we discuss the existential complexity of non-malleable functions. We first indicate, via an oracle separation result, that deriving non-malleable hash and one-way functions via one-way permutations is infeasible. In the full version [6] we also discuss the relation between non-malleability and one-wayness.

5.1 On the Impossibility of Black-Box Reductions

We first show that, under reasonable conditions, there is no black-box reduction from non-malleable hash functions (which might not even be collision-resistant

but rather one-way only) to one-way permutations. For space reasons most of the proofs have been moved to the full version of the paper [6].

Black-Box Reductions. In their seminal paper Impagliazzo and Rudich [24] have shown that some cryptographic primitives cannot be derived from other primitives, at least if the starting primitive is treated as a black box. Instead of separating primitives as in [24] here we follow the more accessible approach of Hsiao and Reyzin [23], giving a relaxed separation result with respect to black-box security reductions. We give a formalization of the oracle-based black-box separation approach that we use in the full version.

For our result we assume that the algorithms of the hash function $\mathcal{H}$ are granted oracle access to a random permutation oracle $\mathcal{P}$ (which is one-way, of course). A black-box reduction to $\mathcal{P}$ is now an algorithm which, with oracle access to $\mathcal{P}$ and a putative successful attacker $\mathcal{A}$ on the non-malleability property, inverts $\mathcal{P}$ with noticeable probability. Such an attacker $\mathcal{A}$ may take advantage of another oracle $\mathcal{O}$ (related to $\mathcal{P}$) which allows it to break the non-malleability but does not help to invert the one-way permutation $\mathcal{P}$. Since neither the construction nor the reduction are given access to $\mathcal{O}$, the reduction must be genuinely black-box.

Defining Oracles $\mathcal{P}$ and $\mathcal{O}$. For now we let $\mathcal{P}$ be a random permutation oracle which in particular is a one-way function. Below we show through derandomization techniques that some fixed $\mathcal{P}$ must also work. For our separation we let the side information of the non-malleable hash function include an image of the uniformly distributed input x under $\mathcal{P}$. More precisely, consider the function $\mathsf{hint}^{\mathcal{P}}_{\mathsf{sep}}$ which on input $(1^k, K, x)$ for random x computes $h_x = \mathcal{P}(0^k||x||\langle\mathsf{HVf}\rangle||K)$ for the description $\langle\mathsf{HVf}\rangle$ of the verification algorithm and finally outputs h_x.[9]

We next construct the oracle $\mathcal{O}$ that helps to break non-malleability. The idea is that using $\mathcal{O}$ it is possible to extract from the image y and "hint" h_x (described above) the pre-image x of y. Since the adversary gets y as input, but the simulator does not, the oracle is only helpful to the adversary. Note that breaking non-malleability means that no simulator of comparable complexity is able to approximate the success probability of $\mathcal{A}^{\mathcal{P},\mathcal{O}}$ closely. To ensure that the simulator has the equal power as $\mathcal{A}^{\mathcal{P},\mathcal{O}}$ we grant the simulator $\mathcal{S}^{\mathcal{P},\mathcal{O}}$ therefore access to both oracles $\mathcal{P}, \mathcal{O}$.

Construction 1. *Let oracle $\mathcal{O}$ take as input a parameter 1^k, an image y and a "hint" h_x. The oracle first finds the pre-image $z||x||\langle\mathsf{HVf}\rangle||K$ of h_x under $\mathcal{P}$ and verifies that $z = 0^k$; if not it immediately returns $\perp$. Else it checks that $\mathsf{HVf}^{\mathcal{P}}(K, x, y) = 1$ and returns x if so (and outputs $\perp$ otherwise).*

[9] We note that the side information h_x does not reveal any essential information about x in the sense that one can show that, for any non-malleable hash function for the uniform input distribution and no side information at all, the hash function remains non-malleable with respect to h_x relative to the random permutation $\mathcal{P}$ (but not relative to $\mathcal{O}$, of course). Also observe that the common strategy of using black-box simulators usually works for any side information, and in particular for the one here.

We show that $\mathcal{O}$ does not help to invert $\mathcal{P}$, thus showing that relative to the oracles there still exists one-way permutations:

Proposition 1. *For any efficient algorithm $\mathcal{B}^{?,?}$, the probability that $\mathcal{B}^{\mathcal{P},\mathcal{O}}$ breaks the one-wayness of $\mathcal{P}$ is negligible.*

In light of this lemma we conclude that there exists a particular $\mathcal{P}$ that is hard to invert for all PPT adversaries with oracles $\mathcal{P}, \mathcal{O}$. The argument is the same as in [23]. For a fixed PPT adversary $\mathcal{B}$, we define the sequence of events (indexed by k) where $\mathcal{B}$ inverts strings of length k with some good probability; for a suitable choice of parameters, the sum of the probabilities (over $\mathcal{P}$) of these events converges and by the first Borel-Cantelli lemma only finitely many of these events may occur, almost surely. Then taking the countable intersection over all PPT $\mathcal{B}$, we get that there is at least one $\mathcal{P}$ with the desired property.

Separation. We require some mild, technical conditions for our non-malleable hash function and the relation. Namely, we assume that

- the hash function is *non-trivial* meaning that it is infeasible to predict an image for uniformly distributed input over $\{0,1\}^k$ (thus ruling out trivial examples like constant hash functions), and
- the relation class $\mathcal{R}$ contains the relation R_{sep} which on input $(\mathcal{X}, x, x^*)$ checks that $\mathcal{X}$ is the uniform distribution on $\{0,1\}^k$, and that $\mathsf{parity}(x) = \bigoplus x_i = \mathsf{parity}(x^*) = \bigoplus x_i^*$. Note that $R_{\mathsf{sep}} \in \mathcal{R}_{\mathrm{pred}}$ for our predicate-based relations, even for the empty function rinfo, and can thus be achieved in principle.

Theorem 2. *Let $\mathcal{H}^{\mathcal{P}} = (\mathsf{HK}^{\mathcal{P}}, \mathsf{H}^{\mathcal{P}}, \mathsf{HVf}^{\mathcal{P}})$ be a non-trivial non-malleable hash function with respect to $\mathsf{hint}^{\mathcal{P}}_{sep}$ and $\mathcal{R} \ni R_{sep}$. Then there exists an adversary $\mathcal{A}^{\mathcal{P},\mathcal{O}}$ that breaks non-malleability of $\mathcal{H}^{\mathcal{P}}$ (for any simulator $\mathcal{S}^{\mathcal{P},\mathcal{O}}$).*

Corollary 1. *There exists no black-box reduction from non-trivial non-malleable functions (with respect to $\mathsf{hint}^{\mathcal{P}}_{sep}$ and $\mathcal{R} \ni R_{sep}$) to one-way permutations.*

At first glance it seems as if our result would transfer (after some minor modifications) to other non-malleable primitives like commitments. This is not the case. The oracle $\mathcal{O}$ in our construction relies on the ability to check whether a pre-image x matches an image y (public verifiability of hash functions), while other primitives such as encryption $\mathcal{E}(m;r)$ and commitments $\mathrm{Com}(m;r)$ use hidden randomness (which is not part of the input of function hint).

Relating Non-Malleability and Perfect One-Wayness. In the full version we show that non-malleability implies a variant of perfect-one-wayness.

6 Applications

In this section we study the usefulness of our notion for cryptographic applications. As an example we show that when one of the two random oracles in the

aforementioned encryption scheme proposed by Bellare and Rogaway in [4] is instantiated with a non-malleable hash function, the scheme remains IND-CCA secure. In addition, we argue that non-malleability is useful in preventing off-line computation attacks against a certain class of cryptographic puzzles.

INSTANTIATING RANDOM ORACLES. We start with recalling the scheme. Let $\mathcal{F}$ be a familiy of trapdoor permutations and G, H be random oracles. The message space of the scheme $\mathrm{BR}^{G,H}[\mathcal{F}] = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is the range of G. The key generation algorithm $\mathcal{K}$ outputs a random $\mathcal{F}$-instance f and its inverse f^{-1} as the public and secret key, respectively. The encryption algorithm $\mathcal{E}$ on inputs f and m picks random r in the domain of f (we assume that $r \in \{0,1\}^k$) and outputs $(\mathsf{f}(r), G(r) \oplus m, H(r||m))$. The decryption algorithm on inputs f^{-1} and (y, g, h) first computes $r \leftarrow \mathsf{f}^{-1}(y)$, then $m \leftarrow g \oplus G(r)$, and outputs m iff $H(r\|m) = h$. The scheme $\mathrm{BR}^{G,H}[\mathcal{F}]$ is proven to be IND-CCA secure in the random oracle model assuming that $\mathcal{F}$ is one-way [4].

Here we study the possibility of realizing the random oracle $\mathcal{H}$ with an actual hash function family $\mathcal{H} = (\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$, a so-called *partial H-instantiation* of the scheme. More precisely, we modify the scheme so that the public key and secret key also contain a key $K \xleftarrow{\$} \mathsf{HK}(1^k)$ specifying a function. Then $\mathcal{E}$ computes $\mathsf{H}(K, r\|m)$ instead of $H(r\|m)$, and $\mathcal{D}$ computes $\mathsf{HVf}(K, r\|m, h)$ instead of checking that $H(r\|m) = h$. We refer to this scheme as $\mathrm{BR}^{G,\mathcal{H}}[\mathcal{F}]$. The following shows that functions that meet our notion of non-malleability are sufficient for a secure partial H-instantiation.

Before stating the sufficient conditions for security to hold, we fix some notation. Below we let the function $\mathsf{rinfo}_{\mathsf{BR}}(x) = \mathrm{msb}_{k/2}(x)$ output the $k/2$ most significant bits of its input. The class of relations we require here for non-malleability is only a subset of the achievable class discussed in Section 4. Namely, we only require a relation of the form $R_{\mathsf{BR}}(\mathcal{X}, x, x^*) = P^*(\mathsf{rinfo}_{\mathsf{BR}}(x), x^*) \wedge P_{\mathsf{pow}}(\mathcal{X})$, where P_{pow} is the predicate that checks that $\mathcal{X}$ is the canonical representation of the uniform distribution on the first k bits, and P^* is the predicate that simply verifies that $\mathrm{msb}_{k/2}(x^*) = \mathsf{rinfo}_{\mathsf{BR}}(x)$. We choose this specific predicate R_{BR} so that it can check if $x = x^*$, while erring with only negligible probability, but still admit the construction of non-malleable hash functions.

Below we will require that the trapdoor permutation family is $\mathrm{msb}_{k/2}$-*partial one-way*, meaning that it is hard to compute the $k/2$ most significant bits of the random input r given a random instance f and $\mathsf{f}(r)$ (cf. [21] for the formal definition). This is a rather mild assumption to impose on $\mathcal{F}$. For example, RSA was shown to be partial one-way under the RSA assumption in [21]. A general approach to construct such a partial one-way family $\mathcal{F}$ is to define $\mathsf{f}(r) = g(\mathrm{msb}_{k/2}(r))\|g(\mathrm{lsb}_{k/2}(r))$ for a trapdoor permutation g.[10]

[10] In fact, this construction also has the useful property that $\mathsf{f}(r)$ is still hard to invert, even if given $\mathrm{msb}_{k/2}(r)$. Thus this trapdoor permutation is suitable for constructing POWHFs secure with respect to side information $(\mathrm{msb}_{k/2}(r), \mathsf{f}(r))$ and therefore, via our construction, non-malleable hash functions for side information $\mathsf{hint}_{\mathsf{BR}}(r) = \mathsf{f}(r)$ and the relation R_{BR}. In other words, non-malleable hash functions for $\mathsf{hint}_{\mathsf{BR}}$ and R_{BR} exist under common cryptographic assumptions.

We need one more technical detail before stating the theorem. We start with some hash function family $\mathcal{H} = (\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$ and trapdoor permutation family $\mathcal{F}$. We write $\mathcal{H} = (\mathsf{HK}_{\mathcal{F}}, \mathsf{H}, \mathsf{HVf})$ for the modified hash function for which key generation outputs a random instance of $\mathcal{F}$ along with the original hash key. Below we write $\mathsf{hint}_{\mathsf{BR}}$ for the function that takes as input a key (K, f) and string x, and outputs $\mathsf{f}(r)$, where r are the first k bits of the input x. We note the IND-*CPA* version of the scheme by Bellare and Rogaway was shown secure in the standard model by Canetti [9], assuming the hash function is a POWHF with respect to a similar hint function.

Theorem 3. *Let $\mathcal{F}$ be an $\mathrm{msb}_{k/2}$-partial one-way trapdoor permutation family and let $\mathcal{H} = (\mathsf{HK}_{\mathcal{F}}, \mathsf{H}, \mathsf{HVf})$ be a collision-resistant hash function which is non-malleable with respect to the function hint_{BR} and to the relation R_{BR}. Assume further that $\mathcal{H}$ is a perfectly one-way hash function with respect to P_{pow} and hint_{BR}. Then $BR^{G,\mathcal{H}}[\mathcal{F}]$ is IND-CCA secure (in the RO model).*

Remark. Although the non-malleability property of the hash implies that no partial information about pre-images is leaked (cf. the full version for a formal statement of this implication), the theorem above requires the hash to be perfectly one-way in the sense of Definition 4, which is a stronger requirement in general. The proof of the theorem is in the full version [6].

Application to cryptographic puzzles. Cryptographic puzzles are a defense mechanism against denial of service attacks (DoS). The idea is that, before spending any resources for the execution of a session between a client and a server, the server requires the client to solve a puzzle. Since solving puzzles requires spending cycles, the use of puzzles prevents a malicious client to engage in a large number of sessions without spending itself a significant amount of resources. One desirable condition is that the server does not store any client-related state.

A simple construction for such puzzles proposed by Juels and Brainard [25] is based on any arbitrary one-way function $h : \{0,1\}^l \to \{0,1\}^l$. First, select at random $x \xleftarrow{\$} \{0,1\}^l$ and compute $y = h(x)$. Then, a puzzle is given by the tuple $(x[1..l-k], y)$ consisting of the first $l-k$ bits of x together with y. To prove it solved the puzzle, the client has to return (x, y). It can be easily seen that the construction above is not entirely satisfactory. In particular, it either fails against replay attacks —where the clients present the same puzzle-solution pair to the server— or the server needs to store all of the x's used to compute the puzzles.

The solution proposed to mitigate the above problem is to compute x as $H(S, t)$, where S is some large bitstring known only to the server, and t is some bitstring that somehow "expires" after a certain amount of time (this can be for example the current system time). The puzzle is then given by $(t, x[1..l-k], y)$, where $y = h(x)$. A solution (or solved puzzle) is (t, x, y) which needs to satisfy the obvious equations, and moreover, t is not an expired bitstring.

In the setting above, non-malleability of H surfaces as an important property. If out of the first two elements $(t, H(S, t))$ of a puzzle solution the adversary can

efficiently construct $(t', H(S, t'))$ for $t' \neq t$, a string which has not yet expired, then the defense sketched above is rendered useless: the adversary can easily construct new puzzles (together with their solutions). Requiring that the function H is non-malleable with respect to the relation $R(s_1, s_2) = 1$ iff $s_1 = (S, t)$ and $s_2 = (S, t')$ for $t \neq t'$ is sufficient to prevent the above attack.

Acknowledgments

We thank all the reviewers for their comments. We also thank Vipul Goyal for stimulating discussions. Alexandra Boldyreva is supported in part by NSF CAREER award 0545659 and NSF Cyber Trust award 0831184. David Cash is supported in part by the aforementioned grants. Marc Fischlin is supported by the Emmy Noether Program Fi 940/2-1 of the German Research Foundation DFG. The work is partially funded by the European Commission through the ICT programme under Contract ICT-2007-216646 ECRYPT II.

References

1. B. Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. FOCS '02, pp. 345–355, IEEE, 2002.
2. B. Barak and M. Prabhakaran and A. Sahai. Concurrent non-malleable zero knowledge. FOCS '05, pp. 563–572, IEEE, 2005.
3. M. Bellare and R. Canetti and H. Krawczyk. Keying hash functions for message authentication. CRYPTO '96, Vol. 1109 of LNCS, pp. 1–15, Springer, 1996.
4. M. Bellare and P. Rogaway, Random oracles are practical: a paradigm for designing efficient protocols, *CCS '93*, pp. 62–73, ACM, 1993.
5. M. Bellare and A. Sahai. Non-malleable encryption: Equivalence between two notions, and an indistinguishability-based characterization. CRYPTO '99, Vol. 1666 of LNCS, pp. 519–536, Springer, 1999.
6. A. Boldyreva, D. Cash, M. Fischlin and B. Warinschi. Foundations of non-malleable hash and one-way functions. Full version of this paper. Cryptology ePrint Archive, Report 2009/065, 2009.
7. A. Boldyreva and M. Fischlin. Analysis of random-oracle instantiation scenarios for OAEP and other practical schemes. CRYPTO '05, Vol. 3621 of LNCS, pp. 412–429, Springer, 2005.
8. A. Boldyreva and M. Fischlin. On the security of OAEP. ASIACRYPT '06, Vol. 4284 of LNCS, pp. 210–225, Springer, 2005.
9. R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. CRYPTO '97, Vol. 1294 of LNCS, pp. 455–469, Springer, 1997.
10. R. Canetti and R. R. Dakdouk. Extractable perfectly one-way functions. ICALP'08, Vol. 5126 of LNCS, pp. 449–460, Springer, 2008.
11. R. Canetti, S. Halevi and M. Steiner. Mitigating dictionary attacks on password-protected local storage. CRYPTO '06, Vol. 4117 of LNCS, pp. 160-179, Springer, 2006.
12. R. Canetti, D. Micciancio and O. Reingold. Perfectly one-way probabilistic hash functions. STOC '98, pp. 131–140, ACM, 1998.
13. R. Canetti, M. Varia. Non-malleable obfuscation. TCC '09, Vol. 5444 of LNCS, pp. 73–90. Springer, 2009.

14. A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. *Robust Non-interactive Zero Knowledge.* CRYPTO '01, Volume 2139 of LNCS, pages 566–598. Springer, 2001.
15. I. Damgård and J. Groth. Non-interactive and reusable non-malleable commitment schemes. STOC '03, pp. 426–437, ACM, 2003.
16. G. Di Crescenzo and Y. Ishai and R. Ostrovsky. Non-interactive and non-malleable commitment. STOC '98, pp. 141–150, ACM, 1998.
17. D. Dolev, C. Dwork and M. Naor. Non-malleable cryptography. SIAM Journal on Computing, Volume 30(2), pp. 391–437, 2000.
18. M. Fischlin. Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications. EUROCRYPT '99, Vol. 1592 of LNCS, pp. 429–444, Springer, 1999.
19. M. Fischlin. Security of NMAC and HMAC based on non-malleability. RSA-CT '08, LNCS, pp. 138–154, Springer, 2008.
20. M. Fischlin and R. Fischlin. Efficient non-malleable commitment schemes. CRYPTO '00, Vol. 1880 of LNCS, pp. 414–432, Springer, 2000.
21. E. Fujisaki, T. Okamoto, D. Pointcheval and J. Stern. RSA-OAEP is secure under the RSA Assumption. CRYPTO '01, Vol. 2139 of LNCS, pp. 260–274, Springer, 2001.
22. O. Goldreich. The foundations of cryptography. (Volume 1), Cambridge University Press, 2004.
23. C.-Y. Hsiao and L. Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins. CRYPTO '04, Vol. 3152 of LNCS, pp. 92-105, Springer, 2004.
24. R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. STOC' 89, pp. 44–61, ACM, 1989.
25. A. Juels and J. Brainard. Client puzzles: A cryptographic countermeasure against connection depletion attacks. NDSS '99, pp. 151–165, 1999.
26. O. Pandey, R. Pass and V. Vaikuntanathan. Adaptive one-way functions and applications. CRYPTO '08, Vol. 5157 of LNCS, pp. 57–74, Springer, 2008.
27. R. Pass and A. Rosen. Concurrent non-malleable commitments. FOCS '05, pp. 563–572, IEEE, 2005.
28. R. Pass and A. Rosen. New and improved constructions of non-malleable cryptographic protocols. STOC '05, pp. 533–542, ACM Press, 2005.
29. A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. FOCS '99, p. 543, IEEE, 1999.