Full Indifferentiable Security of the Xor of Two or More Random Permutations Using the χ^2 Method

Srimanta Bhattacharya(✉) and Mridul Nandi

Indian Statistical Institute, Kolkata, India
mail.srimanta@gmail.com, mridul.nandi@gmail.com

Abstract. The construction XORP (bitwise-xor of outputs of two independent n-bit random permutations) has gained broad attention over the last two decades due to its high security. Very recently, Dai *et al.* (CRYPTO'17), by using a method which they term the *Chi-squared method* (χ^2 method), have shown n-bit security of XORP when the underlying random permutations are kept secret to the adversary. In this work, we consider the case where the underlying random permutations are publicly available to the adversary. The best known security of XORP in this security game (also known as *indifferentiable security*) is $\frac{2n}{3}$-bit, due to Mennink *et al.* (ACNS'15). Later, Lee (IEEE-IT'17) proved a better $\frac{(k-1)n}{k}$-bit security for the general construction XORP$[k]$ which returns the xor of k (≥ 2) independent random permutations. However, the security was shown only for the cases where k is an even integer. In this paper, we improve all these known bounds and prove full, *i.e.,* n-bit (indifferentiable) security of XORP as well as XORP$[k]$ for any k. Our main result is n-bit security of XORP, and we use the χ^2 method to prove it.

Keywords: Random permutation · Indifferentiable security
χ^2 method · XOR construction · Simulator

1 Introduction

The problem to construct *pseudorandom functions* (PRFs) from *pseudorandom permutations* (PRPs) is called "Luby-Rackoff Backwards" [BKR98] (referring to the well known work of Luby and Rackoff who showed how to construct a PRP from a PRF [LR88]). In [BKR98], the authors considered two sequential block cipher calls, where the output of the first call is the key input to the second one. However, this construction achieves security only up to the birthday bound on the output size. Achieving security beyond the birthday bound is somewhat non-trivial. Xoring the outputs of two independent n-bit *random permutations*[1]

[1] In this work, we will essentially focus on information theoretic security in the *ideal model*. Therefore, the permutations and functions that we will consider, will be random (and not pseudorandom).

J. B. Nielsen and V. Rijmen (Eds.): EUROCRYPT 2018, LNCS 10820, pp. 387–412, 2018.
https://doi.org/10.1007/978-3-319-78381-9_15

is a very simple way to construct *random functions* from random permutations. We call it the *XOR construction* and denote it as XORP. We also consider a generalized version of the XOR construction in which we xor k independent n-bit random permutations, and denote it as XORP$[k]$. Lucks [Luc00] showed beyond the birthday bound security for XORP$[k]$ for all $k \geq 2$. In particular, he showed that the construction achieves at least $\frac{kn}{k+1}$-bit security. This bound was further improved in a sequence of papers [BI99, CLP14, Pat10, Pat08b]. Very recently, Dai *et al.* [DHT17] have shown n-bit security for XORP. Earlier, Mennink *et al.* [MP15] showed a reduction proving that the security of XORP$[k]$ can be reduced to that of XORP for any $k \geq 3$. Hence, XORP$[k]$ also achieves n-bit security. The XORP (or its general version XORP$[k]$) construction is important since it has been used to obtain some constructions achieving beyond the birthday bound (or sometimes almost full) security (e.g., CENC [Iwa06, IMV16], PMAC_Plus [Yas11], and ZMAC [IMPS17]).

Moving from secret to public random permutation. While to a certain degree it is possible to view the permutations as secret, there are many reasons to consider the setting where they are public. For example, we sometimes instantiate block ciphers with fixed keys. Moreover, many unkeyed permutations are designed as an underlying primitive of encryption [BDPVA11a], MAC [BDPVA11b], hash functions [BDP+13, RAB+08, Wu11, GKM+09], etc. The CAESAR competition [CAE] received various permutation-based authenticated encryptions, and all of these constructions have been analyzed in the public permutation model.

The security game, in this setting, is clearly different from the standard indistinguishable model due to the public access of the adversary to the underlying permutations. An appropriate notion is the indifferentiability framework, introduced by Maurer *et al.* [MRH04]. Informally, it gives a sufficient condition under which an ideal functionality can be replaced by an indifferentiable-secure construction based on ideal, publicly available underlying primitives. We note that the security game for indifferentiability is also an indistinguishability game in which one has to design a simulator aimed to simulate the underlying primitive. In the past, many constructions were analyzed (e.g., [AMP10, BDPVA08, BMN10]) under this security notion.

Known indifferentiable security bounds of XORP and XORP$[k]$. In this indifferentiability model, Mandal *et al.* [MPN10] proved $\frac{2n}{3}$-bit security for XORP. Later, Mennink *et al.* [MP15] pointed out a subtle but non-negligible flaw in their proof and fixed the security proof. Recently, Lee [Lee17] has shown improved security for the general construction XORP$[k]$. In particular, he has proved $\frac{(k-1)n}{k}$-bit security for the general construction XORP$[k]$ when k is an even integer. Table 1 summarizes the state-of-the-art for XORP and XORP$[k]$ in the public permutation setting.

Table 1. A brief comparison of known bounds and our bounds for the constructions XORP and XORP[k]. Here q denotes the total number of queries made by the adversary to all oracles.

Construction	Best known bound	Our bound
XORP	$q^3/2^{2n}$ [MP15]	$\sqrt{q/2^n}$
XORP[k]	$\frac{q^{k+1}}{2^{nk}}$ ($k \geq 4$ even) [Lee17]	$\sqrt{q/2^n}$

Mirror theory and its limitation. Patarin introduced a combinatorial problem motivated from the PRF-security of XORP[k] type constructions. Informally, *mirror theory* (see [Pat10]) provides a suitable lower bound on the number of solutions satisfying a system of linear equations involving exactly two variables at a time. Together with the *H-coefficient technique* [Vau03, Pat08a, IMV16], this leads to a bound on the PRF-distinguishing advantage of XORP. The mirror theory seems to be very powerful as it can be applied to prove optimal security for many constructions such as EDM, EWCDM, etc. [MN17a, MN17b]. However, the proof of the mirror theory is quite complex with some of its steps lacking necessary details. Later, Patarin [CLP14] himself provided a simpler alternative but sub-optimal proof for XORP[k] (which is a trivial corollary of the mirror theory).

One may wonder whether the same technique can be applied to the indifferentiability setup or not. Here, we note that the mirror theory puts a constraint on the system of equations so that no equation in one variable can be generated through linear combination of equations from the system. On the other hand, in the indifferentiable security game, the adversary can make public permutation calls and observe the responses. So, along with the two variables linear equations, we also have to consider several single variable equations. This shows the limitation of the mirror theory in this setup.

Our contribution and the proof technique. Proving full security of XORP in the public permutation model was an open problem so far. The original simulator [MPN10], used in the security proof of XORP, is conjectured to allow for security up to 2^n queries. However, the authors of [MP15] expressed this as a highly non-trivial exercise. In this paper, we resolve this open problem and prove n-bit indifferentiable security of XORP and XORP[k] for all $k \geq 3$. Full indifferentiable security of XORP is our main result which we state and prove in Theorem 2. Subsequently, in Theorem 3, we show full indifferentiability of XORP[k]; for this, we reduce the security of XORP[k]($k \geq 3$), to the security of XORP, and then apply our main result.

The simulator (described in Sect. 3) that we consider in the security proof of XORP follows the same steps as the simulator of [MPN10, Lee17] in the case of forward queries. However, the simulator differs in the responses to the backward queries. In the case of backward queries, the simulator queries the ideal random function repeatedly (about n times) until it succeeds in its goal.

We follow the recently introduced χ^2 method [DHT17] to prove our claim. This method was implicitly used by Stam [Sta78] while proving a bound on the total variation between a *truncated random permutation* and a random function. Though in a purely statistical context, (to the best of our knowledge) Stam's work can be viewed as the origin of the χ^2 method, which led to a bound on the PRF-security of the truncated random permutation construction (see [GG16, GGM17] for recent results and discussion on this construction). In [DHT17], the authors used this method to obtain bounds on the PRF-security of XORP and the EDM construction [CS16a, MN17b]. Also, using this method full PRF-security of variable output length XOR pseudorandom functions has been shown [BN18].

In this paper, we show another application of the χ^2 method in (symmetric-key) cryptography in the context of XORP$[k]$ type construction. Our main result demonstrates the power of this method as the proof of full security of XORP, in the indifferentiability setup, becomes very hard with the existing methods. However, our proof using the χ^2 method is not a straightforward extension of the proof in the indistinguishability framework due to Dai *et al.*; it is somewhat complicated as, unlike in the indistinguishability framework, we will need to consider the primitive queries (*i.e.,* outputs of the individual permutations). Moreover, we will have to handle the backward queries whose analysis is somewhat involved.

Outline of the paper. In the next section, we cover the preliminaries where we discuss the notion of indifferentiability and the χ^2 method. In Sect. 3, we describe the simulator that we consider in the proof of our main result (Theorem 2). In Sect. 4, we state and prove Theorem 2. Some auxilliary proofs, used in the proof of Theorem 2, are given in Sect. 5. Finally, in Sect. 6, we show full indifferentiability of XORP$[k]$.

2 Preliminaries

In this section, we cover the technical preliminaries required to understand our results. We begin with the notational setup. Then we recall the preliminary security notions related to adversary and its advantage in the context of an indistinguishability game. This is to motivate our subsequent discussion on the notion of indifferentiability. Finally, we briefly describe the χ^2 method which is our main tool.

Notational convention. We will use upper case letters to denote random variables and their corresponding lower case letters to denote particular realizations of the variables. Given an integer s we will use the notation X^s to denote the tuple $(X_1, \ldots, X_s)$ of random variables and use x^s to denote the tuple $(x_1, \ldots, x_s)$ of corresponding realizations. Moreover, we write $\{X^s\}$ to denote the set $\{X_i : 1 \leq i \leq s\}$. Given a set $\mathcal{S}$, we will write $X \leftarrow_{\$} \mathcal{S}$ to mean that X is sampled uniformly at random from the set $\mathcal{S}$.

2.1 Adversary and Advantage

Here, we recall the notion of adversarial advantage in the context of a generic *indistinguishability game*. An *adversary* $\mathscr{A}$ is an *oracle algorithm* that interacts with an *oracle* $\mathscr{O}$ through queries and responses. Finally, it returns a bit $b \in \{0,1\}$. We express this as $\mathscr{A}^{\mathscr{O}} \to b$.

In an *indistinguishability game*, $\mathscr{A}$ interacts with two oracles $\mathscr{O}_1$ and $\mathscr{O}_2$. The goal of $\mathscr{A}$ is to distinguish between $\mathscr{O}_1$ and $\mathscr{O}_2$ only from the corresponding queries and responses. The *advantage* of the adversary in this game, denoted $\mathsf{Adv}_{\mathscr{A}}(\mathscr{O}_1, \mathscr{O}_2)$, is given by

$$\mathsf{Adv}^{\mathrm{dist}}_{\mathscr{O}_1,\mathscr{O}_2}(\mathscr{A}) := |\mathbf{Pr}[\mathscr{A}^{\mathscr{O}_1} \to 1] - \mathbf{Pr}[\mathscr{A}^{\mathscr{O}_2} \to 1]|,$$

where the probabilities are taken over the random coins of $\mathscr{A}, \mathscr{O}_1$, and $\mathscr{O}_2$.

In this work, we will focus on the information theoretic security of the constructions (XORP and $\mathsf{XORP}[k]$). So, we let $\mathscr{A}$ to be computationally unbounded. Therefore, without loss of any generality, we assume $\mathscr{A}$ to be deterministic (it can always fix its internal coin tosses to those which maximizes its advantage). However, we restrict $\mathscr{A}$ to only q queries. Let the corresponding replies from $\mathscr{O}_1$ and $\mathscr{O}_2$ be $X_1^q = (X_{1,1}, \ldots, X_{1,q})$ and $X_2^q = (X_{2,1}, \ldots, X_{2,q})$ respectively. Note that X_1^q and X_2^q are random variables that capture the randomness of the oracles $\mathscr{O}_1$ and $\mathscr{O}_2$ respectively. Both X_1^q and X_2^q are distributed over the output alphabet $\Omega^q = \Omega \times \cdots \times \Omega$ of the oracles. Then in this setting, it is not difficult to see that

$$\begin{aligned}\mathsf{Adv}^{\mathrm{dist}}_{\mathscr{O}_1,\mathscr{O}_2}(\mathscr{A}) &= |\mathbf{Pr}[\mathscr{A}^{\mathscr{O}_1} \to 1] - \mathbf{Pr}[\mathscr{A}^{\mathscr{O}_2} \to 1]| \\ &\leq \max_{\mathscr{E} \subseteq \Omega^q} \sum_{x^q \in \mathscr{E}} (\mathbf{Pr}[X_1^q = x^q] - \mathbf{Pr}[X_2^q = x^q]). \qquad (1)\end{aligned}$$

The quantity on the r.h.s. of (1) is the *statistical distance* or the *total variation distance* between X_1^q and X_2^q. We will consider it slightly more formally in Sect. 2.3. We denote by $\mathsf{Adv}^{\mathrm{dist}}_{\mathscr{O}_1,\mathscr{O}_2}(q)$ the maximum of the distinguishing advantages $\mathsf{Adv}^{\mathrm{dist}}_{\mathscr{O}_1,\mathscr{O}_2}(\mathscr{A})$ among all the adversaries $\mathscr{A}$ making at most q queries.

2.2 Indifferentiability

The notion of indifferentiability was introduced by Maurer *et al.* in [MRH04]. It is a stronger security notion than indistinguishability in the following sense. Informally, let a construction T have oracle access to an ideal primitive F. Then in an indistinguishability game, when T is presented as an oracle to the adversary $\mathscr{A}$, it can only query T in a *black-box* manner, i.e., $\mathscr{A}$ can not query F. Whereas in the indifferentiability game, $\mathscr{A}$ can query both T and F.

As shown in Fig. 1, in the indifferentiability game, in the *real world*, a construction T has oracle access to an *ideal primitive* F. On the other hand, in the *ideal world*, the simulator S has access to another ideal primitive G. $\mathscr{A}$ can query any of these four entities with the goal of distinguishing between the two worlds. In this case, $\mathscr{A}$'s advantage can be written as

$$\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{T}^{\mathsf{F}},\mathsf{G}^{\mathsf{S}}}(\mathscr{A}) = |\mathbf{Pr}[\mathscr{A}^{\mathsf{T},\mathsf{F}} \to 1] - \mathbf{Pr}[\mathscr{A}^{\mathsf{G},\mathsf{S}} \to 1]|.$$

In order to prove indifferentiability of T from G, it is sufficient to construct a simulator S in such a way that $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{T}^{\mathsf{F}},\mathsf{G}^{\mathsf{S}}}(\mathscr{A})$ becomes negligible for any adversary $\mathscr{A}$. The following definition captures this idea more formally.

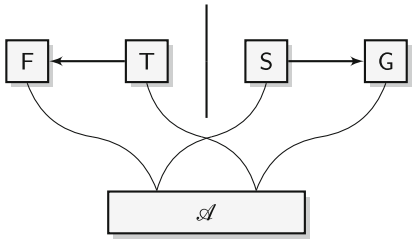

Fig. 1. Indifferentiability game

Definition 1 (Indifferentiability [MRH04]). *A Turing machine* T *with oracle access to an ideal primitive* F *is said to be* $(t, q_{\mathsf{T}}, q_{\mathsf{F}}, \varepsilon)$*-indifferentiable from an ideal primitive* G *if there exists a simulator* S *with oracle access to* G *and running time at most* t*, such that for any adversary* $\mathscr{A}$*, it holds that*

$$\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{T}^{\mathsf{F}},\mathsf{G}^{\mathsf{S}}}(\mathscr{A}) < \varepsilon.$$

$\mathscr{A}$ *makes at most* q_{T} *queries to* T *or* G *and at most* q_{F} *queries to* F *or* S*. Similarly,* T^{F} *is said to be computationally indifferentiable from* G *if the running time of* $\mathscr{A}$ *is bounded above by a polynomial in the security parameter and* ε *is a negligible function of the security parameter.*

Remark 1. For our purpose, we will not consider the parameter t. Also, we will not consider q_{T} and q_{F} separately and focus on their sum $q = q_{\mathsf{T}} + q_{\mathsf{F}}$, which is the total number of queries made by $\mathscr{A}$. Moreover, when F and S are adequately understood we will write the advantage term as $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{T},\mathsf{G}}(\mathscr{A})$.

We write $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{T},\mathsf{G}}(q) = \max_{\mathscr{A}} \mathsf{Adv}^{\mathrm{diff}}_{\mathsf{T},\mathsf{G}}(\mathscr{A})$, where maximum is taken over all adversaries making at most q queries to its oracles.

Indifferentiable security of XORP and $\mathsf{XORP}[k]$. We first describe the XORP and $\mathsf{XORP}[k]$ constructions. Let Perm denote the set of all permutations over the set $\{0,1\}^n$. Let Π_0 and Π_1 be two independent random permutations, *i.e.*,

$\Pi_0, \Pi_1 \leftarrow_\$ \mathsf{Perm}$. The XORP construction takes an input x from $\{0,1\}^n$ and returns the element $\Pi_0(x) \oplus \Pi_1(x)$. This construction can be further generalized to k permutations. Let $\Pi_0, \ldots, \Pi_{k-1}$ be k independent random permutations. We define

$$\mathsf{XORP}[k](x) = \bigoplus_{i=0}^{k-1} \Pi_i(x). \tag{2}$$

So, XORP[2] is same as XORP. Now, we describe the setting of indifferentiable security in our context.

Real world. In the real world, the construction XORP has oracle access to the random permutations Π_0 and Π_1. When the adversary $\mathscr{A}$ queries the construction XORP with a value $x \in \{0,1\}^n$, XORP queries the oracles Π_0 and Π_1 with x and receives back $\Pi_0(x)$ and $\Pi_1(x)$ respectively. Finally, it computes $\Pi_0(x) \oplus \Pi_1(x)$ and returns it to $\mathscr{A}$. In addition to querying the XORP construction, $\mathscr{A}$ can directly query the oracles Π_0 and Π_1 and obtain the values of $\Pi_0(y), \Pi_1(y), \Pi_0^{-1}(y)$, and $\Pi_1^{-1}(y)$ for any $y \in \{0,1\}^n$. The queries for $\Pi_0(y)$ and $\Pi_1(y)$ are *forward queries* and the queries $\Pi_0^{-1}(y)$ and $\Pi_1^{-1}(y)$ are *backward queries*.

Ideal world. In the ideal world, $\mathscr{A}$ queries the random function \$ and the simulator S. S has oracle access to \$. However, S does not have access to (the transcripts of) the interactions between $\mathscr{A}$ and \$. The purpose of S is to simulate the output behavior of the oracles Π_0 and Π_1. That is, for $b \in \{0,1\}$, when $\mathscr{A}$ makes a forward query (x,b) with $x \in \{0,1\}^n$, S returns a random variable $V_b \in \{0,1\}^n$. So, for $b \in \{0,1\}$, V_b simulates $\Pi_\mathsf{b}(x)$. Similarly, when $\mathscr{A}$ makes a backward query (y,b) (with $y \in \{0,1\}^n$ and $b \in \{0,1\}$) S returns a random variable $V_b \in \{0,1\}^n \cup \{\perp\}$. $V_b \in \{0,1\}^n$ simulates ${\Pi_\mathsf{b}}^{-1}(y)$. The output $V_b = \perp$ indicates that S aborted after a fixed number of iterations. This will be more clear when we will describe the simulator S in Sect. 3. In order to prove that XORP is indifferentiable from \$ it is enough to construct simulator S in such a way that no adversary $\mathscr{A}$ can distinguish between the distributions of V_b and Π_b. In other words, advantage of any adversary $\mathscr{A}$, which, in this case, can be written as below,

$$\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(\mathscr{A}) = |\mathbf{Pr}[\mathscr{A}^{\mathsf{XOR},(\Pi_0,\Pi_1,\Pi_0^{-1},\Pi_1^{-1})} \to 1] - \mathbf{Pr}[\mathscr{A}^{\$,\mathsf{S}} \to 1]|$$

becomes negligible. In our case, we will restrict $\mathscr{A}$ to q queries and obtain a concrete upper bound on $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(\mathscr{A})$ (in terms of parameters q and n). This will be sufficient to show indifferentiability of XORP with \$. For the XORP[k] construction, we obtain similar upper bound on $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP}[k],\$}(\mathscr{A})$.

2.3 χ^2 Method for Bounding Total Variation

Here, we provide a brief description of the χ^2 method. Given a set Ω, let $X^q := (X_1, \ldots, X_q)$ and $Z^q := (Z_1, \ldots, Z_q)$ be two random vectors distributed over

$\Omega^q = \Omega \times \cdots \times \Omega$ (q times) according to the distributions $\mathbf{P_0}$ and $\mathbf{P_1}$ respectively. Then the *total variation distance* or *statistical distance* between the distributions $\mathbf{P_0}$ and $\mathbf{P_1}$ is defined as

$$\|\mathbf{P_0} - \mathbf{P_1}\| := \frac{1}{2} \sum_{x^q \in \Omega^q} |\mathbf{P_0}(x^q) - \mathbf{P_1}(x^q)| = \max_{\mathscr{E} \in \Omega^q} \left(\sum_{x^q \in \mathscr{E}} \mathbf{P_0}(x^q) - \mathbf{P_1}(x^q) \right).$$

In what follows, we will require the following conditional distributions.

$$\mathbf{P_{0}}_{|x^{i-1}}(x_i) := \mathbf{Pr}[X_i = x_i \mid X_1 = x_1, \ldots, X_{i-1} = x_{i-1}],$$
$$\mathbf{P_{1}}_{|x^{i-1}}(x_i) := \mathbf{Pr}[Z_i = x_i \mid Z_1 = x_1, \ldots, Z_{i-1} = x_{i-1}].$$

When $i = 1$, $\mathbf{P_{0}}_{|x^{i-1}}[x_1]$ represents $\mathbf{Pr}[X_1 = x_1]$. Similarly, for $\mathbf{P_{1}}_{|x^{i-1}}[x_1]$. Let $x^{i-1} \in \Omega^{i-1}$, $i \geq 1$. The χ^2-distance[2] between these two conditional probability distributions is defined as

$$\chi^2(\mathbf{P_{0}}_{|x^{i-1}}, \mathbf{P_{1}}_{|x^{i-1}}) := \sum_{x_i \in \Omega} \frac{(\mathbf{P_{0}}_{|x^{i-1}}(x_i) - \mathbf{P_{1}}_{|x^{i-1}}(x_i))^2}{\mathbf{P_{1}}_{|x^{i-1}}(x_i)}. \tag{3}$$

Note that for the above definition to work, it is required that the support of the distribution $\mathbf{P_{0}}_{|x^{i-1}}$ be contained within the support of the distribution $\mathbf{P_{1}}_{|x^{i-1}}$. Further, when the distributions $\mathbf{P_{0}}_{|x^{i-1}}$ and $\mathbf{P_{1}}_{|x^{i-1}}$ are clear from the context we will use the notation $\chi^2(x^{i-1})$ for $\chi^2(\mathbf{P_{0}}_{|x^{i-1}}, \mathbf{P_{1}}_{|x^{i-1}})$.

In a very recent work [DHT17], Dai *et al.* introduced a new method, which they term the χ^2 method (Chi-squared method), to bound the statistical distance between two joint distributions in terms of the expectations of the χ^2-distances of the corresponding conditional distributions. At the heart of the χ^2 method is the following theorem, stated in our notation and setting.

Theorem 1 ([DHT17]). *Following the notation as above and suppose the support of the distribution* $\mathbf{P_{0}}_{|x^{i-1}}$ *is contained within the support of the distribution* $\mathbf{P_{1}}_{|x^{i-1}}$ *for all* x^{i-1}*, then*

$$\|\mathbf{P_0} - \mathbf{P_1}\| \leq \left(\frac{1}{2} \sum_{i=1}^{q} \mathbf{Ex}[\chi^2(X^{i-1})] \right)^{\frac{1}{2}}, \tag{4}$$

where for each i*, the expectation is over the* $(i-1)$*-th marginal distribution of* $\mathbf{P_0}$*.*

As an aside, we mention that the main ingredients of the proof of Theorem 1 are (*i*) Pinsker's inequality, (*ii*) chain rule of *Kullback-Leibler divergence* (KL divergence), and (*iii*) Jensen's inequality[3]: Pinsker's inequality upper bounds statistical distance between the distributions by the KL divergence between the

[2] χ^2-distance has its origin in mathematical statistics dating back to the work of Pearson (see [LV87]). It may be observed that χ^2-distance is not symmetric and hence it is not a metric.

[3] See [CT06] for a background on these topics.

two distributions, chain rule of KL divergence expresses the KL divergence of two joint distributions as the sum of the KL divergences between corresponding conditional distributions, and finally Jensen's inequality is used to upper bound the KL divergence between two distributions by their χ^2-divergence (χ^2-distance).

In [DHT17], Dai *et al.* have applied Theorem 1 to show PRF-security of two well known constructions, namely the xor of two random permutations [Pat08b, Pat10, BI99, Luc00] and the encrypted Davies-Meyer (EDM) construction [CS16a, MN17a]. Subsequently, in [BN18], this method has been applied to prove full PRF-security of the variable output length XOR pseudorandom function construction. This method seems to have potential for further application to obtain better bounds (and simplified proofs) on the PRF-security of other constructions where proofs so far have evaded more classical methods, such as the H-coefficient method [Pat08a]. In fact, much earlier, Stam [Sta78] used this method, implicitly and in a purely statistical context, to obtain a PRF-security bound of the truncated random permutation construction.

3 Simulator and Transcripts

3.1 Description of the Simulator

Here, we describe the simulator S used in the proof of Theorem 2.[4] The goal of the simulator S is to mimic the permutations $\Pi := (\Pi_0(.), \Pi_1(.), {\Pi_0}^{-1}(.), {\Pi_1}^{-1}(.))$ in such a way that (XORP, Π) and $(\$, \mathsf{S})$ look indistinguishable. So, S has interfaces corresponding to the forward and backward queries of the random permutations Π_0 and Π_1. Formally, S consists of a pair of stateful randomized algorithms $\mathsf{SIM}_{\mathsf{FWD}}$ (which is invoked for the responses to the forward queries) and $\mathsf{SIM}_{\mathsf{BCK}}$ (which is invoked for responses to the backward queries). More precisely, for $x \in \{0,1\}^n$ and $b \in \{0,1\}$, when an adversary $\mathscr{A}$ makes a forward query (x, b) to S, S runs the algorithm $\mathsf{SIM}_{\mathsf{FWD}}$ and returns a random variable $V_b \in \{0,1\}^n$. So, for $b \in \{0,1\}$, V_b simulates $\Pi_{\mathsf{b}}(x)$. Similarly, when $\mathscr{A}$ makes a backward query (y, b) (with $y \in \{0,1\}^n$ and $b \in \{0,1\}$) to S, S runs the algorithm $\mathsf{SIM}_{\mathsf{BCK}}$ and returns a random variable $V_b \in \{0,1\}^n \cup \{\perp\}$. $V_b \in \{0,1\}^n$ simulates ${\Pi_{\mathsf{b}}}^{-1}(y)$. Note that S has access to the random function \$ which returns random elements from $\{0,1\}^n$ on every fresh query. The goal of the simulator S is to simulate the output behavior of $\Pi_0(.), \Pi_1(.), {\Pi_0}^{-1}(.)$, and ${\Pi_1}^{-1}(.)$ in the ideal world in such a way that it remains consistent with the XORP construction, which is given by the condition

$$\$(x) = \mathsf{SIM}_{\mathsf{FWD}}(x, 0) \oplus \mathsf{SIM}_{\mathsf{FWD}}(x, 1) \text{ for } x \in \{0,1\}^n.$$

However, S may fail to maintain the condition. Whenever it fails (which happens only for the backward queries), $\mathsf{SIM}_{\mathsf{BCK}}$ returns $\perp$. Before returning $\perp$ it makes several attempts where it interacts with \$. If after n attempts it fails to maintain the condition (we will show that would happen with very low probability), it aborts. $V_b = \perp$ indicates that event.

[4] We will consider another simulator in the proof of Theorem 3.

Description of the internal state. In order to be consistent with its replies, *i.e.*, to output the same V_b corresponding to the same queries (forward or backward), S is stateful, *i.e.*, it maintains a history of all the previous interactions (*i.e.*, queries and responses) with $\mathscr{A}$. To do this, S internally maintains three sets $\mathscr{D}, \mathscr{R}_0$, and $\mathscr{R}_1$, and also maintains two lists (indexed by elements of $\mathscr{D}$) $\mathscr{L}_0, \mathscr{L}_1$.

The set $\mathscr{D}$ contains all $x \in \{0,1\}^n$ belonging to the forward queries (x, b) made by $\mathscr{A}$ and all $V_b \in \{0,1\}^n$ that the simulator output on a backward query made by $\mathscr{A}$. For $b \in \{0,1\}$, the set $\mathscr{R}_b$ contains all $y \in \{0,1\}^n$ belonging to the backward queries (y, b) made by $\mathscr{A}$ along with all V_b output by S on a forward query made by $\mathscr{A}$. The lists $\mathscr{L}_0, \mathscr{L}_1$ capture the input-output mapping of S. More precisely, for $b \in \{0,1\}, x \in \mathscr{D}, y \in \mathscr{R}_b$, $\mathscr{L}_b(x) = y$ implies either $V_b = y$ was output on a forward query (x, b) or $V_b = x$ was output on a backward query (y, b). More importantly, for all $x \in \mathscr{D}$, the relationship $\mathscr{L}_0(x) \oplus \mathscr{L}_1(x) = \$(x)$ is always satisfied.

Now, we describe how the simulator works via the algorithms $\mathsf{SIM}_{\mathsf{FWD}}$ and $\mathsf{SIM}_{\mathsf{BCK}}$. Details of the these algorithms are given in Figs. 2 and 3. In the following, we assume that $\mathscr{A}$ always makes fresh queries since otherwise the simulator can repeat the previous responses (as it maintains internal states keeping all previous queries and responses).

Algorithm $\mathsf{SIM}_{\mathsf{FWD}}$ (see Fig. 2). On an input $(x \in \{0,1\}^n, b \in \{0,1\})$, S queries $\$$ to obtain $Z = \$(x)$. Then, S samples V_b randomly from the set $\{0,1\}^n \setminus \{\mathscr{R}_b \cup \{Z \oplus \mathscr{R}_{1-b}\}\}$, where $Z \oplus \mathscr{R}_{1-b}$ denotes the set $\{Z \oplus y | y \in \mathscr{R}_{1-b}\}$. Here, it can

Algorithm: $\mathsf{SIM}_{\mathsf{FWD}}$
Input: $x \in \{0,1\}^n, b \in \{0,1\}$
Output: $V_b \in \{0,1\}^n$

```
1 :      // check if the query on x is already answered
2 :  if x ∈ D
3 :     return L_b(x)
4 :      // otherwise execute the main part
5 :  Z ← $(x)
6 :  V_b ←$ {0,1}^n \ {R_b ∪ {Z ⊕ R_{1−b}}}
7 :      // update the state
8 :  R_b ← R_b ∪ {V_b}, R_{1−b} ← R_{1−b} ∪ {Z ⊕ V_b}
9 :  D ← D ∪ {x}
10 :  L_b(x) ← V_b, L_{1−b}(x) ← Z ⊕ V_b
11 :     // return the output
12 :  return V_b
```

Fig. 2. Description of the simulator for all forward queries.

be observed that the set $\{0,1\}^n \setminus \{\mathscr{R}_b \cup \{Z \oplus \mathscr{R}_{1-b}\}\}$ is non-empty, provided $q < 2^{n-1}$. Therefore, for $q < 2^{n-1}$, the sampling is always possible. Subsequently, S sets V_b and $Z \oplus V_b$ as outputs of $\mathsf{SIM}_{\mathsf{FWD}}(x,b)$ and $\mathsf{SIM}_{\mathsf{FWD}}(x,1-b)$ respectively (and hence $\mathsf{SIM}_{\mathsf{FWD}}(x,0) \oplus \mathsf{SIM}_{\mathsf{FWD}}(x,1) = \$(x)$). Before S returns V_b to the adversary, it updates all internal sets accordingly.

Algorithm $\mathsf{SIM}_{\mathsf{BCK}}$ (see Fig. 3). Next, we present the algorithm $\mathsf{SIM}_{\mathsf{BCK}}$. On an input $(y \in \{0,1\}^n, b \in \{0,1\})$, S samples an element V_b randomly from outside the set $\mathscr{D}$ and then obtains $\$(V_b)$ by querying $\$$. Now, there is a certain chance that $\$(V_b) \oplus y$ is in the range set $\mathscr{R}_{1-b}$, which would then violate the permutation property of Π_{1-b} that S is simulating. So, S continue with similar attempts until it samples a V_b such that $\$(V_b) \oplus y \notin \mathscr{R}_{1-b}$. It makes at most n such attempts. If it fails after all these n attempts, it returns $\perp$. In all these attempts, the S maintains an auxiliary set $\mathscr{D}'$ which is not a part of its state and only used locally during an execution. At the beginning of the algorithm, $\mathscr{D}'$ is initialized to the current domain $\mathscr{D}$. At the start of each iteration, a fresh V_b is sampled from the set $\{0,1\}^n \setminus \mathscr{D}'$. If the conditions $y \oplus \$(V_b) \in \mathscr{R}_{1-b}$ is satisfied (*i.e.*, the sampled V_b turns out to be a bad choice), then V_b is appended to $\mathscr{D}'$ and the next iteration begins. Note that if $q + n < 2^n$ then the set $\{0,1\}^n \setminus \mathscr{D}'$ is always non-empty so that the sampling of V_b (in Step 6) is possible in every iteration. But $q + n < 2^n$ is trivially satisfied for $n \geq 3$ and $q < 2^{n-1}$. When the condition is not satisfied (*i.e.,* when $y \oplus \$(V_b) \notin \mathscr{R}_{1-b}$) then S returns V_b after appropriately updating all the internal sets.

Remark 2. Here, as an aside, one may notice that there is a chance of collision due to two backward queries made to the two random permutations in the real world or two interfaces in the ideal world. We explain this with the following example. Assume that $\mathscr{A}$ makes backward queries $(y,0)$ and $(y',1)$ in the real world. Then it is easy to see that there is a positive probability of getting the same output in both the cases (as Π_0 and Π_1 are sampled independently from the set Perm). On the other hand, in the ideal world, when $\mathscr{A}$ makes the query $(y,0)$, then if $y \oplus \$(V_0) \notin \mathscr{R}_1$ (which has positive probability) then at Step 13 of $\mathsf{SIM}_{\mathsf{BCK}}$ $\mathscr{L}_1(V_0)$ is set to $\$(V_0) \oplus y$ and V_0 is returned to $\mathscr{A}$. Now, if $\mathscr{A}$ makes the query $(\$(V_0) \oplus y, 1)$ then due to the check at Step 2, V_0 is again returned to $\mathscr{A}$. Therefore, there is a positive probability of collision for the queries $(y,0)$ and $(y',1)$ in the ideal world as well (as was to be expected since the simulator is simulating the permutations Π_0 and Π_1), where $y' = \$(V_0) \oplus y$ in this case.

3.2 Additional Information to the Adversary

After the adversary $\mathscr{A}$ has finished its interaction in the real/ideal world, *i.e.*, when it has made q queries and received corresponding replies, it is provided with the following additional information. Note that the additional information does not degrade $\mathscr{A}$'s advantage as it is always possible to discard it. Below we assume x, x_i, y are from $\{0,1\}^n$ and b is from $\{0,1\}$.

Algorithm: $\mathsf{SIM}_{\mathsf{BCK}}$
Input: $y \in \{0,1\}^n, b \in \{0,1\}$
Output: $V_b \in \{0,1\}^n \cup \{\perp\}$

// check **if** the query on y is already answered
if $y = \mathscr{L}_b(x)$ **for** $x \in \mathscr{D}$
  return x
$\mathscr{D}' \leftarrow \mathscr{D}$
repeat n times
  $V_b \leftarrow_{\$} \{0,1\}^n \setminus \mathscr{D}'$
  $Z \leftarrow \$(V_b)$
  // check **if** $Z \oplus y$ is outside of the set $\mathscr{R}_{1-b}$
  if $Z \oplus y \notin \mathscr{R}_{1-b}$
    // update the state
    $\mathscr{D} \leftarrow \mathscr{D} \cup \{V_b\}$
    $\mathscr{R}_b \leftarrow \mathscr{R}_b \cup \{y\}, \mathscr{R}_{1-b} \leftarrow \mathscr{R}_{1-b} \cup \{Z \oplus y\}$
    $\mathscr{L}_b(V_b) \leftarrow y, \mathscr{L}_{1-b}(V_b) \leftarrow Z \oplus y$
    // **return** the inverse
    return V_b
  // otherwise append V_b **to** the set $\mathscr{D}'$ sothat it is not sampled **in** the next iteration
  $\mathscr{D}' \leftarrow \mathscr{D}' \cup \{V_b\}$
    // abort after t iterations
return $\perp$

Fig. 3. Description of the simulator for all backward queries.

1. For each query x made to the construction XORP, $\mathscr{A}$ is given the values $\Pi_0(x)$ and $\Pi_1(x)$. Similarly, for each query x made to the random function \$, $\mathscr{A}$ is given the outputs of the simulator S corresponding to the forward queries $(x,0)$ and $(x,1)$.
2. For each forward query (x,b) made to Π_{b} (*i.e.*, for each value of $\Pi_{\mathsf{b}}(x)$), it is also given $\Pi_{1-\mathsf{b}}(x)$. Similarly, for each forward query (x,b) made to S, $\mathscr{A}$ is also given the value corresponding to the forward query $(x, 1-b)$.
3. For each backward query (y,b) made to Π_{b} (*i.e.*, for each value of $\Pi_{\mathsf{b}}^{-1}(y)$), it is also given $\Pi_{1-\mathsf{b}}(\Pi_{\mathsf{b}}^{-1}(y))$. For each backward query (y,b) made to S, $\mathscr{A}$ is also given the value corresponding to the forward query $(x, 1-b)$, where x is the value returned by S on the backward query (y,b).

With access to this extra information, $\mathscr{A}$ knows the tuple $(x_i, \Pi_0(x_i), \Pi_1(x_i))$ corresponding to its i-th query in the real world. Note that from $\Pi_0(x_i)$ and $\Pi_1(x_i)$, $\mathscr{A}$ can always obtain $\Pi_0(x_i) \oplus \Pi_1(x_i)$ (which is, in fact, the output of XORP when queried with x_i). Therefore, we do not include this redundant

information in the tuple. When $\Pi_0(x_i)$ and $\Pi_1(x_i)$ are treated as random variables, we will denote $\Pi_0(x_i)$ by $U_{0,i}$ and $\Pi_1(x_i)$ by $U_{1,i}$. So, the tuple $(x_i, U_{0,i}, U_{1,i})$ is a random variable and an arbitrary but fixed value of this random variable will be denoted by $(x_i, u_{0,i}, u_{1,i})$. Similarly, in the ideal world, corresponding to the i-th query, $\mathscr{A}$ knows the tuple $(x_i, V_{0,i}, V_{1,i})$, where for $b \in \{0,1\}$, $V_{b,i}$ is the reply of S to the forward query (x_i, b). Similar to the previous case, we will denote a fixed value of the random variable $(x_i, V_{0,i}, V_{1,i})$ by $(x_i, v_{0,i}, v_{1,i})$. In the case where the backward query resulted in an abort, *i.e.*, the output was $\perp$, we take $x_i = \perp$ and $v_{0,i}$ and $v_{1,i}$ can be arbitrary (but fixed). In fact, in this case, $v_{0,i}$ and $v_{1,i}$ are purely included to maintain uniformity of presentation and will be disregarded in subsequent calculations. Further, slightly abusing the notation for its simplicity, we will denote any such tuple (*i.e.*, a tuple with $x_i = \perp$) by $\perp$. Note that we did not include the query type (*i.e.*, forward or backward) information in the tuple as, in our calculation, we will consider both the possibilities for a tuple. However, for the sake of completeness, one can assume that $\mathscr{A}$ has this information.

Without loss of any generality we will assume that $\mathscr{A}$ does not repeat its queries as the response will be the same for a repeated query. Also, we will discard any duplicate copy of a tuple that may have occurred due to the extra information supplied to $\mathscr{A}$[5].

(Extended) transcript of the adversary. In the real world, the sequence of random variables $(x_i, U_{0,i}, U_{1,i})$, with $1 \le i \le q$, is supported on the set $\mathscr{T}_u$ of sequences $(x_i, u_{0,i}, u_{1,i}), 1 \le i \le q$, $x_i, u_{0,i}, u_{1,i} \in \{0,1\}^n$ and $x_i \neq x_j, u_{0,i} \neq u_{0,j}, u_{1,i} \neq u_{1,j}$ for $1 \le i < j \le q$. Whereas in the ideal world the sequence of random variables $(x_i, V_{0,i}, V_{1,i})$, with $1 \le i \le q$, is supported on the set $\mathscr{T}_v$ of sequences $(x_i, v_{0,i}, v_{1,i}), 1 \le i \le q$, $x_i \in \{0,1\}^n \cup \{\perp\}$, $v_{0,i}, v_{1,i} \in \{0,1\}^n$ and $x_i \neq x_j, v_{0,i} \neq v_{0,j}, v_{1,i} \neq v_{1,j}$ for each $1 \le i < j \le q$ such that $x_i \neq \perp \neq x_j$. So, we have $\mathscr{T}_u \subset \mathscr{T}_v$. We term elements of $\mathscr{T}_u$ and $\mathscr{T}_v$ *views* of the adversary. In our subsequent treatment, we will solely work with the views from the real and the ideal world, and the fact that $\mathscr{T}_u \subset \mathscr{T}_v$ will be essential for the application of the χ^2 method.

4 Main Result

In this section, we state and prove our main result. We continue in the setup of the previous section. To simplify the presentation we denote 2^n by N. Our main result is the following.

Theorem 2. *Let $N \ge 16$ and $q < \frac{N}{2}$. Then*

$$\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(q) \le \sqrt{\frac{1.25q}{N}}$$

[5] For example, such a duplicate of a tuple $(x_i, u_{0,i}, u_{1,i})$ can occur in the real world if $\mathscr{A}$ queries the XORP with x_i and later makes a backward query to Π_0 with $u_{0,i}$.

Proof. Before presenting the technical details we will provide a brief outline of the proof to help the reader follow the underlying idea and the flow of the proof. But before that we will describe our notational setup for the proof.

Notational setup: To simplify the notation we will denote the random variable $(x_i, U_{0,i}, U_{1,i})$ by S_i and $(x_i, V_{0,i}, V_{1,i})$ by T_i. So, S_i (resp. T_i) follows the distribution of the real (resp. ideal) world which we denote by $\mathsf{p}_0^{\mathsf{fwd}}(.)$ (resp $\mathsf{p}_1^{\mathsf{fwd}}$) when S_i (resp T_i) is a forward query and by $\mathsf{p}_0^{\mathsf{bck}}(.)$ (resp $\mathsf{p}_1^{\mathsf{bck}}$) when S_i (resp T_i) is a backward query. Hence, we denote $\mathbf{Pr}[S_i = s_i]$ by $\mathsf{p}_0^{\mathsf{fwd}}(s_i)$ and $\mathbf{Pr}[T_i = t_i]$ by $\mathsf{p}_1^{\mathsf{fwd}}(t_i)$ when S_i and T_i are forward queries and likewise for backward queries. Further, we will abuse the notation to denote the joint distribution of S^{i-1} by $\mathsf{p}_0^{\mathsf{fwd}}$ when S^{i-1} corresponds to $i-1$ forward queries and by $\mathsf{p}_0^{\mathsf{bck}}$ when S^{i-1} corresponds to $i-1$ backward queries. Moreover, for fixed s_i and s^{i-1}, we denote $\mathbf{Pr}[S_i = s_i \mid S_1 = s_1, \ldots, S_{i-1} = s_{i-1}]$ by $\mathsf{p}_0^{\mathsf{fwd}}(s_i \mid s^{i-1})$ when S_i corresponds to a forward query; likewise for the other cases.

Outline of the proof: The main tool we use in our proof is Theorem 1. Our goal is to evaluate the r.h.s. of (4). In doing so, we calculate $\mathbf{Ex}[\chi^2(S^{i-1})]$ over the real world distributions ($\mathsf{p}_0^{\mathsf{fwd}}$ and $\mathsf{p}_0^{\mathsf{bck}}$). More precisely, we consider the two cases; (*i*) when s_i is a forward query, and (*ii*) when s_i is a backward query. For the forward query case, we first calculate $\chi^2(s^{i-1})$ for fixed s^{i-1}, which is given by the sum of

$$\frac{(\mathsf{p}_0^{\mathsf{fwd}}(s_i \mid s^{i-1}) - \mathsf{p}_1^{\mathsf{fwd}}(s_i \mid s^{i-1}))^2}{\mathsf{p}_1^{\mathsf{fwd}}(s_i \mid s^{i-1})}$$

taken over all possible s_i given s^{i-1}. Here, we note that the support $\mathscr{T}_u$ of real world distributions ($\mathsf{p}_0^{\mathsf{fwd}}$ and $\mathsf{p}_0^{\mathsf{bck}}$) is included in the supports $\mathscr{T}_u$ and $\mathscr{T}_v$ of the ideal world distributions $\mathsf{p}_0^{\mathsf{fwd}}$ and $\mathsf{p}_0^{\mathsf{bck}}$ respectively. Hence, $\chi^2(s^{i-1})$ is well defined. Next, we consider the random variable S^{i-1} in the real world. Each $S_j \in \{S^{i-1}\}$ may correspond to a forward query or a backward query. However, since the distributions $\mathsf{p}_0^{\mathsf{fwd}}$ and $\mathsf{p}_0^{\mathsf{bck}}$ are identical, the distribution of S^{i-1} does not depend on the query type of each individual S_j. So, we treat $\chi^2(S^{i-1})$ as a random variable and take its expectation under the distribution of S^{i-1}. Finally, we take the sum of $\mathbf{Ex}[\chi^2(S^{i-1})]$ for all i in the range $1 \le i \le q$, which turns out to be $\frac{8q^3}{N^3}$.

Corresponding steps for the backward query case are exactly similar to the forward query case when $s_i \neq \perp$. The case when $s_i = \perp$ is treated separately. Summing the expectations $\mathbf{Ex}[\chi^2(S^{i-1})]$ for the two subcases (*i.e.*, for $s_i \neq \perp$ and $s_i = \perp$) we obtain the final sum (taken over all i in the range $1 \le i \le q$) for the backward query case to be $\frac{2.5q}{N}$. Finally, we get the upper bound on $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(q)$ by applying Theorem 1, where we get an upper bound on the r.h.s. of (4) by taking the maximum of the forward and backward queries for all the q queries (which in this case turns out to be the backward query).

Technical details: Following the above discussion we first consider the case when s_i is a forward query, and then consider the case when it is a backward query. To simplify notation, from here on, we will denote $i-1$ by r.

Forward query

First, we calculate $\mathsf{p}_0^{\mathsf{fwd}}(s_i \mid s^r)$ and $\mathsf{p}_1^{\mathsf{fwd}}(s_i \mid s^r)$ for fixed s_i and s^r, where $s_i = (x_i, u_{0,i}, u_{1,i})$. $\mathsf{p}_0^{\mathsf{fwd}}(s_i \mid s^r)$ is straightforward to calculate. Since $x_i \notin \{x^r\}$, $\Pi_0(x_i)$ and $\Pi_1(x_i)$ are two independent random samples drawn from outside the sets $\{u_0^r\}$ and $\{u_1^r\}$ respectively. Thus

$$\begin{aligned}\mathsf{p}_0^{\mathsf{fwd}}(s_i \mid s^r) &= \mathbf{Pr}[S_i = (x_i, u_{0,i}, u_{1,i}) \mid S_1 = (x_1, u_{0,1}, u_{1,1}), \ldots, \\ &\qquad S_r = (x_r, u_{0,r}, u_{1,r})] \\ &= \mathbf{Pr}[\Pi_0(x_i) = u_{0,i} \wedge \Pi_1(x_i) = u_{1,i} \mid \Pi_0(x_j) = u_{0,j} \wedge \\ &\qquad \Pi_1(x_j) = u_{1,j} \forall 1 \le j \le r] \\ &= \frac{1}{(N-r)^2} \end{aligned} \tag{5}$$

To calculate $\mathsf{p}_1^{\mathsf{fwd}}(s_i \mid s^r)$, we consider, without loss of any generality, the execution of the algorithm $\mathsf{SIM}_{\mathsf{FWD}}$ algorithm on the forward query $(v_{0,i}, 0)$ (the case when the forward query is $(v_{1,i}, 1)$ is identical). In this case, $\mathscr{D} = \{x^r\}, \mathscr{R}_0 = \{u_0^r\}, \mathscr{R}_1 = \{u_1^r\}$. Then we have

$$\begin{aligned}\mathsf{p}_1^{\mathsf{fwd}}(s_i \mid s^r) &= \mathbf{Pr}[T_i = (x_i, v_{0,i}, v_{1,i}) \mid T_1 = (x_1, v_{0,1}, v_{1,1}), \ldots, T_r = (x_r, v_{0,r}, v_{1,r})] \\ &= \mathbf{Pr}[\$(x_i) = v_{0,i} \oplus v_{1,i} \wedge V_0 = v_{0,i} \mid \mathscr{D} = \{x^r\}, \mathscr{R}_0 = \{u_0^r\}, \mathscr{R}_1 = \{u_1^r\}] \\ &= \frac{1}{N} \times \frac{1}{N - |\mathscr{W}_{x_i}|}, \end{aligned} \tag{6}$$

where $\mathscr{W}_{x_i} = \mathscr{R}_0 \cup \{\$(x_i) \oplus \mathscr{R}_1\}$. From (5) and (6) we derive the expression for $\chi^2(s^r)$ below.

$$\begin{aligned}\chi^2(s^r) &= \sum_{s_i} \frac{(\mathsf{p}_0^{\mathsf{fwd}}(s_i|s^r) - \mathsf{p}_1^{\mathsf{fwd}}(s_i|s^r))^2}{\mathsf{p}_1^{\mathsf{fwd}}(s_i|s^r)} \\ &= \sum_{s_i} \frac{\left(\frac{1}{(N-r)^2} - \frac{1}{N(N-|\mathscr{W}_{x_i}|)}\right)^2}{\frac{1}{N(N-|\mathscr{W}_{x_i}|)}} \\ &= \sum_{s_i} \frac{N\left(|\mathscr{W}_{x_i}| - \frac{2rN-r^2}{N}\right)^2}{(N-|\mathscr{W}_{x_i}|)(N-r)^4}. \end{aligned} \tag{7}$$

The sum in (7) is over all possible s_i's given s^r. The number of such number of such s_i's is $(N-r)(N-|\mathscr{W}_{x_i}|)$. Therefore,

$$\chi^2(s^r) = \frac{N\left(|\mathscr{W}_{x_i}| - \frac{2rN-r^2}{N}\right)^2}{(N-r)^3} \tag{8}$$

Let S^r be chosen according to the distribution $\mathsf{p}_0^{\mathsf{fwd}}$. Then $\mathscr{D}, \mathscr{R}_0, \mathscr{R}_1$ are random variables. This, in turn, means $\mathscr{W}_{x_i}$ and $\chi^2(S^r)$ are also random variables (as

functions of $\mathscr{D}, \mathscr{R}_0, \mathscr{R}_1$). Our goal is to calculate the expectation of $\chi^2(S^r)$ under the distribution $\mathsf{p}_0^{\mathsf{fwd}}$. For notational simplicity, we denote the random variable $|\mathscr{W}_{x_i}|$ by $\mathtt{W}$ (mildly violating our notational convention). So, from (8) we have

$$\begin{aligned}\mathbf{Ex}[\chi^2(Z^r)] &= \mathbf{Ex}\left[\frac{N\left(\mathtt{W} - \frac{2rN-r^2}{N}\right)^2}{(N-r)^3}\right] \\ &= \frac{N}{(N-r)^3} \times \mathbf{Ex}\left[\left(\mathtt{W} - \frac{2rN-r^2}{N}\right)^2\right]. \end{aligned} \tag{9}$$

In the next lemma, whose proof is postponed to Sect. 5, we calculate $\mathbf{Ex}[\mathtt{W}]$.

Lemma 1. *With the above notation*

$$\mathbf{Ex}[\mathtt{W}] = \frac{2rN-r^2}{N}, \text{ and } \mathbf{Var}[\mathtt{W}] \le \frac{r^2}{N}.$$

Using Lemma 1, (9) can be written as

$$\begin{aligned}\mathbf{Ex}[\chi^2(S^r)] &= \frac{N}{(N-r)^3} \times \mathbf{Ex}\left[(\mathtt{W} - \mathbf{Ex}[\mathtt{W}])^2\right] \\ &= \frac{N}{(N-r)^3} \times \mathbf{Var}[\mathtt{W}].\end{aligned}$$

In Lemma 1, we also showed that $\mathbf{Var}[\mathtt{W}] \le \frac{r^2}{N}$. This leads to the following final expression for the forward query case.

$$\begin{aligned}\sum_{i=1}^{q} \mathbf{Ex}[\chi^2(S^r)] &\le \sum_{i=1}^{q} \frac{r^2}{(N-r)^3} \\ &\le \frac{8q^3}{N^3}. \end{aligned} \tag{10}$$

In (10), we used the fact $r < q$ and $q < \frac{N}{2}$.

Backward query

Let $\mathscr{Z}$ be the set of all possible s_i's which are not 'abort', *i.e.*, $s_i \neq \perp$. Then for backward queries we have the following split.

$$\begin{aligned}\mathbf{Ex}[\chi^2(S^r)] = \mathbf{Ex}&\left[\sum_{s_i \in \mathscr{Z}} \frac{(\mathsf{p}_0^{\mathsf{bck}}(s_i \mid S^r) - \mathsf{p}_1^{\mathsf{bck}}(s_i \mid S^r))^2}{\mathsf{p}_1^{\mathsf{bck}}(s_i \mid S^r)}\right] \\ &+ \mathbf{Ex}\left[\frac{(\mathsf{p}_0^{\mathsf{bck}}(\perp \mid S^r) - \mathsf{p}_1^{\mathsf{bck}}(\perp \mid S^r))^2}{\mathsf{p}_1^{\mathsf{bck}}(\perp \mid S^r)}\right] \end{aligned} \tag{11}$$

We evaluate the two expectations on the r.h.s. of (11) in the following two cases.

Case 1: $s_i \in \mathscr{X}$

In this case, we have for fixed s^r,

$$\mathsf{p}_0^{\mathsf{bck}}(s_i \mid s^r) = \mathsf{p}_0^{\mathsf{fwd}}(s_i \mid s^r) = \frac{1}{(N-r)^2}.$$

Next, we calculate $\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)$. For this, we need to consider the execution of the algorithm $\mathsf{SIM}_{\mathsf{BCK}}$. Let the backward query, without loss of any generality, be $(v_{0,i}, 0)$. Further, let us denote by V_0^j the V_0 sampled by the algorithm $\mathsf{SIM}_{\mathsf{BCK}}$ at the j-th iteration, where by j-th iteration we mean j-th repeated execution of the steps 6 to 19 of $\mathsf{SIM}_{\mathsf{BCK}}$. Let us assume that $\mathsf{SIM}_{\mathsf{BCK}}$ succeeds at the ℓ-th iteration for $1 \le \ell \le n$, *i.e.*, for $1 \le j \le \ell - 1$, $\$(V_0^j) \oplus v_{0,i} \in \mathscr{R}_1$, and $V_0^\ell = x_i$, where $\mathscr{R}_1 = \{v_1^r\}$ (also $\mathscr{R}_0 = \{v_0^r\}, \mathscr{D} = \{x^r\}$). Let us denote by $\mathsf{BAD}_{\ell-1}$ the event $\$(V_0^1) \oplus v_{0,i}, \ldots, \$(V_0^{\ell-1}) \oplus v_{0,i} \in \mathscr{R}_1$ and by E the event $\mathscr{D} = \{x^r\} \wedge \mathscr{R}_0 = \{v_0^r\} \wedge \mathscr{R}_1 = \{v_1^r\}$. Then

$$\begin{aligned}
\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r) &= \mathbf{Pr}[T_i = (x_i, v_{0,i}, v_{1,i}) \mid T_r = (x_r, v_{0,r}, v_{1,r}), \ldots, T_1 = (x_1, v_{0,1}, v_{1,1})] \\
&= \sum_{\ell=1}^{n} \mathbf{Pr}[\mathsf{BAD}_{\ell-1} \wedge V_0^\ell = x_i \wedge \$(x_i) = v_{0,i} \oplus v_{1,i} \mid \mathsf{E}] \\
&= \sum_{\ell=1}^{n} \mathbf{Pr}[V_0^\ell = x_i \wedge \$(x_i) = v_{0,i} \oplus v_{1,i} \mid \mathsf{BAD}_{\ell-1}, \mathsf{E}] \times \mathbf{Pr}[\mathsf{BAD}_{\ell-1} \mid \mathsf{E}]
\end{aligned}$$

Now, $\mathbf{Pr}[\mathsf{BAD}_{\ell-1} \mid \mathsf{E}]$ can be calculated as

$$\begin{aligned}
\mathbf{Pr}[\mathsf{BAD}_{\ell-1} \mid \mathsf{E}] &= \prod_{j=1}^{\ell-1} \mathbf{Pr}[\$(V_0^j) \oplus v_{0,i} \in \mathscr{R}_1 = \{v_1^r\} \mid \mathsf{E}] \\
&= \left(\frac{r}{N}\right)^{\ell-1}. \qquad (12)
\end{aligned}$$

To justify (12) we first note that the distribution $\mathsf{p}_0^{\mathsf{bck}}(.)$ is supported on the set of tuples $s^r = (s_1, \ldots, s_r)$ such that none of the s_j, with $1 \le j \le r$, is $\perp$. So, in the $\mathsf{SIM}_{\mathsf{BCK}}$ algorithm the set $\mathscr{R}_1$ has size r. Also, at the j-th iteration, with $1 \le j \le \ell - 1$ a fresh V_0^j (sampled from outside the set $\mathscr{D}'$) is given to \$. Therefore, $\mathsf{BAD}_{\ell-1}$ occurs when $\ell - 1$ independent events each with probability $\frac{r}{N}$ occur, leading to the expression in (12).

Next, at the ℓ-th iteration the set $\mathscr{D}'$ has size $r + \ell - 1$. Since V_0^ℓ is sampled at random from the set $\{0,1\}^n \setminus \mathscr{D}'$, we immediately have

$$\mathbf{Pr}[V_0^\ell = x_i \wedge \$(x_i) = v_{0,i} \oplus v_{1,i} \mid \mathsf{BAD}_{\ell-1}, \mathsf{E}] = \frac{1}{N} \times \frac{1}{N - r - \ell + 1}. \qquad (13)$$

By combining (12) and (13) we get

$$\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r) = \sum_{\ell=1}^{n} \frac{1}{N} \times \frac{1}{N - r - \ell + 1} \times \left(\frac{r}{N}\right)^{\ell-1}. \qquad (14)$$

In the following lemma, we derive a lower and an upper bound on $\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)$. Proof of the lemma is given in Sect. 5.

Lemma 2. *With the above notation, the following bounds hold for* $\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)$.

$$\frac{1}{(N-r)^2} \times \left(1 - \left(\frac{r}{N}\right)^n\right) \leq \mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r) \leq \frac{4}{N(N-r)}.$$

Let us denote the lower and upper bounds in Lemma 2 by $\mathtt{L}$ and $\mathtt{U}$ respectively. Then

$$\frac{\left(\mathsf{p}_0^{\mathsf{bck}}(s_i \mid s^r) - \mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)\right)^2}{\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)} \leq \max\left\{\frac{\left(\mathtt{U} - \frac{1}{(N-r)^2}\right)^2}{\mathtt{U}}, \frac{\left(\mathtt{L} - \frac{1}{(N-r)^2}\right)^2}{\mathtt{L}}\right\}. \tag{15}$$

(15) is justified because the function $\frac{\left(y - \frac{1}{(N-r)^2}\right)^2}{y}$ attains its minimum ($= 0$) at $y = \frac{1}{(N-r)^2}$ and is strictly increasing for $y \geq \frac{1}{(N-r)^2}$ and strictly decreasing for $y \leq \frac{1}{(N-r)^2}$. Now,

$$\frac{\left(\mathtt{U} - \frac{1}{(N-r)^2}\right)^2}{\mathtt{U}} = \frac{3N-4r}{4N(N-r)^3},$$

and

$$\frac{\left(\mathtt{L} - \frac{1}{(N-r)^2}\right)^2}{\mathtt{L}} = \frac{\left(\frac{r}{N}\right)^{2n}}{(N-r)^2 \times \left(1 - \left(\frac{r}{N}\right)^n\right)}.$$

Further, considering that $|\mathscr{Z}|$ is at most $(N - |\mathscr{D}|)(N - |\mathscr{R}_1|) = (N-r)^2$, we get

$$\sum_{s_i \in \mathscr{Z}} \frac{\left(\mathsf{p}_0^{\mathsf{bck}}(s_i \mid s^r) - \mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)\right)^2}{\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)} \leq \max\left\{\frac{3N-4r}{4N(N-r)}, \frac{\left(\frac{r}{N}\right)^{2n}}{\left(1 - \left(\frac{r}{N}\right)^n\right)}\right\}.$$

Therefore, when S^r is a random variable that follows the distribution $\mathsf{p}_0^{\mathsf{bck}}$, we obtain the following expectation under the distribution $\mathsf{p}_0^{\mathsf{bck}}$.

$$\mathbf{Ex}\left[\sum_{s_i \in \mathscr{Z}} \frac{\left(\mathsf{p}_0^{\mathsf{bck}}(s_i \mid S^r) - \mathsf{p}_1^{\mathsf{bck}}(s_i \mid S^r)\right)^2}{\mathsf{p}_1^{\mathsf{bck}}(s_i \mid S^r)}\right] \leq \max\left\{\frac{3N-4r}{4N(N-r)}, \frac{\left(\frac{r}{N}\right)^{2n}}{\left(1 - \left(\frac{r}{N}\right)^n\right)}\right\}. \tag{16}$$

Case 2: $s_i = \perp$

In the real world, there is no abort, so $\mathsf{p}_0^{\mathsf{bck}}(\perp \mid S^r) = 0$. Therefore, similar to (12),

$$\begin{aligned}\mathbf{Ex}\left[\frac{\left(\mathsf{p}_0^{\mathsf{bck}}(\perp \mid S^r) - \mathsf{p}_1^{\mathsf{bck}}(\perp \mid S^r)\right)^2}{\mathsf{p}_1^{\mathsf{bck}}(\perp \mid S^r)}\right] &= \mathbf{Ex}\left[\mathsf{p}_1^{\mathsf{bck}}(\perp \mid S^r)\right] \\ &= \mathsf{p}_1^{\mathsf{bck}}(\perp) \\ &= \left(\frac{r}{N}\right)^n. \end{aligned} \tag{17}$$

From (11), (16), and (17) we derive

$$\sum_{r=0}^{q-1} \mathbf{Ex}[\chi^2(S^r)] \le \sum_{r=0}^{q-1} \max\left\{\frac{3N-4r}{4N(N-r)}, \frac{\left(\frac{r}{N}\right)^{2n}}{\left(1-\left(\frac{r}{N}\right)^n\right)}\right\} + \left(\frac{r}{N}\right)^n$$
$$\le q \times \left(\max\left\{\frac{3}{4(N-q)}, \frac{\left(\frac{q}{N}\right)^{2n}}{\left(1-\left(\frac{q}{N}\right)^n\right)}\right\} + \left(\frac{q}{N}\right)^n\right).$$

For $q < \frac{N}{2}$ we have the following bounds,

$$\frac{3}{4(N-q)} < \frac{3}{2N}, \quad \frac{\left(\frac{q}{N}\right)^{2n}}{\left(1-\left(\frac{q}{N}\right)^n\right)} < \frac{1}{N(N-1)}, \text{ and } \left(\frac{q}{N}\right)^n < \frac{1}{N}.$$

Hence, we have for the backward query

$$\sum_{r=0}^{q-1} \mathbf{Ex}[\chi^2(S^r)] \le \frac{2.5q}{N}. \tag{18}$$

Finally, we get the following upper bound on $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(q)$.

$$\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(q) = \|S^q - T^q\| \tag{19}$$
$$\le \sqrt{\frac{1}{2}\sum_{r=0}^{q-1} \mathbf{Ex}[\chi^2(S^r)]} \tag{20}$$
$$\le \sqrt{\frac{1.25q}{N}}, \tag{21}$$

where (19) is from the definition of $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(q)$. (20) is given by (4) and (21) is given by the maximum of (10) and (18) (which is (18)) for the q queries. □

5 Auxiliary Proofs

In this section we state and prove Lemmas 1 and 2. We begin with Lemma 1 where we work with the same notation and setting of the Forward Query part of the proof of Theorem 2.

5.1 Proof of Lemma 1

Lemma 1. *With the notation of Theorem 2,*

$$\mathbf{Ex}[\mathtt{W}] = \frac{2rN - r^2}{N}, \text{ and } \mathbf{Var}[\mathtt{W}] \le \frac{r^2}{N}.$$

Proof. When Z^r is chosen according to the distribution $\mathsf{p}_0^{\mathsf{fwd}}$ the sets $\{U_0^r\}$ and $\{U_1^r\}$ are two random subsets (sampled independently) of $\{0,1\}^n$ of cardinality r. Also, in keeping with the notation of Theorem 2, we assume x_i to be a fixed element of $\{0,1\}^n$. Now, for each $g \in \{0,1\}^n$, we define an indicator random variable I_g as follows.

$$I_g = \begin{cases} 1 \text{ if } g \in \{U_0^r\} \text{ and } g \oplus x_i \in \{U_1^r\} \\ 0 \text{ otherwise.} \end{cases}$$

Therefore,

$$\mathbf{Ex}[I_g] = \mathbf{Pr}[I_g = 1] = \mathbf{Pr}[g \in \{U_0^r\} \wedge g \oplus x_i \in \{U_1^r\}] = \frac{r}{N} \times \frac{r}{N} = \frac{r^2}{N^2}. \quad (22)$$

Also,

$$\mathtt{W} = 2r - \sum_{g \in \{0,1\}^n} I_g.$$

Thus,

$$\mathbf{Ex}[\mathtt{W}] = 2r - \mathbf{Ex}\left[\sum_{g \in \{0,1\}^n} I_g\right] = 2r - \sum_{g \in \{0,1\}^n} \mathbf{Ex}[I_g] = \frac{2rN - r^2}{N}.$$

Next, to calculate $\mathbf{Var}[\mathtt{W}]$ we use the following relationship.

$$\mathbf{Var}[\mathtt{W}] = \mathbf{Var}\left[\sum_{g \in \{0,1\}^n} I_g\right] = \sum_{g \in \{0,1\}^n} \mathbf{Var}[I_g] + \sum_{g \neq h \in \{0,1\}^n} \mathbf{Cov}[I_g, I_h].$$

$\mathbf{Var}[I_g]$ is straightforward to calculate from the defintion;

$$\begin{aligned} \mathbf{Var}[I_g] = \mathbf{Ex}[I_g^2] - \mathbf{Ex}[I_g]^2 &= \mathbf{Ex}[I_g](1 - \mathbf{Ex}[I_g]) \\ &= \frac{r^2}{N^2} \times \left(1 - \frac{r^2}{N^2}\right) \\ &< \frac{r^2}{N^2}. \end{aligned}$$

From the definition, $\mathbf{Cov}[I_g, I_h]$ is given by $\mathbf{Cov}[I_g, I_h] = \mathbf{Ex}[I_g I_h] - \mathbf{Ex}[I_g]\mathbf{Ex}[I_h]$. Since $\mathbf{Ex}[I_g] = \mathbf{Ex}[I_h] = \frac{r^2}{N^2}$ is given by (22), the task reduces to the calculation of $\mathbf{Ex}[I_g I_h]$ which we consider below.

$$\begin{aligned} \mathbf{Ex}[I_g I_h] &= \mathbf{Pr}[I_g = 1 \wedge I_h = 1] \\ &= \mathbf{Pr}[g \in \{U_0^r\} \wedge h \in \{U_0^r\} \wedge g \oplus x_i \in \{U_1^r\} \wedge h \oplus x_i \in \{U_1^r\}] \\ &= \mathbf{Pr}[g \in \{U_0^r\} \wedge h \in \{U_0^r\}] \times \mathbf{Pr}[g \oplus x_i \in \{U_1^r\} \wedge h \oplus x_i \in \{U_1^r\}] \\ &= \frac{r(r-1)}{N(N-1)} \times \frac{r(r-1)}{N(N-1)} \\ &= \left(\frac{r(r-1)}{N(N-1)}\right)^2 \end{aligned}$$

Therefore, $\mathbf{Cov}[I_g, I_h] = \left(\frac{r(r-1)}{N(N-1)}\right)^2 - \left(\frac{r^2}{N^2}\right)^2 < 0$. This implies that

$$\mathbf{Var}[\mathtt{W}] < N \times \frac{r^2}{N^2} = \frac{r^2}{N}.$$

This finishes the proof of the lemma. □

5.2 Proof of Lemma 2

Lemma 2. *With the notation of Theorem 2, the following bounds hold for* $\mathsf{p}_1^{\mathsf{bck}}(s_i \mid s^r)$.

$$\frac{1}{(N-r)^2} \times \left(1 - \left(\frac{r}{N}\right)^n\right) \le \mathsf{p}_1^{\mathsf{bck}}(z_i \mid z^r) \le \frac{4}{N(N-r)}.$$

Proof. The lower bound is justified as follows.

$$\begin{aligned}
\mathsf{p}_1^{\mathsf{bck}}(z_i \mid z^r) &= \sum_{\ell=1}^{n} \frac{1}{N} \times \frac{1}{N-r-\ell+1} \times \left(\frac{r}{N}\right)^{\ell-1} \\
&\ge \frac{1}{N(N-r)} \times \sum_{\ell=1}^{n} \left(\frac{r}{N}\right)^{\ell-1} \\
&= \frac{1}{N(N-r)} \times \frac{1-\left(\frac{r}{N}\right)^n}{1-\left(\frac{r}{N}\right)} \\
&= \frac{1}{(N-r)^2} \times \left(1 - \left(\frac{r}{N}\right)^n\right).
\end{aligned}$$

For the upper bound, we get

$$\begin{aligned}
\mathsf{p}_1^{\mathsf{bck}}(z_i \mid z^r) = \sum_{\ell=1}^{n} \frac{1}{N} \times \frac{1}{N-r-\ell+1} \times \left(\frac{r}{N}\right)^{\ell-1} &\le \frac{4}{N^2} \times \sum_{\ell=1}^{\infty} \left(\frac{r}{N}\right)^{\ell-1} \qquad (23) \\
&= \frac{4}{N(N-r)}.
\end{aligned}$$

The first term on the r.h.s. of (23) follows by noting that $r < q < \frac{N}{2}$ and $\ell < n = \log N \le \frac{N}{4}$, for $N \ge 16$. □

6 Extension to the Xor of k Permutations

In this section, we apply our main result (Theorem 2) to show full indifferentiable security of the $\mathsf{XORP}[k]$ construction for any k. Following Theorem 2, it is sufficient to consider $\mathsf{XORP}[k]$ with $k \ge 3$. In particular, our result is the following.

Theorem 3. *Let $N \geq 16$ and $q < \frac{N}{2}$. Then, there exists a simulator S' for $\mathsf{XORP}[k], k \geq 3$, such that for any adversary $\mathscr{A}'$, there exists an adversary $\mathscr{A}$ with*

$$\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP}[k],\$}(\mathscr{A}') = \mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(\mathscr{A})$$

and hence, $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP}[k],\$}(q) \leq \sqrt{\frac{1.25q}{N}}$.

Proof. The indifferentiable security analysis of $\mathsf{XORP}[k]$ follows a reduction technique which is similar to the technique used in [MP15] to prove PRF-security of $\mathsf{XORP}[k]$ in the indistinguishability setting. However, in our case, we additionally need to consider the simulator S'.

Brief description of S'. First, we recall the simulator S for XORP from Sect. 3. The simulator S' works almost the same way as S works. It first samples $(k-2)$ independent random permutations $\Pi_2, \ldots, \Pi_{k-1}$. Note that the sampling can be simulated as a *lazy sampling* in an efficient manner instead of sampling the whole permutations at a time.

In case of a forward or backward query (x, i), with $i \geq 2$, S' responds honestly (*i.e.,* it uses its own sampled random permutation as mentioned above). When $i \in \{0, 1\}$, it behaves exactly in the same way as S except that it computes $\$'(x) = \$(x) \oplus \Pi_2(x) \oplus \cdots \Pi_{k-1}(x)$ and then applies Step 5 of $\mathsf{SIM}_{\mathsf{FWD}}$ (see Fig. 2) in case of a forward query, or Step 7 of $\mathsf{SIM}_{\mathsf{BCK}}$ (see Fig. 3) in case of a backward query.

Next, we describe the reduction for the adversaries. Suppose there is an adversary $\mathscr{A}'$ against $\mathsf{XORP}[k]$ and consider the simulator S' defined above. Now, we construct an adversary $\mathscr{A}$ against XORP and the simulator S. The adversary $\mathscr{A}$ first stores the permutations $\Pi_2, \ldots, \Pi_{k-1}$ (again using lazy sampling to make those efficient). Next, $\mathscr{A}$ runs the algorithm $\mathscr{A}'$ which can make two types of queries, namely (a) primitive or simulator queries and (b) construction or random function queries. Below, we consider these two types of queries.

(a) In case of a primitive or simulator query (x, i) (either forward or backward), $\mathscr{A}$ first checks whether $i \geq 2$ or not. If $i \geq 2$, then $\mathscr{A}$ can simulate the response on its own, *i.e.*, it computes $\Pi_i(x)$ or $\Pi_i^{-1}(x)$, where $\Pi_i \in \{\Pi_2, \ldots, \Pi_{k-1}\}$, and sends the output back to $\mathscr{A}'$. If $i = 0$ or 1, then $\mathscr{A}$ forwards the query to its simulator/primitive oracle and whatever response it gets again forwards to $\mathscr{A}'$; so, it basically relays the queries and responses.

(b) In case of a construction or random function query, $\mathscr{A}$ forwards the query to its corresponding construction/random function oracle. Suppose $\mathscr{A}$ gets Z as a response. Then it computes $Z' = Z \oplus \bigoplus_{i=2}^{k-1} \Pi_i(x)$, and sends Z' back to $\mathscr{A}'$.

Note that $\mathscr{A}$ is actually interacting with $(\mathsf{XORP}, (\Pi_0, \Pi_1, \Pi_0^{-1}, \Pi_1^{-1}))$, whereas the interaction interface of $\mathscr{A}'$ is equivalent to

$$(\mathsf{XORP}[k], (\Pi_0, \ldots, \Pi_{k-1}, \Pi_0^{-1}, \ldots, \Pi_{k-1}^{-1})).$$

Now, assume that $\mathscr{A}$ is interacting with $(\$, \mathsf{S})$, the interaction interface of $\mathscr{A}'$ is then equivalent to $(\$ \oplus \mathsf{XORP}[k-2], \mathsf{S}')$. It is easy to see the correctness of the first oracle as $\mathscr{A}'$ xors his computation of $\mathsf{XORP}[k-2]$ with the output of \$. Similarly, one can show the simulator interface of $\mathscr{A}'$ is S'. Note that $\$ \oplus \mathsf{XORP}[k-2]$ is completely independent of $\mathsf{XORP}[k-2]$, and we can consider it as another independent random function $\$'$. Thus, the interface of $\mathscr{A}'$ is equivalent to $(\$', \mathsf{S}')$. So, $\mathscr{A}$ perfectly simulates the real and the ideal world of $\mathscr{A}'$. Therefore, $\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP}[k],\$}(\mathscr{A}') = \mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP},\$}(\mathscr{A})$. By Theorem 2, we finally have

$$\mathsf{Adv}^{\mathrm{diff}}_{\mathsf{XORP}[k],\$}(q) \leq \sqrt{\frac{1.25q}{N}}.$$

□

7 Conclusion

Proving full security of XORP construction in the secret or public permutation model (*i.e.,* indifferentiable security) is a challenging problem. Recently, Dai et al. introduced a method, called the χ^2 method, using which they were able to obtain full PRF-security of XORP in the secret random permutation model. The full security in the public permutation model for this construction was an open problem. In this paper, we apply the χ^2 method to the XORP construction to prove its full indifferentiable security. We believe this method can also be used for other cryptographic constructions for which the full security is not known.

Here, we also remark that though our bound shows full (*i.e.*, n-bit) indifferentiable security of XORP and $\mathsf{XORP}[k]$, in practice (*i.e.*, for realistic setting of parameters), it does not lead to full n-bit security (mainly due to the presence of the square root in the bound). As an immediate goal, it will be interesting to investigate if a more sophisticated application of the χ^2 method can get rid of the square root in our bound.

Acknowledgement. We are indebted to the reviewers for their patient reading and valuable comments which improved the quality of this paper significantly.

This work is supported in part by the WISEKEY project, which we gratefully acknowledge.

References

[AMP10] Andreeva, E., Mennink, B., Preneel, B.: On the indifferentiability of the Grøstl hash function. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 88–105. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15317-4_7

[BDP+13] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G., NIST, G.: Keccak and the SHA-3 Standardization (2013)

[BDPVA08] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the indifferentiability of the sponge construction. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 181–197. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_11

[BDPVA11a] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Duplexing the sponge: single-pass authenticated encryption and other applications. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 320–337. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28496-0_19

[BDPVA11b] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the security of the keyed sponge construction. In: Symmetric Key Encryption Workshop (SKEW 2011) (2011)

[BI99] Bellare, M., Impagliazzo, R.: A tool for obtaining tighter security analyses of pseudorandom function based constructions, with applications to PRP to PRF conversion. IACR Cryptol. ePrint Arch. **1999**, 24 (1999)

[BKR98] Bellare, M., Krovetz, T., Rogaway, P.: Luby-Rackoff backwards: Increasing security by making block ciphers non-invertible. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 266–280. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054132

[BMN10] Bhattacharyya, R., Mandal, A., Nandi, M.: Security analysis of the mode of JH hash function. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 168–191. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13858-4_10

[BN18] Bhattacharya, S., Nandi, M.: Revisiting variable output length pseudorandom functions. IACR Trans. Symmetric Cryptol. **2018**(1) (2018, to appear)

[CAE] CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness. http://competitions.cr.yp.to/caesar.html/

[CLP14] Cogliati, B., Lampe, R., Patarin, J.: The indistinguishability of the XOR of k permutations. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 285–302. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46706-0_15

[CS16a] Cogliati, B., Seurin, Y.: EWCDM: an efficient, beyond-birthday secure, nonce-misuse resistant MAC. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 121–149. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_5

[CT06] Cover, T.M., Thomas, J.A.: Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing), Wiley-Interscience (2006)

[DHT17] Dai, W., Hoang, V.T., Tessaro, S.: Information-theoretic indistinguishabilityvia the chi-squared method. In: Katz and Shacham [KS17], pp. 497–523 (2017)

[GG16] Gilboa, S., Gueron, S.: The Advantage of Truncated Permutations, CoRR abs/1610.02518 (2016)

[GGM17] Gilboa, S., Gueron, S., Morris, B.: How many queries are needed to distinguish a truncated random permutation from a random function? J. Cryptol. **31**(1), 162–171 (2017)

[GKM+09] Gauravaram, P., Knudsen, L.R., Matusiewicz, K., Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: Grøstl-a SHA-3 candidate. In: Dagstuhl Seminar Proceedings, Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2009)

[IMPS17] Iwata, T., Minematsu, K., Peyrin, T., Seurin, Y.: ZMAC: a fast tweakable block cipher mode for highly secure message authentication. IACR Cryptol. ePrint Arch. **2017**, 535 (2017)

[IMV16] Iwata, T., Mennink, B., Vizár, D.: CENC is optimally secure. IACR Cryptol. ePrint Arch. **2016**, 1087 (2016)

[Iwa06] Iwata, T.: New blockcipher modes of operation with beyond the birthday bound security. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 310–327. Springer, Heidelberg (2006). https://doi.org/10.1007/11799313_20

[KS17] Katz, J., Shacham, H. (eds.): CRYPTO 2017. LNCS, vol. 10403. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9

[Lee17] Lee, J.: Indifferentiability of the sum of random permutations towards optimal security. IEEE Trans. Inf. Theory **63**(6), 4050–4054 (2017)

[LR88] Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM J. Comput. **17**(2), 373–386 (1988)

[Luc00] Lucks, S.: The sum of PRPs Is a secure PRF. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 470–484. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_34

[LV87] Liese, F., Vajda, I.: Convex Statistical Distances. Teubner, Leipzig (1987)

[MN17a] Mennink, B., Neves, S.: Encrypted Davies-Meyer and its dual: Towards optimal security using Mirror theory, Cryptology ePrint Archive, Report 2017/xxx, to be published in CRYPTO 2017 (2017). http://eprint.iacr.org/2017/537

[MN17b] Mennink, B., Neves, S.: Encrypted davies-meyer and its dual: towards optimal security using mirror theory. In: Katz and Shacham [KS17], pp. 556–583 (2017)

[MP15] Mennink, B., Preneel, B.: On the XOR of multiple random permutations. In: Malkin, T., Kolesnikov, V., Lewko, A.B., Polychronakis, M. (eds.) ACNS 2015. LNCS, vol. 9092, pp. 619–634. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-28166-7_30

[MPN10] Mandal, A., Patarin, J., Nachef, V.: Indifferentiability beyond the birthday bound for the xor of two public random permutations. In: Gong, G., Gupta, K.C. (eds.) INDOCRYPT 2010. LNCS, vol. 6498, pp. 69–81. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17401-8_6

[MRH04] Maurer, U., Renner, R., Holenstein, C.: Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24638-1_2

[Pat08a] Patarin, J.: The "Coefficients H" technique. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 328–345. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04159-4_21

[Pat08b] Patarin, J.: A proof of security in $O(2^n)$ for the xor of two random permutations. In: Safavi-Naini, R. (ed.) ICITS 2008. LNCS, vol. 5155, pp. 232–248. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85093-9_22

[Pat10] Patarin, J.: Introduction to Mirror Theory: Analysis of Systems of Linear Equalities and Linear Non Equalities for Cryptography. Cryptology ePrint Archive, Report 2017/287 (2010). http://eprint.iacr.org/2010/287

[RAB+08] Rivest, R.L., Agre, B., Bailey, D.V., Crutchfield, C., Dodis, Y., Fleming, K.E., Khan, A., Krishnamurthy, J., Lin, Y., Reyzin, L., et al.: The MD6 hash function-a proposal to NIST for SHA-3. NIST **2**(3) (2008, submitted)

[Sta78] Stam, A.J.: Distance between sampling with and without replacement. Statistica Neerlandica **32**(2), 81–91 (1978)

[Vau03] Vaudenay, S.: Decorrelation: a theory for block cipher security. J. Cryptol. **16**(4), 249–286 (2003)

[Wu11] Wu, H.: The hash function JH, NIST (round 3), 6 (2011, submitted)

[Yas11] Yasuda, K.: A new variant of PMAC: beyond the birthday bound. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 596–609. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_34