Random Oracles and Auxiliary Input

Dominique Unruh*

Saarland University, Saarbrücken, Germany, unruh@cs.uni-sb.de

Abstract. We introduce a variant of the random oracle model where oracle-dependent auxiliary input is allowed. In this setting, the adversary gets an auxiliary input that can contain information about the random oracle. Using simple examples we show that this model should be preferred over the classical variant where the auxiliary input is independent of the random oracle.
In the presence of oracle-dependent auxiliary input, the most important proof technique in the random oracle model—lazy sampling—does not apply directly. We present a theorem and a variant of the lazy sampling technique that allows one to perform proofs in the new model almost as easily as in the old one.
As an application of our approach and to illustrate how existing proofs can be adapted, we prove that RSA-OAEP is IND-CCA2 secure in the random oracle model with oracle-dependent auxiliary input.
Keywords: Random oracles, auxiliary input, proof techniques, foundations.

1 Introduction

In [3] the following heuristic was advocated as a practical way to design cryptographic protocols:[1] To prove the security of a cryptographic scheme, one first introduces a *random oracle* $\mathcal{O}$, i.e., a randomly chosen function to which all parties including the adversary have access. One then proves the security of the scheme that uses the random oracle and subsequently replaces the random oracle by a suitably chosen function (or family of functions) H. The *random oracle heuristic* now states that if the scheme using $\mathcal{O}$ is secure, the scheme using H is secure as well.

Unfortunately, a counter-example to this heuristic has been given in [6]. It was shown that there exist public key encryption and signature schemes that are secure in the random oracle model but lose their security when instantiated with *any* function or family of functions. Nonetheless, the random oracle heuristic still is an important design guideline for implementing cryptographic schemes.

Furthermore, [15] pointed out that zero-knowledge proofs in the random oracle model can lose their deniability when instantiated with a fixed function. In contrast to the result of [6], this happens even for natural protocols. However, [15] was able to give conditions under which this effect does not occur and gave a protocol that fulfilled these conditions.

Although the heuristic is known not to be sound in general, no practical scheme is known where it fails, and schemes that are proven to be secure using this heuristic tend

* Part of this work was done while the author was at the IAKS, University of Karlsruhe, Germany

[1] However, the basic idea seems to have already appeared earlier.

to be simpler and more efficient than schemes that are shown to be secure in the standard model. As a consequence, schemes used in practise are often based on the random oracle heuristic, e.g., the RSA-OAEP encryption scheme, introduced in [2] and standardised in [17], is one of the most widely used public-key encryption schemes, and its security is based on the random oracle heuristic.

In the light of the results of [6] and [15], and of the practical importance of the random oracle heuristic, it is important to try and learn what the exact limitations of the heuristic are, and, if possible, give criteria to distinguish those protocols in the random oracle model that become insecure when instantiated due to those limitations, and those protocols where we can at least hope—if not prove—that their instantiations are secure. The augmented definition of zero-knowledge by [15] is an example of such a criterion.

In this paper, we uncover another such limitation of the random oracle world. We will see that there are natural schemes secure in the random oracle model that become insecure *with respect to auxiliary input* (or equivalently, with respect to nonuniform adversaries) when instantiated. As [15] did for the deniability, we give augmented definitions for the random oracle model with auxiliary input that allow one to distinguish protocols that fail upon instantiation from those that do not (at least not due to the abovementioned limitation).

Although such a result does not imply the soundness of the random oracle model, it helps to better understand which protocol can reasonably be expected to be secure when instantiated with a fixed function.

We will now investigate the problem of auxiliary input in the random oracle model in more detail. An important concept in cryptology is the *auxiliary input*. The auxiliary input is a string that is given to the adversary at the beginning of the execution of some cryptographic protocol. This string is usually chosen nonuniformly and depends on all protocol inputs. In other words, the auxiliary input models the possibility that the adversary has some additional knowledge concerning the situation at the beginning of the protocol. This additional knowledge may, e.g., represent information acquired in prior protocol runs. It turns out that in many cases the presence of an auxiliary input is an essential concept for proving secure sequential composition. Therefore, most modern cryptographic schemes are designed to be secure even in the presence of an auxiliary input (given that the underlying complexity assumptions hold against nonuniform adversaries).

However, when we try to combine these two concepts, the random oracle model and the auxiliary input, undesirable effects may occur. We will demonstrate this by studying the definitions of two simple security notions: one-wayness and collision-resistance. First, consider the notion of a one-way function. We construct a function $f := \mathcal{O}$ in the random oracle model and ask whether it is one-way. For this, we substitute $\mathcal{O}$ for f in the definition of one-wayness with respect to auxiliary input, and get the following definition: The function f is one-way if for any polynomial-time adversary A and any auxiliary input z, the following probability is negligible in the security parameter k:

$$P\big(x \xleftarrow{\$} \{0,1\}^k,\ x' \leftarrow A^{\mathcal{O}}(1^k, z, \mathcal{O}(x)) :\ \mathcal{O}(x') = \mathcal{O}(x)\big). \tag{1}$$

Here $\mathcal{O}$ is a randomly chosen function (with some given domain and range), and the adversary is given black-box access to $\mathcal{O}$. It is now easy to see that $f := \mathcal{O}$ is indeed secure in the above sense: The adversary can make at most a polynomial number of queries, and each query except $\mathcal{O}(x)$ returns a uniformly random image (exploiting this latter fact is later called the lazy sampling technique). From this fact one can conclude that the adversary must make an exponential number of queries to find a preimage of $\mathcal{O}(x)$, hence f is secure. The presence of the auxiliary input does not have noticeable impact on the proof. The random oracle heuristic now claims that $f := H$ is oneway for a sufficiently unstructured function H, even in the presence of auxiliary input. So far, nothing out of the ordinary has happened.

We now try to use the same approach for another security property: collision-resistance. Again, we set $f := \mathcal{O}$, and then collision-resistance of f means that for any polynomial-time adversary A and any auxiliary input z, the following probability is negligible:

$$P\big((x_1, x_2) \leftarrow A^{\mathcal{O}}(1^k, z) :\ x_1 \neq x_2 \text{ and } \mathcal{O}(x') = \mathcal{O}(x)\big). \tag{2}$$

This can again easily be proven using the lazy sampling technique: the answers to the adversary's queries are independent random values, and finding a collision requires two of these random values to be identical which happens only with negligible probability. Again, the auxiliary input does not help the adversary, since it does not contain any information on where a collision might be. We now use the random oracle heuristic, replace $\mathcal{O}$ by some sufficiently unstructured function H, so that $f = H$, and then claim that f is collision-resistant in the presence of an auxiliary input. But this of course is impossible, since the auxiliary input may simply contain a collision of H, since H is a fixed function.[2]

Hence, the random oracle heuristic should not be applied to collision-resistance. On the other hand, we would like to prove the one-wayness of $f := \mathcal{O}$ in the random oracle model. We hence need a stronger variant of the random oracle heuristic that does not allow one to prove the collision-resistance of f, but still allows one to prove its one-wayness. An inspection of our proof above reveals the mistake we made: In the random oracle model, the auxiliary input was chosen before the random oracle, so it could not contain a collision. After instantiation, the function H was fixed, so the auxiliary input did depend on H and therefore could provide a collision. The random oracle heuristic should hence be recast as follows in the case of auxiliary input: When a scheme is secure in the random oracle model with *oracle-dependent* auxiliary input, it is still secure after replacing the random oracle by a sufficiently unstructured fixed function H, even in the presence of auxiliary input.

It remains to clarify the formal meaning of oracle-dependent auxiliary input. Unfortunately, we cannot simply say: "for randomly chosen $\mathcal{O}$ and every z". At least, the semantics underlying constructions like (1) and (2) get highly nontrivial in this case. Fortunately, there is another possibility. By an oracle function z we mean a function that

[2] If we replace $\mathcal{O}$ by a family of functions, i.e., some parameter i is chosen at the beginning of the protocol, and then a funcion H_i is used, then the problem described here does not occur. Unfortunately, one is not always free to use such a family of functions. On one hand, the index has in some way to be chosen, and we do not want to leave that choice to the corrupted parties. On the other hand, practical applications usually instantiate the random oracle using a fixed function like SHA-1 or SHA-256 [8].

returns a string $z^{\mathcal{O}}$ for each possible value of the random oracle $\mathcal{O}$. So formally, z is simply a function that maps functions to strings. Then a scheme is called secure in the random oracle model with oracle-dependent auxiliary input if for any polynomial-time adversary A and for any oracle function z into strings of polynomial length (that may depend on k, of course), the adversary cannot break the scheme *even when given $z^{\mathcal{O}}$ as auxiliary input*.

As with the traditional random oracle model, the exact form these definitions take depends on the security notion under consideration. For example, the one-wayness and the collision-resistance of $f := \mathcal{O}$ take the following form: for any polynomial-time adversary A and any oracle function z into strings of polynomial-length, we have

$$P\big(x \stackrel{\$}{\leftarrow} \{0,1\}^k,\ x' \leftarrow A^{\mathcal{O}}(1^k, z^{\mathcal{O}}, \mathcal{O}(x)) :\ \mathcal{O}(x') = \mathcal{O}(x)\big) \tag{3}$$

or

$$P\big((x_1, x_2) \leftarrow A^{\mathcal{O}}(1^k, z^{\mathcal{O}}) :\ x_1 \neq x_2 \text{ and } \mathcal{O}(x') = \mathcal{O}(x)\big), \tag{4}$$

respectively. However, we can give a simple guideline on how to transform a security definition in the random oracle model with an oracle-*independent* auxiliary input z into a security definition with oracle-*dependent* auxiliary input. First, one quantifies over oracle functions z instead of strings z. And then one replaces all occurrences of the string z by $z^{\mathcal{O}}$.

It is now easy to see that (4) is not negligible: let $z^{\mathcal{O}}$ encode a collision x_1, x_2, and let the adversary output that collision. Since such a collision always exists (assuming a *length-reducing* f), this breaks the collision-resistance of f in the presence of oracle-dependent auxiliary input, as we would have expected.

On the other hand, we expect (3) to be negligible in the presence of oracle-dependent auxiliary input. However, it is not so easy to see whether there may not be some possibility to encode information about the random oracle in a string of polynomial length that allows one to find a preimage with non-negligible probability. Although one-wayness is one of the weakest conceivable security notions, proving its security with respect to oracle-dependent auxiliary input is quite difficult. (We encourage the reader to try and find an elementary proof for the one-wayness of f.[3]) The reason for this difficulty lies in the fact that it is not possible any more to apply the lazy sampling technique: given some information $z^{\mathcal{O}}$ on the random oracle $\mathcal{O}$, the images under the random oracle are not independently nor uniformly distributed any more. We therefore need new techniques if we want to be able to cope with oracle-dependent auxiliary input and to prove more complex cryptographic schemes secure in this model. Such techniques will be presented in this paper.

On nonuniform and uniform auxiliary input. In this work, we always consider nonuniform auxiliary inputs, that is, the auxiliary input is not required to be the result of an efficient computation. This is the most common modelling of auxiliary input in the cryptographic community. However, it is also possible to consider uniform auxiliary inputs: In this case, the auxiliary input is not an arbitrary sequence of strings, but is instead the output of a *uniform* probabilistic algorithm. The main motivation of the auxiliary input, namely to model information gained from prior executions of cryptographic

[3] We give a proof using the techniques from this paper in Lemma 10.

protocols on the same data, and thus to allow for composability, is preserved by this uniform approach. (See [11] for a detailed analysis.) The main disadvantage of the uniform approach is that definitions and proofs get more complicated due to the presence of another machine. This is why the nonuniform auxiliary input is more commonly used.

Applying the uniform approach to our setting, a uniform oracle-dependent auxiliary input would be the output of a polynomial-time oracle Turing machine Z with access to the random oracle. Since that Turing machine could only make a polynomial number of queries, using the lazy sampling technique would be easy: all positions of the random oracle that have not been queried by Z can be considered random.

However, if we use the random oracle heuristic to motivate the security of a protocol with respect to *uniform* auxiliary input, the result is incompatible with existing theorems and definitions in the *nonuniform* auxiliary input model. So to use the random oracle heuristic together with existing results, we either have to reprove all existing results for the uniform case, or we have to use *nonuniform* oracle-dependent auxiliary input. It is the latter approach we follow in this work.

Instantiating the random oracle with keyed families of functions. Above, we showed that the random oracle is (unsurprisingly) not collision-resistant in the presence of auxiliary input. It follows that we may not instantiate the random oracle with a fixed function if we need collision-resistance. On the other hand, replacing the random-oracle by a keyed family of functions may be secure, since the auxiliary input cannot encode a collision for each function. We do not claim that it is necessary to use oracle-dependent auxiliary input when instantiating with families of functions. Rather, oracle-dependent auxiliary input provides a tool for distinguishing the cases where the use of a single function[4] is sufficient (e.g., in the case where we require only one-wayness) and where a keyed family of functions is necessary (e.g., in the case that we require collision-resistance). Since instantiating with a single function is much simpler (e.g., we do not have to worry about who chooses the key), and is the usual practice in real-world protocols, examining random oracle based protocols with respect to oracle-dependent auxiliary input may give additional insight into when instantiation with single functions is permitted and when we have to use keyed families. Another disadvantage of using a family of functions is that we have to ensure that the key is honestly generated, which may introduce additional difficulties if no trusted party is available for this task.

Designing special protocols. An alternative to the approach in this paper would be to systematically construct or transform a protocol so that it is secure with respect to oracle-dependent auxiliary input (instead of verifying a given protocol). However, here the same arguments as in the previous paragraph apply. First, we might not be interested in a new protocol, but might want to examine the security of an existing protocol (that possibly even has already been implemented). Further, efficiency considerations might prevent the use of more elaborate constructions.

[4] Here, by a single function we mean that the function is not parametrised by a key that has to be known by all parties. However, the function may depend on the security parameter. Otherwise a property like collision-resistance trivially cannot be fulfilled by a single function, even against uniform adversaries. See also [16] in this context.

1.1 Our results

We introduce and motivate the random oracle model with *oracle-dependent auxiliary input* (preceding section). In this model, the auxiliary input given to the adversary may depend on the random oracle.

In order to be able to prove security in the new model, we introduce a new variant of the lazy sampling technique that is applicable even in the presence of oracle-dependent auxiliary input. We show that one can replace the random-oracle $\mathcal{O}$ by a new random oracle $\mathcal{P}$ that is independent of the auxiliary input, except for a *presampling*. That is, a small fraction of the total random oracle $\mathcal{P}$ is fixed (and dependent on the auxiliary input), while all other images are chosen independently and uniformly at random (and in particular are independent of the auxiliary input). In this new setting, lazy sampling is possible again: an oracle query that is not in the presampled set is given a random answer.

This also gives some insight into why some schemes are secure and some fail in the presence of oracle-dependent auxiliary input: Intuitively the protocols that fail are those for which you can have a "reason for a failure" (e.g., a collision) contained in a few entries of the random oracle.

As a technical tool, we also formulate a *security amplification* technique: for many security notions, security with respect to nonuniform polynomial-time adversaries implies security with respect to nonuniform adversaries whose running time is bounded by some suitable superpolynomial function f. This technique is useful in the context of oracle-dependent auxiliary input, since some reduction proofs with presampling tend to introduce superpolynomial adversaries.

As an application of our techniques, we show that RSA-OAEP is IND-CCA2 secure in the random oracle model *with oracle-dependent auxiliary input*. Our proof closely follows the proof of [9] where the security of RSA-OAEP was shown in the classical random oracle model. This allows the reader to better compare the differences in the proof introduced by the oracle-dependent auxiliary input. However, we believe that the result does not only exemplify our techniques but is worthwhile in its own light: it gives the first evidence that RSA-OAEP as used in practical application (i.e., with the random oracle instantiated with a fixed function H), is secure *even in the presence of an auxiliary input.*

1.2 Related work

In [19], the problem of composition of zero-knowledge proofs in the random-oracle model is investigated. It is shown that to guarantee sequential composition, oracle-dependent auxiliary input is necessary. Their definition of oracle-dependent auxiliary input is somewhat weaker than ours in that the machine z generating the auxiliary input is allowed only a polynomial number of queries to the random oracle (it is similar to uniform oracle-dependent auxiliary input in that respect). They give protocols that are secure with respect to that notion. It would be interesting to know whether the techniques developed here allow to show their protocols to be secure even with respect to our stronger notion of oracle-dependent auxiliary input.

In [10], it was shown that a random *permutation* is one-way with respect to oracle-dependent auxiliary input. They showed that the advantage of the adversary is in $2^{-\Omega(k)}$ which is essentially the same bound as we achieve for random *functions* in Section 3. However, their proof is specific to the property of one-wayness and does not generalise to our setting. According to [10], a similar result was shown for random *functions* in [12]. However, their proofs apply only to the one-wayness of the random oracle, while our results imply that many more cryptographic properties of the random oracle are preserved in the presence of oracle-dependent auxiliary input.

In [14,5,4,7], unconditional security proofs in the bounded-storage model were investigated. In this model, one assumes that the adversary is computationally unlimited, but that it may only store a limited amount of data. One assumes that at the beginning of the protocol some large source of randomness (e.g., a random oracle) is available to all parties. The security of the protocol then roughly hinges on the following idea: The honest parties store some (small) random part of the source. Since the adversary does not know which part has been chosen, and since it may not store the whole source, with high probability the honest parties will find some part of the random source they both have information about, but that is unknown to the adversary. To prove the security in this model, it is crucial to show that the adversary cannot compress the source in a manner that contains enough information to break the protocol. This is very similar to the scenario investigated here, since the oracle-dependent auxiliary input can be seen as compressed information on the random oracle. Our results differ from those in the bounded-storage model in two ways: first, our results cover a more general case, since we consider the effect of auxiliary input on arbitrary protocols, while in the bounded-storage model a single protocol is analysed that is specially designed to extract information from the random source that cannot be extracted given only a part of the source. On the other hand, precisely due to the specialised nature of the protocols, the bounds achieved in the bounded-storage model are better than those presented here. In particular, there are protocols in the bounded storage model that are secure given a random source of polynomial size [7], while our results are—at least with the present bounds—only useful if the domain of the random oracle has superpolynomial size (cf. the exact bounds given by Theorem 2). It would be interesting to know whether our techniques can be used in the context of the bounded-storage model, and to what extent the techniques developed in the bounded-storage model can be applied to improve our bounds.

1.3 Further applications

Besides the application described above, namely to be able to use the random-oracle heuristic in the case of auxiliary input, our main result (the lazy sampling technique) may also be useful in other situations.

In [10] it was shown that a random permutation is one-way in the presence of oracle-dependent auxiliary input. This was the main ingredient for several lower bounds on black-box constructions using one-way permutations. Using our techniques, we might find lower bounds on black-box constructions based on other cryptographic primitives: namely, we would show that the random oracle (or a protocol using the random oracle) has a given security property X even in the presence of oracle-dependent auxiliary input.

Then using techniques from [10], lower bounds on black-box constructions based on cryptographic primitives fulfilling X might be derived.

1.4 Organisation

In Section 1 we introduce and motivate the concept of oracle-dependent auxiliary input. In Section 2 we present the main result of this paper: a theorem that allows one to use the lazy sampling technique even in the presence of oracle-dependent auxiliary input. In Section 3 we give a simple example to show how to use the lazy sampling technique. In Section 4 we present the security amplification technique. This technique allows one to use superpolynomial adversaries in reduction proofs, which sometimes is needed when using the lazy sampling technique. In Section 5 we prove that RSA-OAEP is IND-CCA2 secure in the random oracle model with oracle-dependent auxiliary input. Details and proofs left out in this paper are given in the full version [18].

1.5 Notation

For random variables A and B, we denote the Shannon-entropy of A by $H(A)$, and the conditional entropy of A given B by $H(A|B)$. The statistical distance between A and B is denoted $\Delta(A;B)$. The operator $\log$ means the logarithm base 2. The variable k always denotes the security parameter. In asymptotic statements of theorems or definitions, some variables implicitly depend on the security parameter k. These variables are then listed at the end of the theorem/definition. We call a nonnegative function in k negligible, if it lies in $k^{-\omega(1)}$. We call a nonnegative function non-negligible if it is not negligible.

Let $\mathcal{O}$ always denote the random oracle. Let $Domain$ be the domain and $Range$ the range of the random oracle, i.e., $\mathcal{O}$ is a uniformly random function from $Domain \to Range$. In an asymptotic setting, $\mathcal{O}$, $Domain$ and $Range$ implicitly depend on the security parameter k. In this case we always assume $\#Domain$ and $\#Range$ to grow at least exponentially in k.

An *oracle function g into X* is a mapping from $Domain \to Range$ into X. We write the image of some function $\mathcal{O}$ under g as $g^{\mathcal{O}}$.

An *assignment* S is a list $S = (x_1 \to y_1, \dots, x_n \to y_n)$ with $x_i \in Domain$ and $y_i \in Range$ and with $x_i \neq x_j$ for $i \neq j$. The length of S is n. We call y_i the image of x_i under S. We write $x \in S$ if $x_i = x$ for some i. The image $\operatorname{im} S$ is defined as $\operatorname{im} S = \{y_1, \dots, y_n\}$.

2 Lazy sampling with auxiliary input

The main result of this paper is the following theorem which guarantees that we can replace a random oracle with oracle-dependent auxiliary input by a new random oracle that is independent of the auxiliary input with the exception of some fraction of its domain (which is *presampled*). In order to formulate the theorem, we first need to state what exactly we mean by an oracle with presampling:

Definition 1 (Random oracle with presampling). *Let $S = (x_1 \to y_1, \ldots, x_n \to y_n)$ be an assignment. Then the* random oracle $\mathcal{P}$ with presampling S *is defined as follows:*

When queried $x \in \mathit{Domain}$ with $x = x_i$ for some $i \in \{1, \ldots, n\}$, the oracle returns y_i. If x has already been queried, the same answer is given again. Otherwise, a uniformly random element y is chosen from Range and returned.

We can now state the main theorem.

Theorem 2 (Lazy sampling with auxiliary input). *Let $f \geq 1$ and $q \geq 0$ be integers. Let z be an oracle function with finite range Z and $p := \log \#Z$.*

Then there is an oracle function S, such that $S^{\mathcal{O}}$ is an assignment of length at most f, so that the following holds: For any probabilistic oracle Turing machine A that makes at most q queries to the random oracle, it is

$$\Delta\big(A^{\mathcal{O}}(z^{\mathcal{O}});\ A^{\mathcal{P}}(z^{\mathcal{O}})\big) \leq \sqrt{\frac{pq}{2f}}$$

where $\mathcal{P}$ is the random oracle with presampling $S^{\mathcal{O}}$.

Before presenting the actual proof, we give a short sketch that is intended to serve as a guide through the rest of the proof. To ease comparison with the details given later, we provide forward references to the lemmas of the actual proof.

For any i, let J_i be the maximum amount of information that a sequence of i queries to the random oracle $\mathcal{O}$ gives about the auxiliary input $z^{\mathcal{O}}$. Since $|z| = p$, $J_i \leq p$ for all i. Let F_i be the sequence of i queries that achieves this bound, that is, the mutual information between $z^{\mathcal{O}}$ and the oracle's answers to F_i is J_i.

Assume that the queries F_i have already been performed. Let G be a sequence of q queries. Then the answers to the queries F_i and G together contain at most J_{q+i} bits of information about z. Thus the answers to G contain at most $J_{q+i} - J_i$ bits of information about z beyond what is already known from the answers to F_i.

Consider the quantities $J_0, J_q, J_{2q}, \ldots, J_{f+q}$ (assuming that q divides f). Since $J_0 \geq 0$ and $J_{f+q} \leq p$, there must be some $f' \leq f$ such that the $J_{f'+q} - J_{f'} \leq \frac{pq}{f}$. Thus, given the answers to $F := F_{f'}$, any sequence G of q queries reveals at most $\frac{pq}{f}$ bits about the auxiliary input z. In other words, the answers to G are almost independent of z (assuming that $\frac{pq}{f}$ is sufficiently small). Thus, if we fix the oracle $\mathcal{P}$ to match the answers to F, but choose $\mathcal{P}$ independently of z everywhere else, with q queries we cannot distinguish between $\mathcal{P}$ and the original oracle $\mathcal{O}$. This gives Theorem 2 (except for the concrete bound $\sqrt{pq/2f}$).

In reality, however, the queries performed by A are adaptive, i.e., they depend on z and on the answer to prior queries. So we cannot talk about a fixed sequence G of queries made by A. To overcome this problem, we introduce the concept of an adaptive list (Definitions 3 and 4), which is a generalisation of a sequence of queries where the queries are allowed to be adaptive. When considering adaptive lists, it does not make immediate sense to speak about the mutual information between the answers to an adaptive list G and the auxiliary input $z^{\mathcal{O}}$. In Definition 5 we therefore introduce quantities $J(G)$ and $J(G|F)$ denoting the information that the answers to the adaptive list G contains about $z^{\mathcal{O}}$ (given the answers to the adaptive list F in the case of $J(G|F)$). For this quantity, we show that $J(F) \leq p$ (Lemma 7) and give a chain rule for the information contained in the concatenation of adaptive lists (Lemma 6). Then we can construct the sequence F as in the proof sketch above (Lemma 8). However, F is now an adaptive list. Finally, Theorem 2 is proven (page 12) by showing that the adversary A can be considered as an adaptive list G of length q, and therefore cannot distinguish the answers to queries outside F from uniform randomness. For convenience, in Corollary 9 we formulate an asymptotic version of Theorem 2.

We now give the details of the proof, broken down to several lemmas. First we have to define the concept of an adaptive list. To capture the possibility of adaptive queries, an adaptive list is formally just a deterministic oracle Turing machine. An adaptive list of length n makes n queries to the oracle and outputs an assignment containing the queries and the results of these queries. To be able to talk about the concatenation of adaptive lists, we slightly extend this idea. An adaptive list takes an auxiliary input z, but also an assignment X. This assignment can be thought of as the queries made by an adaptive list executed earlier. So in a concatenation of two adaptive lists, the queries of the second adaptive list can depend on the results of the queries made by the first adaptive list. For definitional convenience, an adaptive list does not only output its queries, but also the queries received as input. An adaptive list expecting a queries as input and then making $b - a$ queries, we call an $a \to b$ adaptive list. We require that an adaptive list never repeats a query. Note that an adaptive list is indeed a generalisation of a non-adaptive sequence of queries: a sequence $(x_1, \ldots, x_n)$ corresponds to the $0 \to n$ adaptive list querying the positions x_1 to x_n and returning the results.

Definition 3 (Adaptive list). *Let $\#Domain \geq b \geq a \geq 0$. An $a \to b$* adaptive list *M is defined as a deterministic oracle Turing machine that takes an assignment $X = (x_1 \to y_1, \ldots, x_a \to y_a)$ and a string $z \in \Sigma^*$ as input and satisfies the following properties*

- *$M = M(X, z)$ does not query the oracle at positions $x_1, \ldots, x_a$.*
- *M never queries the oracle twice at the same position.*
- *M queries the oracle exactly $b - a$ times.*
- *Let $x'_1, \ldots, x'_{b-a}$ be the positions of the oracle calls made by M (in that order). Let $y'_i := \mathcal{O}(x'_i)$ be the corresponding oracle answers.*
- *Then M outputs the assignment $(x_1 \to y_1, \ldots, x_a \to y_a, x'_1 \to y'_1, \ldots, x'_{b-a} \to y'_{b-a})$.*

We can now define simple operations on adaptive lists. The length of an adaptive list is the number of queries it makes, and the composition of two adaptive lists is the adaptive list that first queries the first list, and then executes the second, which gets the queries made by the first as input.

Definition 4 (Operations on adaptive lists). *Let M be an $a \to b$ adaptive list. Then the* length *$|M|$ is defined as $|M| := b - a$.*

Let N be an $a \to b$ adaptive list, and M some $b \to c$ adaptive list. Then the composition *$M \circ N$ is defined as the oracle Turing machine that upon input of an assignment X and a string $z \in \Sigma^*$ outputs $M^{\mathcal{O}}(N^{\mathcal{O}}(X, z), z)$.*

Obviously, $M \circ N$ is an $a \to c$ adaptive list, and $|M \circ N| = |M| + |N|$.

We can now define the quantity $J(M|N)$ for adaptive lists M, N. Intuitively, $J(M|N)$ denotes the information that the queries made by M (when executed after N) contain about the auxiliary input $z^{\mathcal{O}}$ beyond what is already known from the queries made by N. Since the results to the queries made by M should be uniformly random if they are independent of $z^{\mathcal{O}}$, we define $J(M|N)$ as the quantity by which the conditional entropy of M's queries given N's queries and $z^{\mathcal{O}}$ is lower than the hypothetical value of $|M| \cdot \log \#\mathit{Range}$.

Definition 5 (Information of an adaptive list). *Let N be some $0 \to b$ adaptive list, and M some $b \to c$ adaptive list. Let further $\mathcal{O}$ be the random oracle and z a random variable (where z does not need to be independent of $\mathcal{O}$).*

Then the information $J(M|N)$ *is defined by*

$$J(M|N) := |M| \cdot \log \#\mathit{Range} - H(M \circ N^{\mathcal{O}}(z)|N^{\mathcal{O}}(z), z).$$

(Note that $J(M|N)$ implicitly depends on the joint distribution of $\mathcal{O}$ and z.)

We write short $J(M)$ for $J(M|\varnothing)$ where $\varnothing$ is the adaptive list making no queries.

We now give two simple properties of the information $J(M|N)$: a chain rule and an upper bound in terms of the auxiliary input's length.

Lemma 6 (Chain rule for the information). *Let N be some $0 \to b$ adaptive list, M_2 some $b \to c$ adaptive list, and M_1 some $c \to d$ adaptive list. Then*

$$J(M_1 \circ M_2|N) \geq J(M_1|M_2 \circ N) + J(M_2|N).$$

Lemma 7 (Bounds for the information). *Let z be a random variable with finite range Z and $p := \log \#Z$. Let F be some $0 \to b$ adaptive list. Then $J(F) \leq p$.*

The proofs of these lemmas as well as of the subsequent ones are given in the full version [18].

Let $J_i := \max_F J(F)$ where F ranges over all adaptive lists of length $|F| = i$. Choose F_i such that $J(F_i) = J_i$. Consider the quantities $J_0, J_q, J_{2q}, \dots, J_{f+q}$ (assuming that q divides f). Since $J_0 \geq 0$ and $J_{f+q} \leq p$ by Lemma 7, there must be some $f' \leq f$ such that $J_{f'+q} - J_{f'} \leq \frac{pq}{f} =: \varepsilon$. Defining $F := F_{f'}$ we get $J(G|F) \leq J(G \circ F_{f'}) - J(F_{f'}) \leq J_{q+f'} - J_{f'} \leq \varepsilon$ by Lemma 6.

By definition of $J(G|F)$, this implies that the results of the queries made by G are only ε away from the maximum possible entropy $|G| \cdot \log \#\mathit{Range}$. This implies using a result from [13] that the statistical distance between those query-results and the uniform distribution is bounded by $\sqrt{\varepsilon/2}$, even when given the results of the queries made by F and the auxiliary input $z^{\mathcal{O}}$. This is formally captured by the following lemma which is the core of the proof of Theorem 2.

Lemma 8 (The adaptive list F). *Let $f, q \geq 1$ be integers. Let z be a random variable with finite range Z (z may depend on the random oracle $\mathcal{O}$) and $p := \log \#Z$. Let U_n denote the uniform distributions on n-tupels over $\#\mathit{Range}$ (independent of z and $\mathcal{O}$).*

Then there is an adaptive list F with $|F| \leq f$, such that for any $|F| \to \min\{|F| + q, \#Domain\}$ adaptive list G, it is

$$\Delta\big(\nabla G \circ F^{\mathcal{O}}(z), F^{\mathcal{O}}(z), z;\ U_{|G|}, F^{\mathcal{O}}(z), z\big) \leq \sqrt{\frac{pq}{2f}}.$$

Here $\nabla G \circ F$ denotes the oracle Turing machine that behaves as $G \circ F$ but only outputs the oracle answers *that G got (instead of also outputting G's input and G's queries). More formally, if $G \circ F^{\mathcal{O}}(z) = (x_1 \to y_1, \dots, x_{|F|+|G|} \to y_{|F|+|G|})$, we have $\nabla G \circ F^{\mathcal{O}}(z) = (y_{|F|+1}, \dots, y_{|F|+|G|})$.*

Using Lemma 8, proving the main Theorem 2 is easy. For some adversary A let $\mu := \Delta\big(A^{\mathcal{O}}(z^{\mathcal{O}});\ A^{\mathcal{P}}(z^{\mathcal{O}})\big)$. By fixing the worst-case random-tape, we can make the adversary A deterministic. Then A's output depends only on its input $z^{\mathcal{O}}$ and the answers to its oracle queries. So if we let A just output the queries it made, the statistical distance μ does not diminish. Further, if we give the presampled queries $S^{\mathcal{O}}$ as an additional input to A, we can assume A to make exactly q distinct queries, and not to query any $x \in S^{\mathcal{O}}$. But then A fulfils the definition of an adaptive list, so by Lemma 8 we have $\mu \leq \sqrt{\frac{pq}{2f}}$, which proves Theorem 2.

We give the full details of the proof in the full version [18].

An interesting question is whether the bound $\sqrt{pq/2f}$ on the statistical distance Δ achieved by Theorem 2 is tight. In particular, the bound falls only sublinearly with f, while we were unable to find a counterexample where Δ did not fall exponentially with f. So a tighter bound may be possible. However, this would need to use new proof techniques, since the approach in this paper uses an averaging argument that will at best give a bound that falls polynomially in f (cf. the computation of $J_{f'+q} - J_{f'}$ below Lemma 7 above.)

Finally, for convenience we state an asymptotic version of Theorem 2 that hides the exact bounds achieved there:

Corollary 9 (Lazy sampling with auxiliary input, asymptotic version). *For any superpolynomial function f and any polynomial q and oracle function z into strings of polynomial length, there is an oracle function S, such that $S^{\mathcal{O}}$ is an assignment of length at most f, so that for any probabilistic oracle Turing machine A making at most q queries, the following random variables are statistically indistinguishable:*

$$A^{\mathcal{O}}(1^k, z^{\mathcal{O}}) \qquad \textit{and} \qquad A^{\mathcal{P}}(1^k, z^{\mathcal{O}}).$$

Here $\mathcal{P}$ is the random oracle with presampling $S^{\mathcal{O}}$.

(In this corollary, $\mathcal{O}$, z, and S depend implicitly on the security parameter k.)

Proof. Immediate from Theorem 2. □

3 Example: one-wayness of the random oracle

To give a first impression on how the lazy sampling technique is used in the random oracle model with oracle-dependent auxiliary input, we show a very simple result: If we let $f := \mathcal{O}$, then f is a one-way function.

In the full version [18], as a second example we show that $f := \mathcal{O}$ is given-preimage collision-resistant.

Lemma 10 (The random oracle is one-way). *Let $g := \mathcal{O}$ where $\mathcal{O}$ denotes the random oracle. Then g is a one-way function in the random oracle model with oracle-dependent auxiliary input.*

More formally, for any probabilistic polynomial-time oracle Turing machine A and any oracle function z into strings of polynomial length, the following probability is negligible (in k):

$$\mathrm{Adv}_A := P\big(x \xleftarrow{\$} \mathit{Domain},\ x' \leftarrow A^{\mathcal{O}}(1^k, z^{\mathcal{O}}, \mathcal{O}(x)) :\ \mathcal{O}(x') = \mathcal{O}(x)\big)$$

(In this lemma, $\mathcal{O}$, Domain, f, and z depend implicitly on the security parameter k.)

We present this proof in some detail, to illustrate how Theorem 2 or Corollary 9 can be used. Since these steps are almost identical in most situations, knowledge of this proof facilitates understanding of the proofs given later on.

Proof. Let $f := \min\{\sqrt{\#\mathit{Range}}, \sqrt{\#\mathit{Domain}}\}$. Let $\tilde{A}$ be the oracle Turing machine that chooses a random x from Domain_k, then let $A(1^k, z^{\mathcal{O}}, \mathcal{O}(x))$ choose x', and outputs 1 if and only if $\mathcal{O}(x') = \mathcal{O}(x)$. Then $\mathrm{Adv}_A = P\big(\tilde{A}^{\mathcal{O}}(1^k, z^{\mathcal{O}}) = 1\big)$.

Since A is polynomial-time, $\tilde{A}$ makes only a polynomial number of queries, so Corollary 9 applies to $\tilde{A}$, hence $\tilde{A}^{\mathcal{O}}(1^k, z^{\mathcal{O}})$ and $\tilde{A}^{\mathcal{P}}(1^k, z^{\mathcal{O}})$ are statistically indistinguishable (where $\mathcal{P}$ is the random oracle with presampling $S^{\mathcal{O}}$, and S is as in Corollary 9). Then consider the following game:

$$\text{Game 1:} \qquad x \xleftarrow{\$} \mathit{Domain},\ x' \leftarrow A^{\mathcal{P}}(1^k, z^{\mathcal{O}}, \mathcal{P}(x)) :\ \mathcal{P}(x') = \mathcal{P}(x).$$

We call the probability that the last expression evaluates to true (i.e., that $\mathcal{P}(x') = \mathcal{P}(x)$) the advantage Adv_1 of the game. Since Adv_1 is the probability that $\tilde{A}^{\mathcal{P}}(1^k, z^{\mathcal{O}})$ outputs 1, $|\mathrm{Adv}_A - \mathrm{Adv}_1|$ is negligible.

(This step probably occurs at the beginning of virtually all proofs that use Theorem 2 or Corollary 9. We are now in the situation that with at most f exceptions, the oracle query $\mathcal{P}(x)$ returns a fresh random value, and can use standard techniques based on lazy sampling.)

We now modify A in the following way resulting in a machine A_2: A_2 expects an assignment S as an additional argument. Whenever A would query the random oracle $\mathcal{P}$ with a value x, A_2 first checks if $x \in S$. If so, A returns the image of x under S. Otherwise, A_2 queries its oracle. Then consider the following game:

$$\text{Game 2:} \qquad x \xleftarrow{\$} \mathit{Domain},\ y \leftarrow \mathcal{P}(x),\ x' \leftarrow A_2^{\mathcal{P}}(1^k, z^{\mathcal{O}}, y, S^{\mathcal{O}}) :\ y = \mathcal{P}(x')$$

Obviously, $\mathrm{Adv}_1 = \mathrm{Adv}_2$.

Since for some $x \notin S^{\mathcal{O}}$ (which happens with probability at least $1 - f/\#\mathit{Domain}$), the oracle $\mathcal{P}$ returns a random $y \in \#\mathit{Range}$, the probability that $y \in \operatorname{im} S^{\mathcal{O}}$ is at most $f/\#\mathit{Domain} + f/\#\mathit{Range}$. Furthermore, if $x' \in S^{\mathcal{O}}$ but $y \notin \operatorname{im} S^{\mathcal{O}}$, the predicate $y = \mathcal{P}(x')$ will be false.

So $|\mathrm{Adv}_2 - \mathrm{Adv}_3| \le P(y \in \mathrm{im}\, S^{\mathcal{O}})$ is negligible for the following game 3:

$$\text{Game 3:} \qquad x \xleftarrow{\$} \mathit{Domain},\ y \leftarrow \mathcal{P}(x),\ x' \leftarrow A_2^{\mathcal{P}}(1^k, z^{\mathcal{O}}, y, S^{\mathcal{O}}) : \\ \texttt{if } (x' \in S^{\mathcal{O}}) \texttt{ then false else } y = \mathcal{P}(x').$$

Note that in game 3, A_2 never queries $\mathcal{P}$ at a position in $S^{\mathcal{O}}$. Furthermore, the query $\mathcal{P}(x')$ is only executed if $x' \notin S$. So $\mathcal{P}$ is only queried at a position in $S^{\mathcal{O}}$, if $x \in S^{\mathcal{O}}$, which has probability at most $f/\#\mathit{Domain}$. But when queried at positions outside $S^{\mathcal{O}}$, $\mathcal{P}$ behaves like a normal random oracle (i.e., without presampling). We can therefore replace the oracle $\mathcal{P}$ by a random oracle $\mathcal{R}$ (independent of $\mathcal{O}$):

$$\text{Game 4:} \qquad x \xleftarrow{\$} \mathit{Domain},\ y \leftarrow \mathcal{R}(x),\ x' \leftarrow A_2^{\mathcal{R}}(1^k, z^{\mathcal{O}}, y, S^{\mathcal{O}}) : \\ \texttt{if } (x' \in S^{\mathcal{O}}) \texttt{ then false else } y = \mathcal{R}(x').$$

Then $|\mathrm{Adv}_3 - \mathrm{Adv}_4| \le P(x \in S^{\mathcal{O}})$ is negligible.

(We have now succeeded in completely separating the oracle from the auxiliary input; $\mathcal{R}$ is independent from $(z^{\mathcal{O}}, S^{\mathcal{O}})$. From here on, the proof is a standard proof of one-wayness of the random oracle. Note however, that $S^{\mathcal{O}}$ has a length that may be superpolynomial, so A_2 is not polynomially bounded any more. In our case, this does not pose a problem, since we only use the fact that A_2 uses a polynomial number of queries. In proof that additionally need computational assumptions, one might need additional tools which we present in Section 4.)

Consider the following game:

$$\text{Game 5:} \qquad x \xleftarrow{\$} \mathit{Domain},\ y \xleftarrow{\$} \mathit{Range},\ x' \leftarrow A_2^{\mathcal{R}}(1^k, z^{\mathcal{O}}, y, S^{\mathcal{O}}) : \\ \texttt{if } (x' \in S^{\mathcal{O}}) \texttt{ then false else } y = \mathcal{R}(x').$$

Since A was polynomially bounded, there is a polynomial q bounding the number of oracle queries of A_2. The probability that A_2 queries $\mathcal{R}$ at position x is therefore at most $q/\#\mathit{Domain}$ (since x is randomly chosen and never used). Furthermore, game 4 and 5 only differ if A_2 queries $\mathcal{R}$ at position x. So $|\mathrm{Adv}_4 - \mathrm{Adv}_5| \le q/\#\mathit{Domain}$ is negligible.

Since A_2 makes at most q queries, the probability that one of these returns y is at most $q/\#\mathit{Domain}$. If x' returns a value x' it has not queried before, the probability that $y = \mathcal{R}(x')$ is at most $1/\#\mathit{Domain}$. So $\mathrm{Adv}_5 \le (q-1)/\#\mathit{Domain}$ is negligible.

Collecting the bounds shown so far, we see that Adv_A is negligible. □

In the preceding proof, we have only verified that the advantage of the adversary is negligible. By using Theorem 2 instead of Corollary 9 and computing the exact bounds, we even get $\mathrm{Adv}_A \in 2^{-\Omega(k)}$ which is essentially the same bound as given in [12] and [10].

4 Security amplification

When using a random oracle with presampling, reduction proofs sometimes run into situations where the adversary gets the presampling $S^{\mathcal{O}}$ as an input. Unfortunately, this presampling is usually of superpolynomial size, so the resulting adversary is not polynomial-time any more and a reduction to complexity assumptions relative to polynomial-time adversaries is bound to fail. (E.g., in the proof of Lemma 10 the adversary A_2 was not polynomial-time in the security parameter any more. In that case however, this did not matter since we only used the polynomial bound on the number of queries made by A_2.) An example of a situation where superpolynomial adversaries occur, and do pose a problem, is the proof that RSA-OAEP is secure with respect to oracle-dependent auxiliary input, cf. Section 5. One possibility is simply to assume a stronger security notion; in the case of RSA-OAEP one could use, e.g., the RSA-assumption against quasi-polynomial adversaries.

Fortunately, there is another way which allows to use standard assumptions (i.e., with respect to polynomial-time adversaries) in many cases. We show that for some kinds of security notions, security against polynomial-time adversaries implies security against adversaries with f-bounded runtime, where f is a suitably chosen *superpolynomial* function. Using this fact we can finish our reduction proof: Corollary 9 guarantees that for any superpolynomial function f', we can replace the random oracle by a random oracle with presampling of length f'. We then choose f' to be the largest function such that all adversaries constructed in our proof are still f-bounded. Such an f' is still superpolynomial, so Corollary 9 applies. On the other hand, the resulting adversaries are efficient enough for the reduction to go through. This proof method is applied in Section 5 to show the security of RSA-OAEP.

Instead of giving a general proof of our *security amplification* technique, we give here a proof for the security notion of partial-domain one-wayness (Definition 11). The proof can easily be adapted to other security notions (in particular, our proof does not exploit how the advantage Adv is defined for this particular notion). In the full version [18] we give a more general characterisation of the security notions for which security amplification is possible.[5]

Definition 11 (Partial-domain one-way). *A family of 1-1 functions $f_{pk} : B \times C \to D$ is* partial-domain one-way*, if for any nonuniform polynomial-time adversary A, the following advantage is negligible:*

$$\mathrm{Adv}_{A,k} := P\big(pk \leftarrow K(1^k),\ (s,t) \stackrel{\$}{\leftarrow} B{\times}C,\ y \leftarrow f_{pk}(s,t),\ s' \leftarrow A(1^k, y) :\ s = s'\big).$$

Here K denotes the index generation algorithm for the family f_{pk} of functions. Partial-domain one-way against f-bounded adversaries *for some function f is defined analogously.*

(In this definition, B, C, and D depend implicitly on the security parameter k.)

Lemma 12 (Security amplification for partial-domain one-wayness). *Let the family f_{pk} be partial-domain one-way (against polynomial-time nonuniform adversaries). Then there exists a superpolynomial function f such that f_{pk} is partial-domain one-way against f-bounded nonuniform adversaries.*

[5] This includes one-wayness, partial-domain one-wayness, IND-CPA, IND-CCA2, black-box stand-alone security of function evaluations, UC (where the amplification concerns the running time of the environment), black-box zero-knowledge, arguments, black-box arguments of knowledge.

Proof. For $n \in \mathbb{N}$ let $\mu_n(k) := \max_{|A| \leq n}(\mathrm{Adv}_{A,k})$ where A goes over all circuits of size at most n. Assume there was a polynomial p with integer coefficients (an integer polynomial for short) such that $\mu_{p(k)}(k)$ is not negligible in k. Then there is a nonuniform adversary A consisting of circuits A_k with $|A_k| \leq p(k)$ such that $\mathrm{Adv}_{A,k} \geq \mu_{p(k)}(k)$ is non-negligible. Since A is polynomial-time, this contradicts the assumption that the f_{pk} are partial-domain one-way. Hence $\mu_p := \mu_{p(k)}(k)$ is negligible for all integer polynomials p.

We say that a function μ asymptotically dominates a function ν if for all sufficiently large k we have $\mu(k) \geq \nu(k)$. [1] proves that for any countable set S of negligible functions, there is a negligible function μ^* that asymptotically dominates all $\mu \in S$.

Therefore, there is a negligible function μ^*, that asymptotically dominates μ_p for every integer polynomial p.

Let $f(k) := \max\{p \in \mathbb{N} : \mu_p(k) \leq \mu^*(k)\}$. Then $\mu_{f^X(k)}(k) \leq \mu^*(k)$ is negligible. So for any nonuniform f-bounded adversary A the advantage $\mathrm{Adv}_{A,k}$ is negligible. Furthermore, we can show that f is superpolynomial. Assume this is not the case. Then there an integer polynomial p such that $p > f$ infinitely often. But then $\mu_p > \mu^*$ holds infinitely often, in contradiction to the choice of μ^* (by definition of f). Thus f is superpolynomial. □

5 OAEP encryption

In [9] it was shown that RSA-OAEP (introduced by [2]) is secure in the random oracle model under the RSA-assumption. However, their proof only covers the case that no auxiliary input is given (or at least that the auxiliary input is not oracle-dependent). In this section, we extend this result to encompass the case of oracle-dependent auxiliary input. On one hand, this gives a nontrivial example of the application of the lazy sampling technique in combination with the security amplification technique. On the other hand, this result is important in its own light, since it gives evidence that RSA-OAEP may be secure with respect to an auxiliary input, even when the random oracle has been instantiated with a fixed function.

To read this section, it is helpful to have at least basic knowledge of the OAEP construction and its proof from [9]. We recommend [9] as an introduction.

Theorem 13 (OAEP is secure with respect to oracle-dependent auxiliary input). *Let f_{pk} be a family of partial-domain one-way trapdoor 1-1 functions (with the property, that the elements of the domain of f_{pk} consist of two components each of superlogarithmic length).*

Then the OAEP encryption scheme based on f_{pk} is IND-CCA2 secure in the random oracle model with oracle-dependent auxiliary input.

This theorem implies that RSA-OAEP is IND-CCA2 secure under the RSA-assumption with respect to oracle-dependent auxiliary input, since in [9] it is shown that the RSA family of functions is partial-domain one-way.

At this point, we only describe on a high level, in what points our proof differs from the proof in [9]. In the full version [18], we reproduce the full proof of [9] and highlight our changes for comparison.

In [9], the proof has roughly the following outer form: First, the IND-CCA2 game is formulated for the special case of the OAEP encryption scheme. Then the game is rewritten in a series of small changes, to finally yield a plaintext extractor. If the first game had a non-negligible success probability (i.e., the OAEP encryption scheme was not IND-CCA2 secure), the plaintext extractor had, for some random ciphertext $f_{pk}(s,t)$, a non-negligible probability of outputting s. This breaks the assumption that f_{pk} is partial-domain one-way.

Our proof starts with the same game, except that the adversary now has access to an oracle-dependent auxiliary input $z^{\mathcal{O}}$. Then we can use Corollary 9 to replace the random oracle $\mathcal{O}$ by a random oracle $\mathcal{P}$ with presampling $S^{\mathcal{O}}$ of a yet to determine superpolynomial subexponential length $\mathfrak{f}$ (similar to the first step in the proof of Lemma 10).[6] In this new situation, for randomly chosen $x \in Domain$, with overwhelming probability, the oracle response $\mathcal{P}(x)$ is uniformly distributed. Using this fact, most of the rewriting steps in the sequence of games are the same as in [9], sometimes with slightly larger errors to account for the possibility of randomly choosing an $x \in S^{\mathcal{O}}$. Only in the construction of the plaintext extractor additional care has to be taken. Here the original argument uses that the answer to an oracle query can be assumed to be random if the adversary has not yet queried it. From this they conclude any ciphertext the decryption oracle would accept can also be decrypted by encrypting and comparing all oracles queries that have been made by the adversary so far. This does not hold any more since the auxiliary input $z^{\mathcal{O}}$ can supply additional information on the presampled queries $S^{\mathcal{O}}$. We thus have to change the plaintext extractor not only to encrypt all oracle queries but also all presampled queries $S^{\mathcal{O}}$. Therefore the plaintext extractor is not polynomial-time anymore, but instead a nonuniform machine with running time $p(\mathfrak{f})$ for some polynomial p. We consequently do not directly obtain a contradiction to the partial-domain one-wayness, since therefore the plaintext extractor would have to be polynomial-time.

However, we can use the security amplification technique. By Lemma 12, there is a superpolynomial function f' such that f_{pk} is partial-domain one-way even against nonuniform f'-bounded adversaries. By choosing $\mathfrak{f}$ small enough (but still superpolynomial), it is $p(\mathfrak{f}) \leq f'$, so the plaintext extractor is f'-bounded, and the fact that the plaintext extractor returns s for some $f_{pk}(s,t)$ with non-negligible probability is a contradiction.

6 Open questions

We have shown how to apply the lazy sampling technique to the case of oracle-dependent auxiliary input. Going further, the following open problems come to mind:

- Polynomial presampling: In Corollary 9, we require the length f of the presampling to be superpolynomial. This makes reduction proofs more difficult, in particular it necessitates the use of the security amplification technique. It would be preferable to be able to use a polynomial length f (in this case, the length would of course have to depend on the length of auxiliary input and the number of queries made by the adversary).

[6] The actual proof uses Theorem 2, but the asymptotic version is sufficient.

– The random oracle as considered here is only a specific example of the class of random objects that are given as oracle to the parties. Other examples include random permutations (with or without access to the inverse), the generic group model, ideal ciphers, or just random oracles with a skewed distribution. When using these to motivate security results, the same arguments apply as in the case of random oracles, and oracle-dependent auxiliary input should be considered. It is then necessary to extend the lazy sampling technique to these constructions as well.

In the full version [18], we discuss these open questions in slightly more detail.

Acknowledgements. We thank Michael Backes, Dennis Hofheinz, Yuval Ishai, Jörn Müller-Quade, Hoeteck Wee, and Jürg Wullschleger for valuable discussions. We further thank the anonymous referees for helpful comments.

References

1. Mihir Bellare. A note on negligible functions. *Journal of Cryptology*, 15(4):271–284, 2002.
2. Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption—how to encrypt with RSA. In *Proceedings of EUROCRYPT '94*, pages 92–111.
3. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of CCS 1993*, pages 62–73.
4. Christian Cachin, Claude Crépeau, and Julien Marcil. Oblivious transfer with a memory-bounded receiver. In *Proceedings of STOC 2002*, pages 493–502.
5. Christian Cachin and Ueli Maurer. Unconditional security against memory-bounded adversaries. In *Proceedings of CRYPTO '97*, pages 292–306.
6. Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. In *Proceedings of STOC 1998*, pages 209–218.
7. Stefan Dziembowski and Ueli Maurer. Tight security proofs for the bounded-storage model. In *Proceedings of STOC 2002*, pages 341–350.
8. Federal Information Processing Standards Publications. *FIBS PUB 180-2: Secure Hash Standard (SHS)*, August 2002.
9. Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. *Journal of Cryptology*, 17(2):81–104, 2004.
10. Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM J Computing*, 35(1):217–246, 2005.
11. Oded Goldreich. A uniform-complexity treatment of encryption and zero-knowledge. *Journal of Cryptology*, 6(1):21–53, 1993.
12. Russel Impagliazzo. Very strong one-way functions and pseudo-random generators exist relative to a random oracle. Manuscript, 1996.
13. Solomon Kullback. A lower bound for discrimination information in terms of variation (corresp.). *IEEE Transactions on Information Theory*, 13(1):126–127, 1967.
14. Ueli Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.
15. Rafael Pass. On deniability in the common reference string and random oracle model. In *Proceedings of CRYPTO 2003*, volume 2729, pages 316–337.
16. Phillip Rogaway. Formalizing human ignorance: Collision-resistant hashing without the keys. In *Proceedings of Vietcrypt 2006*, pages 221–228.
17. RSA Laboratories. *PKCS #1: RSA Cryptography Standard, Version 2.1*, 2002.
18. Dominique Unruh. Random oracles and auxiliary input. IACR ePrint 2007/168. Full version of this paper.
19. Hoeteck Wee. Zero knowledge in the random oracle model, revisited. Manuscript, 2006.