Post-Quantum Security of Keyed Sum of Permutations and Its Siblings*

Nilanjan Datta[1,2], Avijit Dutta[1,2], Sougata Mandal[1,3], Hrithik Nandi[1,3] and Amlan Sinha[1,2]

[1] Institute for Advancing Intelligence (IAI), TCG CREST, Kolkata, India
[2] Academy of Scientific and Innovative Research (AcSIR), Ghaziabad, India
[3] Ramakrishna Mission Vivekananda Educational and Research Institute, Belur, India
{nilanjan.datta,avijit.dutta,sougata.mandal,hrithik.nandi.85,amlan.sinha.168}@tcgcrest.org

Abstract. The rapid advancement of quantum computing poses significant challenges to the security of existing cryptographic constructions. Several constructions that are provably secure in the classical setting, e.g., the 3-round Luby–Rackoff, Even–Mansour, Keyed Sum of Permutations, become vulnerable when the adversary is granted quantum oracle access (the Q2 model). In contrast, when the adversary is restricted to classical oracle queries while retaining the ability to perform quantum computations locally (the Q1 model), such attacks no longer apply. In this paper, we investigate the Q1 security of the Keyed Sum of Permutations construction and two closely related variants - one employing identical permutations and another using a single key. We prove that all three constructions achieve $n/3$-bit security in the Q1 model. In addition, for the same-key variant, we exhibit a key-recovery attack with matching complexity, thereby establishing the tightness of our security bound. For the remaining two constructions, we derive key-recovery attacks with complexity $2^{2n/3}$.

Keywords: Keyed Sum of Permutations · Post-Quantum Security · Q1 Model · Reprogramming Lemma · Resampling Lemma

1 Introduction

With the rapid progress of quantum cryptanalysis, assessing the security of cryptographic constructions under quantum attack models has become a central problem in modern cryptography. In the realm of public-key cryptography, quantum algorithms such as Shor's [15] demonstrate that integer factorization and discrete logarithms can be solved in polynomial time, thereby undermining the security foundations of most widely deployed asymmetric primitives. Consequently, schemes whose security relies on these hardness assumptions are no longer considered secure in the presence of quantum adversaries. Symmetric cryptography is also affected, albeit in a more nuanced manner. Grover's algorithm [10] accelerates brute-force search, including exhaustive key search, yielding a quadratic improvement over classical methods and effectively reducing the effective key strength by a factor of two. However, the impact of quantum computation extends beyond generic search speedups. For certain symmetric constructions, specialized quantum algorithms exploiting structural properties, e.g., period-finding techniques, enable significantly stronger attacks.

Two principal adversarial models are commonly considered: Q1 and Q2. In the Q2 model, the adversary is given full quantum access to both public components and the

*This article is the full version of the paper that appeared at ASIACRYPT 2026.

keyed construction, allowing it to issue superposition queries through quantum oracles and leverage powerful algorithms such as Simon's and Grover's. This strong capability often enables polynomial-time attacks against many classical symmetric schemes, rendering them insecure in this setting. However, the Q2 model assumes the existence of a quantum interface to secret-key implementations, an assumption that is generally unrealistic in practical deployments. In contrast, the Q1 model more accurately reflects real-world scenarios. In this model, the adversary may query public, key-less components in superposition but can interact with the keyed construction only through classical queries, since practical systems typically expose only classical interfaces. Although the adversary may possess a quantum computer, it cannot realize a quantum oracle for the unknown keyed instance. This asymmetric access results in a hybrid attack setting that combines quantum information obtained from public components with limited classical observations of the secret-key algorithm. Consequently, standard quantum algorithms cannot be applied directly, and security proofs must carefully address the intricate dependencies between the two types of queries. Establishing tight security bounds in the Q1 model is therefore essential both for evaluating the post-quantum security of deployed symmetric cryptosystems and for advancing foundational proof techniques in post-quantum cryptography.

Related Works. In recent years, the cryptographic community has made substantial progress in analyzing the post-quantum security of symmetric constructions in the Q1 adversarial model. Jaeger et al. [11] first proved a Q1 security lower bound for the FX construction, demonstrating that it achieves $(k+n)/3$-bit security against non-adaptive attacks. Their result matches the previously known attacks of Bonnetain et al. [7], thus confirming the tightness of the bound. In [2] Alagic et al. established that the Even-Mansour construction provides $n/3$-bit Q1 security under adaptive attacks, while the offline Simon's algorithm attack results on the Even-Mansour construction by Bonnetain et al. [7] show that this bound is tight. In [3], Alagic et al. further extended this line of work to the Tweakable Even-Mansour construction, showing that it also achieves $n/3$-bit security in the adaptive Q1 setting. In [4], Bai et al. conducted a systematic study of Key-Alternating Ciphers, establishing Q1 security guarantees for both 2-round and general t-round variants. In [5], Basak et al. proved that 3/4-round Key-Alternating Feistels (KAF) achieve tight $n/3$-bit PRP/SPRP security in the Q1 setting when the round functions are modeled as a random oracle (even if identical). Apart from the single-user setting, Q1 security in the multi-key setting is also being explored in the Q1 setting [13, 14] . Recently, Tan et al. [18] showed that sum of Even-Mansour SoEM21 construction achieves Q1 security up to $n/3$-bits, which matches the earlier attacks due to Li. et al. [12], thereby demonstrating the tightness of the bound.

Our Contributions. In this paper, we consider the Keyed Sum of Permutations (KSoP) [16], a random permutation based PRF that achieves the same provable security as the Sum of Even Mansour in the classical setting while being strictly simpler. Let $\mathsf{P}_1, \mathsf{P}_2 \in \mathcal{P}(n)$ be two independent random permutations. The Keyed Sum of Permutations (KSoP) construction is defined as:

$$\mathsf{KSoP}_{k_1,k_2}[\mathsf{P}_1, \mathsf{P}_2](x) \coloneqq \mathsf{P}_1(x \oplus k_1) \oplus \mathsf{P}_2(x \oplus k_2)\,.$$

The single-key variant of KSoP, dubbed as $\mathsf{KSoP21}$, is defined by

$$\mathsf{KSoP21}_k[\mathsf{P}_1, \mathsf{P}_2](x) \coloneqq \mathsf{P}_1(x \oplus k) \oplus \mathsf{P}_2(x \oplus k)\,.$$

The single-permutation variant of KSoP, dubbed as $\mathsf{KSoP12}$, is defined as

$$\mathsf{KSoP12}_{k_1,k_2}[\mathsf{P}](x) \coloneqq \mathsf{P}(x \oplus k_1) \oplus \mathsf{P}(x \oplus k_2)\,.$$

The pictorial representations of all the three constructions KSoP, $\mathsf{KSoP21}$, and $\mathsf{KSoP12}$ are depicted in Fig. 1, Fig. 2, and Fig. 3, respectively.

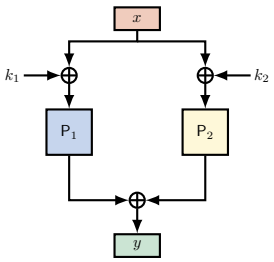


Figure 1: KSoP

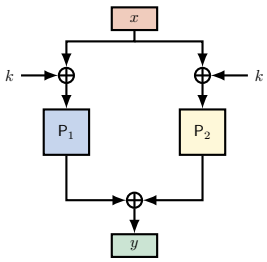


Figure 2: KSoP21

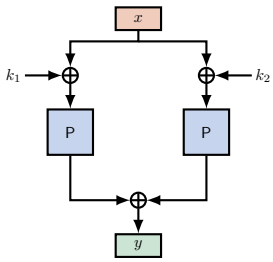


Figure 3: KSoP12

As our main contribution to the paper, we analyzed the post-quantum security of all the above mentioned three constructions and showed that all three constructions achieve $n/3$-bits security. To be precise, we demonstrate that for any adversary $\mathcal{A}$ that making q classical queries and total p quantum queries to its primitive oracle(s), the advantage of $\mathcal{A}$ is bounded by

$$O\left(\frac{q^2}{2^n} + \frac{p\sqrt{q} + q\sqrt{p}}{2^{n/2}}\right).$$

In addition, we have demonstrated a key-recovery attack for KSoP21 in $O(2^{n/3})$ data and time complexity, thereby obtaining tight post-quantum security for KSoP21. For KSoP and KSoP12, we show a key-recovery attack can be mounted in $O(2^{2n/3})$ data and time complexity.

Techniques. The techniques we employ are fairly generic and closely resemble those used in [18]. This suggests that classical security proofs for other similar (summation-type) cryptographic constructions, involving either a single permutation or multiple independent permutations, can likely be adapted to the Q1 setting using similar arguments.

Our proofs build upon the techniques developed by Alagic et al. [2] in their post-quantum analysis of the Even–Mansour construction, suitably adapted to our setting. At a high level, the argument proceeds via a sequence of hybrid experiments in which both the classical construction queries and the quantum primitive queries are gradually replaced by their ideal counterparts. A central technical requirement is to ensure consistency between these two types of queries throughout the hybrid transitions.

Concretely, our analysis relies on two main tools: the Arbitrary Reprogramming (ARP) lemma and the Adaptive Resampling (ARS) lemma. The primary technical challenge arises from handling resampling at multiple points of a single underlying permutation. To address this issue, we extend the ARS lemma to accommodate resampling at multiple points, thereby enabling us to control the distinguishing advantage in this more general setting.

Organization. In Section 2 and Section 3, we present the necessary preliminaries and technical tools used throughout our analysis. We then state and prove the formal security results for KSoP, KSoP21 and KSoP12 in Section 4 and Section 5, respectively (cf. Theorem 1, and Theorem 2). Additionally, in Section 6, we describe the key-recovery attacks on all three constructions. Finally, in Section 7, we discuss the security of the construction KSoPr1, a variant of KSoP with r-many permutations and a single key.

2 Preliminaries

NOTATION. For any natural number n, $\{0,1\}^n$ denotes the set of all bit strings of length n, and $\{0,1\}^*$ denotes the set of all binary strings of arbitrary finite length. $x \xleftarrow{\$} X$ denotes that x is sampled uniformly at random from X. For any $n \in \mathbb{N}$, let $\mathcal{F}(n,n)$ denote the set of all functions $f : \{0,1\}^n \to \{0,1\}^n$, and let $\mathcal{P}(n) \subseteq \mathcal{F}(n,n)$ denote the set of all permutations on $\{0,1\}^n$. Let $[n] = \{0,\ldots,n\}$, for any natural number n. We denote an empty set as $\emptyset$. For any $f \in \mathcal{F}(n,n)$, we define $\mathsf{Support}(f) := \{x\colon f(x) \neq 0\}$. For $X, Y \in \{0,1\}^n$, we write $X \oplus Y$ to denote the bitwise xor of X and Y. For a function $F : \{0,1\}^m \to \{0,1\}^n$ and a set $B \subset \{0,1\}^m \times \{0,1\}^n$ such that each $x \in \{0,1\}^m$ is the first element of at most one tuple in B, define

$$F^{(B)}(x) := \begin{cases} y, & \text{if } (x,y) \in B, \\ F(x), & \text{otherwise.} \end{cases}$$

For any $a, b, x \in \{0,1\}^n$, we define the function $\mathsf{swap}_{a,b}$ as follows:

$$\mathsf{swap}_{a,b}(x) = \begin{cases} a, & x = b, \\ b, & x = a, \\ x, & \text{otherwise.} \end{cases}$$

For any permutation $\mathsf{P} \in \mathcal{P}(n)$, and any two uniformly chosen points $s_0, s_1 \in \{0,1\}^n$, we denote the *reprogrammed* permutation of P w.r.t. (s_0, s_1) as $\mathsf{P}^{(1)}$, which is defined as $\mathsf{P}^{(1)} = \mathsf{P} \circ \mathsf{swap}_{s_0,s_1}$.

When an adversary has *quantum* access to a function $f\colon \{0,1\}^m \to \{0,1\}^n$, this means that the adversary accesses, as a black box, the $(m+n)$ qubit unitary operator $|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$.

Post-Quantum PRF Security. Let $\mathcal{F}(n,n)$ be the set of functions from $\{0,1\}^n$ to $\{0,1\}^n$. Let $\mathcal{A}^{\mathcal{O}}$ be an algorithm having access to an oracle $\mathcal{O}$ and making only *forward* queries $x \mapsto \mathcal{O}(x)$ to it, and outputting a bit. The *pseudorandom function* (PRF) advantage of $\mathcal{A}$ against a family of functions F_k, indexed by a secret key $k \in \{0,1\}^\kappa$, is defined as:

$$\mathbf{Adv}^{\mathrm{PRF}}_{\mathrm{F}}(\mathcal{A}) = \left| \Pr_{k \xleftarrow{\$} \{0,1\}^\kappa}\left[\mathcal{A}^{\mathrm{F}_k} \to 1\right] - \Pr_{\mathsf{F} \xleftarrow{\$} \mathcal{F}(n,n)}\left[\mathcal{A}^{\mathsf{F}} \to 1\right] \right|.$$

In addition, when $\mathcal{A}$ is a quantum algorithm (but access to $\mathcal{O}$ remains classical), we use the notation $\mathbf{Adv}^{\mathtt{pq\text{-}prf}}_{\mathrm{F}}(\mathcal{A})$. We consider PQ-PRF definition when F_k is a *construction*, built over a *primitive* f that is modeled as a random function (or a random permutation). In that case, the primitive is chosen uniformly at random before running the game, and $\mathcal{A}$ accesses it via a quantum oracle $|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$. However, access to the construction remains classical.

3 Technical Tools

In this section, we present the main technical tools required for the Q1 security proofs developed in this paper. Most of these techniques are drawn from existing literature, although in several cases we adapt and refine them to suit our specific setting. For completeness and clarity, we restate the relevant results in a self-contained manner. These tools will play a central role in establishing the security bounds proved in the subsequent sections.

3.1 The Arbitrary Reprogramming Game

We recall the *arbitrary reprogramming lemma*, first introduced in [2]. It considers a distinguisher that first fixes a base oracle, along with a randomized procedure that outputs a set of points at which the oracle will be reprogrammed. The distinguisher is then given quantum access either to the original oracle or to the oracle modified at those specified points. The ARP lemma upper-bounds the distinguisher's advantage in distinguishing between these two oracles. The adaptive reprogramming game for a distinguisher $\mathcal{D}$ is given as follows:

- **Phase 1:** $\mathcal{D}$ outputs descriptions of a function $F_0 = F : \{0,1\}^m \to \{0,1\}^n$ and a randomized algorithm $\mathcal{B}$ whose output is a set $B \subset \{0,1\}^m \times \{0,1\}^n$ where each $x \in \{0,1\}^m$ is the first element of at most one tuple in B. Let

$$B_1 = \{x \mid \exists y : (x,y) \in B\} \quad \text{and} \quad \epsilon = \max_{x \in \{0,1\}^m} \left\{ \Pr_{B \leftarrow \mathcal{B}}[x \in B_1] \right\}.$$

- **Phase 2:** $\mathcal{B}$ is run to obtain B. Let $F_1 = F^{(B)}$. A uniform bit b is chosen, and $\mathcal{D}$ is given quantum access to F_b.
- **Phase 3:** $\mathcal{D}$ loses access to F_b, and receives the randomness r used to invoke $\mathcal{B}$ in Phase 2. Then $\mathcal{D}$ outputs a guess b'.

Lemma 1 (Reprogramming Lemma). *For any $\mathcal{D}$ making q queries in expectation when its oracle is F_0, it holds that*

$$|\Pr[\mathcal{D} \text{ outputs } 1 \mid b = 1] - \Pr[\mathcal{D} \text{ outputs } 1 \mid b = 0]| \leq 2q \cdot \sqrt{\epsilon}.$$

The proof of this lemma can be found in [2].

3.2 The Adaptive Resampling Game

In [2], Alagic et al. first introduced *the adaptive resampling lemma* (ARS) for random permutations, which is a generalization of the "reprogramming lemma" of Grilo et al. [9, Proposition 1]. It considers a distinguisher that is granted quantum access either to a uniformly random permutation P or to a modified permutation obtained by swapping the images of two distinct inputs under P. The ARS lemma provides an upper bound on the distinguisher's advantage in distinguishing between these two. Formally, the ARS game for a distinguisher $\mathcal{D}$ is given as follows:

- **Phase 1:** A permutation P is uniformly chosen from $\mathcal{P}(n)$, and $\mathcal{D}$ is given quantum access to $P^{(0)} = P$ and to its inverse.
- **Phase 2:** $s_0, s_1 \in \{0,1\}^n$ are uniformly chosen, and we let

$$P^{(1)} = P \circ \mathsf{swap}_{s_0,s_1}.$$

  A uniform bit $b \in \{0,1\}$ is chosen, and $\mathcal{D}$ is given s_0, s_1, and quantum access to $P^{(b)}$ and to its inverse. The $\mathcal{D}$ outputs a bit b' (try to guess b).

Lemma 2 (Resampling Lemma). *For any distinguisher $\mathcal{D}$, playing the above adaptive resampling game and making at most q queries (including both forward and inverse queries) to $P^{(0)}$ in the first phase, it holds that*

$$|\Pr[\mathcal{D} \text{ outputs } 1 \mid b = 1] - \Pr[\mathcal{D} \text{ outputs } 1 \mid b = 0]| \;\leq\; 4 \cdot \sqrt{\frac{q}{2^n}}.$$

The proof of this lemma can be found in [2].

3.3 An Extended Adaptive Resampling Game

In our proofs, we must address the more general setting in which a single random permutation is reprogrammed via multiple *swap* operations. To capture this scenario, we extend the ARS lemma to accommodate more than one swap. We formalize this generalized setting through the following extended resampling game defined for a distinguisher $\mathcal{D}$.

- **Phase 1:** A permutation P is uniformly chosen from $\mathcal{P}(n)$, and $\mathcal{D}$ is given quantum access to $P^{(0)} = P$ and to its inverse.

- **Phase 2:** $s_0, s_1, \ldots, s_{2r-2}, s_{2r-1} \in \{0,1\}^n$ are uniformly chosen, and we let

$$P^{(1)} = P \circ \prod_{i=1}^{r} \mathsf{swap}_{s_{2i-2}, s_{2i-1}}.$$

  A uniform bit $b \in \{0,1\}$ is chosen, and $\mathcal{D}$ is given $s_0, s_1, \ldots, s_{2r-2}, s_{2r-1}$, and quantum access to $P^{(b)}$. The $\mathcal{D}$ outputs a bit b' (try to guess b).

Lemma 3 (Extended Resampling Lemma). *For any $\mathcal{D}$ making at most q queries (including both forward and inverse queries) to $P^{(0)}$ in the first phase, it holds that*

$$|\Pr[\mathcal{D} \text{ outputs } 1 \mid b = 1] - \Pr[\mathcal{D} \text{ outputs } 1 \mid b = 0]| \;\leq\; 4\sqrt{\frac{rq}{2^n}},$$

where r is the number of swap operations used to reprogram the permutation.

The proof of the above lemma proceeds along the same lines as the proof of Lemma 2 presented by Alagic et al. [2], with the necessary modifications to accommodate our extended setting. For an arbitrary function $f : \{0,1\}^n \to \{0,1\}^n$, define the state

$$|f\rangle_F = \bigotimes_{x \in \{0,1\}^n} |f(x)\rangle_{F_x}.$$

Let $F = \{F_x\}_{x \in \{0,1\}^n}$ be the collection of registers. The evaluation operator O acts as

$$O_{XYF} |x\rangle_X |y\rangle_Y |f\rangle_F = |x\rangle_X |y \oplus f(x)\rangle_Y |f\rangle_F.$$

For any permutation π, define

$$O^{\text{inv}}_{XYF} |\pi\rangle_F = \left(\sum_{x,y \in \{0,1\}^n} |y\rangle\langle y|_Y \otimes X^x_X \otimes |y\rangle\langle y|_{F_x} \right) |\pi\rangle_F.$$

Here X is the Pauli-X operator and for $x = (x_1, \ldots, x_n)$, $X^x := X^{x_1} \otimes \cdots \otimes X^{x_n}$, and $X^x |\hat{x}\rangle = |\hat{x} \oplus x\rangle$. Then,

$$O^{\text{inv}}_{XYF} |x\rangle_X |y\rangle_Y |\pi\rangle_F = |x \oplus \pi^{-1}(y)\rangle_X |y\rangle_Y |\pi\rangle_F.$$

We may view a uniform permutation as a uniform superposition over all permutations in $\mathcal{P}(n)$. That is, define the state

$$|\phi_0\rangle_F = \frac{1}{\sqrt{(2^n)!}} \sum_{\pi \in \mathcal{P}(n)} |\pi\rangle_F.$$

The final state of any oracle algorithm $\mathcal{D}$ is identically distributed whether we give $\mathcal{D}$ oracle access to a uniformly random permutation π and its inverse π^{-1}, or run $\mathcal{D}$ with access to O and O^{inv} after initializing the F-registers to $|\phi_0\rangle_F$ and finally measuring them at the end.

Now, we state the *Gentle Measurement Lemma* [1, Lemma 8], which is required in our proof.

Lemma 4 (**Gentle Measurement Lemma**). *Let $|\psi\rangle$ be a quantum state and let $\{P_i\}_{i=1}^q$ be a collection of projectors such that for all i, we have $\|\overline{P_i}\,|\psi\rangle\,\|_2^2 \leq \varepsilon_i$. Then,*

$$1 - |\langle\psi|\,(P_q \cdots P_1)\,|\psi\rangle|^2 \leq \sum_{i=1}^{q} \varepsilon_i\,.$$

where we denote the complementary projector of a projector P by $\overline{P} = \mathbb{I} - P$.

Now, we begin the proof of our Extended Resampling Lemma.

Proof (Lemma 3). We split the distinguisher $\mathcal{D}$ into two stages $\mathcal{D} = (\mathcal{D}_0, \mathcal{D}_1)$ corresponding to the two phases of the experiment. We run the experiment using the superposition oracle $|\phi_0\rangle_F$ and measure the F-registers at the end.

On average over the choice of reprogramming positions $\{(s_{2i-2}, s_{2i-1})\}_{i=1}^r$, the adversary–oracle state after $\mathcal{D}_0$ is almost invariant under swapping the registers $F_{s_{2i-2}}$ and $F_{s_{2i-1}}$ unless $\mathcal{D}_0$ makes many oracle queries.

Let $\tau = \{s_0, s_1, \ldots, s_{2r-2}, s_{2r-1}\}$. Now, we define $\sigma := (\prod_{i=1}^r \mathsf{swap}_{s_{2i-2},s_{2i-1}})$. Consider $\mathsf{S} := \mathsf{Support}(\sigma - \mathbb{I})$. We then define the following projectors

$$(P_\tau)_X := \mathbb{I} - \left(\sum_{s\in\mathsf{S}} |s\rangle\,\langle s|\right)$$

$$(P_\tau^{\mathrm{inv}})_{FY} := \sum_{y\in\{0,1\}^n} |y\rangle\,\langle y|_Y \otimes \prod_{s\in\mathsf{S}} (\mathbb{I} - |y\rangle\,\langle y|)_{F_s}\,.$$

Note that we could alternatively define the projectors using the set of distinct sampled points, however, it is more appropriate to define them over S, as it captures exactly those points where the reprogrammed permutation differs from the original permutation. For example, if $\tau = \{a, a, b, c\}$, then it induces the 2-cycle $(b\ c)$. The set of distinct sampled points is $\{a, b, c\}$, whereas $\mathsf{Support}(\sigma - \mathbb{I}) = \{b, c\}$, which precisely identifies the points on which the permutation is reprogrammed. It is easy to verify that for any s_0, s_1:

$$\left[\prod_{i=1}^{r} \mathsf{Swap}_{F_{s_{2i-2}}F_{s_{2i-1}}},\, O^{\mathsf{XYF}}(P_\tau)_X\right] = 0, \tag{1}$$

$$\left[\prod_{i=1}^{r} \mathsf{Swap}_{F_{s_{2i-2}}F_{s_{2i-1}}},\, O^{\mathrm{inv}}_{\mathsf{XYF}}\big(P_\tau^{\mathrm{inv}}\big)_{FY}\right] = 0. \tag{2}$$

where $[\cdot,\cdot]$ denotes the commutator operation, and $\mathsf{Swap}_{A,B}$ is the swap operator, i.e., $\mathsf{Swap}_{A,B}\,|x\rangle_A|x'\rangle_B = |x'\rangle_A|x\rangle_B$ if A and B are distinct registers, and the identity if $A = B$.

Since $\overline{P}_\tau := \mathbb{I} - P_\tau \leq \sum_{s\in\mathsf{S}}(|s\rangle\,\langle s|)$, it follows that for any normalized state $|\psi\rangle_{XE}$ (where E is an arbitrary register),

$$\mathbb{E}_{s_0,\ldots,s_{2r-1}}\left[\left\|(\overline{P}_\tau)_X\,|\psi\rangle_{XE}\right\|_2^2\right] \leq \mathbb{E}_{s_0,\ldots,s_{2r-1}}\left[\langle\psi|\left(\sum_{s\in\mathsf{S}}(|s\rangle\,\langle s|)\right)|\psi\rangle\right] \leq 2r\cdot 2^{-n}. \tag{3}$$

We show a similar statement for P_τ^{inv}. A valid adversary/oracle state $|\psi\rangle_{YXEF}$ (supported on the span of P_n) can be written as

$$|\psi\rangle_{YXEF} = \sum_{x,y\in\{0,1\}^n} c_{xy}\,|y\rangle_Y\,|y\rangle_{F_x}\,|\psi_{xy}\rangle_{XEF_{x^c}}\,,\ x^c = \{0,1\}^n \setminus \{x\}$$

for normalized states $\{|\psi_{xy}\rangle\}_{x,y}$ satisfying $\sum_{x,y}|c_{xy}|^2 = 1$, and $\langle y|\psi_{xy}\rangle_{F_{x'}} = 0$, for all $x' \neq x$. If for any $i \neq j$, $s_i = s_j$, then

$$\left\|(\overline{P}_\tau^{\text{inv}})_{YF}\,|\psi\rangle_{YXEF}\right\|_2^2 = 0 \leq 2r \cdot 2^{-n}.$$

Hence, we have

$$\mathbb{E}_{s_0,\ldots,s_{2r-1}}\left[\left\|(\overline{P}_\tau^{\text{inv}})_{YF}\,|\psi\rangle_{YXEF}\right\|_2^2\right] \leq 2r \cdot 2^{-n}. \tag{4}$$

Without loss of generality, assume $\mathcal{D}_0$ starts in $|\psi_0\rangle = |\psi_0'\rangle\,|\phi_0\rangle$, and computes

$$|\psi\rangle = U_{\mathcal{D}_0}\,|\psi_0\rangle = U_q O_q U_{q-1} O_{q-1} \ldots U_1 O_1\,|\psi_0\rangle\,.$$

For any $s_0, \ldots, s_{2r-1}$, decompose $|\psi\rangle = |\psi_{\text{good}}(\tau)\rangle + |\psi_{\text{bad}}(\tau)\rangle$ and define

$$|\psi_{\text{good}}(\tau)\rangle = z\,U_q O_q P_\tau^q U_{q-1} O_{q-1} P_\tau^{q-1} \ldots U_1 O_1 P_\tau^1\,|\psi_0\rangle\,,$$

where

$$P_\tau^i = \begin{cases} P_\tau & \text{if } O_i = O, \\ P_\tau^{\text{inv}} & \text{if } O_i = O^{\text{inv}}, \end{cases}$$

and $z \in \mathbb{C}$ with $|z| = 1$ is chosen such that $\langle\psi \mid \psi_{\text{good}}(\tau)\rangle \in \mathbb{R}_{\geq 0}$. We can also view it as

$$|\psi_{\text{good}}(\tau)\rangle = z\,U_{\mathcal{D}_0} Q_\tau^q \cdots Q_\tau^1\,|\psi_0\rangle\,,$$

with $Q_\tau^i = \tilde{U}_i^\dagger P_\tau^i \tilde{U}_i$, where $\tilde{U}_i = U_{i-1} O_{i-1} \cdots U_1 O_1$. Now, define

$$\varepsilon_i(\tau) := \left\|\overline{Q}_\tau^{\,i}\,|\psi_0\rangle\right\|_2^2 = \left\|\overline{P}_\tau^{\,i}\tilde{U}_i\,|\psi_0\rangle\right\|_2^2.$$

Then, by Lemma 4, we have

$$1 - |\,\langle\psi \mid \psi_{\text{good}}(\tau)\rangle\,|^2 \leq \sum_{i=1}^{q} \varepsilon_i(\tau)\,. \tag{5}$$

Now we analyze the effect of reprogramming the (superposition) oracle after Phase 1. Consider $S := \prod_{i=1}^{r} \mathsf{Swap}_{F_{s_{2i-2}}, F_{s_{2i-1}}}$. Here, reprogramming swaps the values of the permutation at s_{2i-2} and s_{2i-1}. Observe that $S\,|\phi_0\rangle = |\phi_0\rangle$. As the operators U_j do not act on register F, from Eq. (1) and Eq. (2), we have

$$S\,|\psi_{\text{good}}(\tau)\rangle = |\psi_{\text{good}}(\tau)\rangle\ .$$

From the standard formula for the trace distance of pure states, we get

$$\frac{1}{2}\left\||\psi\rangle\langle\psi| - S\,|\psi\rangle\langle\psi|\,S\right\|_1 = \sqrt{1 - |\,\langle\psi|\,S\,|\psi\rangle\,|^2}\,.$$

Now, we have

$$\begin{aligned}
|\,\langle\psi|\,S\,|\psi\rangle\,| &= |\,\langle\psi_{\text{good}}(\tau)|\,S\,|\psi_{\text{good}}(\tau)\rangle + \langle\psi_{\text{good}}(\tau)|\,S\,|\psi_{\text{bad}}(\tau)\rangle \\
&\quad + \langle\psi_{\text{bad}}(\tau)|\,S\,|\psi_{\text{good}}(\tau)\rangle + \langle\psi_{\text{bad}}(\tau)|\,S\,|\psi_{\text{bad}}(\tau)\rangle\,| \\
&= |\,\langle\psi_{\text{good}}(\tau)|\,|\psi_{\text{good}}(\tau)\rangle + \langle\psi_{\text{good}}(\tau)|\,|\psi_{\text{bad}}(\tau)\rangle \\
&\quad + \langle\psi_{\text{bad}}(\tau)|\,|\psi_{\text{good}}(\tau)\rangle + \langle\psi_{\text{bad}}(\tau)|\,S\,|\psi_{\text{bad}}(\tau)\rangle\,| \\
&= |\,\langle\psi|\psi\rangle + \langle\psi_{\text{bad}}(\tau)|\,(S - \mathbb{I})\,|\psi_{\text{bad}}(\tau)\rangle\,| \\
&\geq 1 - |\,\langle\psi_{\text{bad}}(\tau)|\,(S - \mathbb{I})\,|\psi_{\text{bad}}(\tau)\rangle\,| \\
&\geq 1 - 2\left\||\psi_{\text{bad}}(\tau)\rangle\right\|_2^2.
\end{aligned}$$

Here, the above analysis follows from properties of Unitary operators and from the Cauchy-Schwarz and triangle inequalities. Thus, we have:

$$\frac{1}{2}\left\| |\psi\rangle\langle\psi| - S|\psi\rangle\langle\psi|S \right\|_1 \leq 2\left\| |\psi_{\text{bad}}(\tau)\rangle \right\|_2 .$$

We know that $|\psi_{\text{bad}}(\tau)\rangle = |\psi\rangle - |\psi_{\text{good}}(\tau)\rangle$, therefore

$$\left\| |\psi_{\text{bad}}(\tau)\rangle \right\|_2^2 = 2 - 2\cdot \text{Re}(\langle\psi|\psi_{\text{good}}(\tau)\rangle) = 2 - 2\cdot |\langle\psi|\psi_{\text{good}}(\tau)\rangle| \leq 2\sum_{i=1}^{q}\varepsilon_i(\tau) .$$

Combining the last two results, we obtain

$$\frac{1}{2}\left\| |\psi\rangle\langle\psi| - S|\psi\rangle\langle\psi|S \right\|_1 \leq 2\sqrt{2}\sqrt{\sum_{i=1}^{q}\varepsilon_i(\tau)} . \tag{6}$$

Now, the rest of the proof is similar to that of the proof of [9, Theorem 6]. By the bound relating the distinguishing advantage and trace distance, we have

$$\begin{aligned}
\Pr[\mathcal{D} \text{ outputs } \ b] - \frac{1}{2} &\leq \frac{1}{4}.\mathbb{E}_{s_0,\ldots,s_{2r-1}}\left[\left\| |\psi\rangle\langle\psi| - S|\psi\rangle\langle\psi|S \right\|_1\right] \\
&\leq \sqrt{2}\mathbb{E}_{s_0,\ldots,s_{2r-1}}\left[\sqrt{\sum_{i=1}^{q}\varepsilon_i(\tau)}\right] \\
&\leq \sqrt{2}\sqrt{\mathbb{E}_{s_0,\ldots,s_{2r-1}}\left[\sum_{i=1}^{q}\varepsilon_i(\tau)\right]} \\
&\leq 2\sqrt{\frac{rq}{2^n}} ,
\end{aligned}$$

where above analysis follows from Eq. (3), Eq. (4), Eq. (6) and Jensen's inequality. □

4 Post-Quantum Security of KSoP and KSoP21

In this section, we prove the post-quantum security of the KSoP and KSoP21 constructions. Since the security proofs for both constructions closely resemble each other, we present a unified proof, explicitly highlighting the necessary changes for each construction. Formally, the security result is as follows.

Theorem 1. *For any adversary $\mathcal{A}$ that makes q classical queries to* Cons*, i.e.,* $\mathsf{E}[\mathsf{P}_1, \mathsf{P}_2]$ *and p quantum queries to* $\mathsf{P}_1, \mathsf{P}_2$ *(and their inverses) in total, it holds that*

$$\mathbf{Adv}^{\texttt{pq-prf}}_{\mathsf{Cons}}(\mathcal{A}) \leq \frac{3q^2}{2^n} + \frac{4p\sqrt{q} + 12q\sqrt{p}}{2^{n/2}},$$

where Cons *denotes either the* KSoP *or the* KSoP21 *construction.*

The proof roughly goes as follows. We use a series of hybrids indexed by j ($\mathbf{H}_j$), where the first j classical queries are answered by random functions. The PRF advantage is then expressed depending on the probability to output 1 or to obtain a bad transcript, in all these games. For two consecutive games, to move from $\mathbf{H}_j$ to $\mathbf{H}_{j+1}$, we introduce an intermediate hybrid $\hat{\mathbf{H}}_j$. This is where we use first the reprogramming (Lemma 1), then the resampling (Lemma 2) bounds.

4.1 Proof of Theorem 1

This section is devoted to establishing the security bound as demonstrated in Theorem 1. Due to the structural similarity of the security proofs for KSoP and KSoP21, we present a combined proof of both the variants.

Let $\mathsf{F} \in \mathcal{F}(n,n)$ denote a uniform random function. Let $\mathcal{A}$ be a distinguisher and the goal of $\mathcal{A}$ is to distinguish between the *real world* $(\mathsf{E}[\mathsf{P}_1,\mathsf{P}_2],\mathsf{P}_1,\mathsf{P}_2)$ and the *ideal world* $(\mathsf{F},\mathsf{P}_1,\mathsf{P}_2)$. In the real world, the adversary $\mathcal{A}$ has *classical* access to the construction $\mathsf{E}[\mathsf{P}_1,\mathsf{P}_2]$, and *quantum* access to the permutations $\mathsf{P}_1,\mathsf{P}_2$ in both the forward and inverse directions. On the other hand, in the ideal world $\mathcal{A}$ has *classical* access to a random function F and *quantum* access to the permutations $\mathsf{P}_1,\mathsf{P}_2$ in both the forward and inverse directions.

Without loss of generality, we assume that $\mathcal{A}$ never makes any redundant (classical) construction query. Let the adversary $\mathcal{A}$ makes q many (classical) construction queries and p many (quantum) primitive queries in total. We divide the execution of $\mathcal{A}$ into $q+1$ stages $0,\ldots,q$ where the j-th stage corresponds to the time between j-th and $(j+1)$-th classical queries of $\mathcal{A}$. We allow $\mathcal{A}$ to adaptively distribute its p quantum queries between these stages arbitrarily. We let p_j denote the number of quantum queries made in the j-th stage. We denote the j-th classical query of $\mathcal{A}$ by (x_j,y_j), that means $\mathcal{A}$ queries x_j to the construction and receives response y_j. Let $T_j = \{(x_1,y_1),\ldots,(x_j,y_j)\}$ be the ordered list describing the first j many classical queries made by $\mathcal{A}$.

We use "$\prod$" to denote sequential composition of operations, i.e., $\prod_{i=1}^{n} f_i = f_1 \circ \cdots \circ f_n$. Note that order matters, since in general the composition of operators is not commutative. Now, we define the reprogrammed permutations corresponding to P_1 and P_2 respectively. For any $i \in [j]$, we sequentially choose random values $u_i \in \{0,1\}^n \setminus (\mathcal{U} \cup \mathcal{Y} \cup \mathcal{P}_1 \cup \mathcal{P}_2)$, where $\mathcal{U} = \{u_1,\ldots,u_{i-1}\}$, $\mathcal{Y} = \{y_i \oplus u_1 \oplus y_1,\ldots,y_i \oplus u_{i-1} \oplus y_{i-1}\}$, $\mathcal{P}_1 = \{\mathsf{P}_1(x_1 \oplus k_1),\ldots,\mathsf{P}_1(x_i \oplus k_1)\}$, $\mathcal{P}_2 = \{\mathsf{P}_2(x_1 \oplus k_2) \oplus y_i,\ldots,\mathsf{P}_2(x_i \oplus k_2) \oplus y_i\}$. Now, the corresponding reprogramming operators are defined as

$$\overrightarrow{S}_{T_j,\mathsf{P}_1,k} := \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}_1(x_i \oplus k_1),u_i}, \text{ and } \overrightarrow{S}_{T_j,\mathsf{P}_2,k} = \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}_2(x_i \oplus k_2),(y_i \oplus u_i)}.$$

We define the reprogrammed permutations $\mathsf{P}_{1,T_j,k}$ and $\mathsf{P}_{2,T_j,k}$ by

$$\mathsf{P}_{1,T_j,k} = \overrightarrow{S}_{T_j,\mathsf{P}_1,k} \circ \mathsf{P}_1,\ \mathsf{P}_{2,T_j,k} = \overrightarrow{S}_{T_j,\mathsf{P}_2,k} \circ \mathsf{P}_2.$$

For KSoP21, in all the details above, we consider $k_1 = k_2 = k$.

Now, let $\mathbf{H}_0,\ldots,\mathbf{H}_q$ denote a sequence of $q+1$ hybrids, where $\mathbf{H}_j$ is defined as follows:

1. - For KSoP, sample $k = (k_1,k_2) \xleftarrow{\$} \{0,1\}^{2n}$.
   - For KSoP21, sample $k \xleftarrow{\$} \{0,1\}^n$.
2. Answer the first j classical queries of $\mathcal{A}$ using F and the quantum queries of $\mathcal{A}$ in the first j stages (which ends when the $(j+1)$-th classical query is made) using $\mathsf{P}_1,\mathsf{P}_2$.
3. Answer all subsequent classical queries from $\mathcal{A}$ using $\mathsf{E}_k[\mathsf{P}_1,\mathsf{P}_2]$ and all subsequent quantum queries from $\mathcal{A}$ using $\mathsf{P}_{1,T_j,k}$ and $\mathsf{P}_{2,T_j,k}$.

Since, $T_0 = \emptyset$, we deduce that $\mathbf{H}_0$ corresponds to $\mathcal{A}$'s interaction with the real world. Similarly, one can easily notice that $\mathbf{H}_q$ corresponds to $\mathcal{A}$'s interaction with the ideal world. Thus, we have

$$\mathbf{Adv}^{\mathtt{pq\text{-}prf}}_{\mathsf{Cons}}(\mathcal{A}) = \mid \Pr[\mathcal{A}(\mathbf{H}_0) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_q) \to 1] \mid, \tag{7}$$

where $\mathsf{Cons} = \mathsf{KSoP}$ or $\mathsf{Cons} = \mathsf{KSoP21}$.

Now for any $j \in \{0, \ldots, q\}$, $\mathbf{H}_j$ can be viewed as a string of oracle calls made by $\mathcal{A}$

$$\underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_j,k}, \mathsf{P}_{2,T_j,k})^{q-j}}_{q-j \text{ classical queries}},$$

where each occurrence of F (res. $\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2]$) indicates a single classical query and each occurrence of $\mathsf{P}_1, \mathsf{P}_2$ (res. $\mathsf{P}_{1,T_j,k}, \mathsf{P}_{2,T_j,k}$) indicates the sequence of quantum primitive queries. Now, in this representation we want to compare

$$\mathbf{H}_j \quad : \quad \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_j,k}, \mathsf{P}_{2,T_j,k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_j,k}, \mathsf{P}_{2,T_j,k})^{q-j-1}}_{q-j-1 \text{ classical queries}},$$

$$\mathbf{H}_{j+1} \quad : \quad \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{F}, \quad \mathsf{P}_1, \quad \mathsf{P}_2, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k})^{q-j-1}}_{q-j-1 \text{ classical queries}}.$$

To aid this comparison we introduce an intermediate hybrid $\hat{\mathbf{H}}_j$ defined exactly as $\mathbf{H}_j$, except that the $(j+1)$-th classical query of $\mathcal{A}$ is answered using F rather than $\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2]$. Accordingly, all subsequent quantum queries are answered using the reprogrammed permutations $\mathsf{P}_{1,T_{j+1},k}$ and $\mathsf{P}_{2,T_{j+1},k}$.

Now, we compare the following hybrids in pairs from top to bottom:

$$\mathbf{H}_j : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_j,k}, \mathsf{P}_{2,T_j,k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_j,k}, \mathsf{P}_{2,T_j,k})^{q-j-1}}_{q-j-1 \text{ classical queries}},$$

$$\hat{\mathbf{H}}_j : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{F}, \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k})^{q-j-1}}_{q-j-1 \text{ classical queries}},$$

$$\mathbf{H}_{j+1} : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{F}, \quad \mathsf{P}_1, \quad \mathsf{P}_2, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k})^{q-j-1}}_{q-j-1 \text{ classical queries}}.$$

In the following, we state two important lemmas - Lemma 5 and Lemma 6 that establish the bounds on the distinguishability between $\hat{\mathbf{H}}_j$ and $\mathbf{H}_{j+1}$, between $\mathbf{H}_j$ and $\hat{\mathbf{H}}_j$, respectively. The proofs of the lemmas are given in Section 4.2 and Section 4.3.

Lemma 5. *For $j = 0, \ldots, q-1$, we have*

$$|\Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_{j+1}) \to 1]| \leq 2p_{j+1}\sqrt{\frac{4(j+1)}{2^n}},$$

where p_{j+1} is the expected number of quantum queries $\mathcal{A}$ makes in the $(j+1)$-th stage of the game.

Lemma 6. *For $j = 0, \ldots, q-1$, we have*

$$|\Pr[\mathcal{A}(\mathbf{H}_j) \to 1] - \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1]| \leq \frac{(q+2j)}{2^n} + 12\sqrt{\frac{p}{2^n}},$$

where q and p are the expected number of classical and quantum queries made by $\mathcal{A}$.

Given the above two lemmas and Eq. (7), we have

$$\begin{aligned}
\mathbf{Adv}^{\texttt{pq-prf}}_{\mathsf{KSoP}}(\mathcal{A}) =& \mid \Pr[\mathcal{A}(\mathbf{H}_0) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_q) \to 1] \mid \\
&\leq \sum_{j=0}^{q-1} \Big(\mid \Pr[\mathcal{A}(\mathbf{H}_j) \to 1] - \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] \mid \\
&\qquad\qquad + \mid \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_{j+1}) \to 1] \mid \Big) \\
&\leq \sum_{j=0}^{q-1} \left(\frac{(q+2j)}{2^n} + 12\sqrt{\frac{p}{2^n}} + 2p_{j+1}\sqrt{\frac{4(j+1)}{2^n}} \right) \\
&\leq \frac{3q^2}{2^n} + 12q\sqrt{\frac{p}{2^n}} + 4p\sqrt{\frac{q}{2^n}} \\
&\leq \frac{3q^2}{2^n} + \frac{4p\sqrt{q} + 12q\sqrt{p}}{2^{n/2}}.
\end{aligned}$$

4.2 Proof of Lemma 5

Recall, we can have the following string of oracle call representations for the two hybrids,

$$\hat{\mathbf{H}}_j : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{F}, \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k})^{q-j-1}}_{q-j-1 \text{ classical queries}},$$

$$\mathbf{H}_{j+1} : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{F}, \mathsf{P}_1, \mathsf{P}_2, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k})^{q-j-1}}_{q-j-1 \text{ classical queries}}.$$

It is clear that the two hybrids are identical, except for the $(j+1)$-th stage, where $\hat{\mathbf{H}}_j$ uses reprogrammed versions of the permutations P_1 and P_2, and $\mathbf{H}_{j+1}$ uses the original permutations P_1 and P_2. We will show that the distinguishing advantage of these two hybrids can be captured by the *reprogramming* bounds of the permutations.

More formally, given the adversary $\mathcal{A}$, we construct an ARP distinguisher $\mathcal{D}$ as follows (note that Phases 1, 2, 3 correspond to Phases 1, 2, 3 in the ARP game respectively).

Phase 1. For all the queries up to and including the $(j+1)$-th classical query:

1. $\mathcal{D}$ samples uniform random permutations $\mathsf{P}_1, \mathsf{P}_2 \leftarrow \mathcal{P}(n)$ and uniform random function $\mathsf{F} \leftarrow \mathcal{F}(n.n)$.
2. All classical queries from $\mathcal{A}$ are answered using F. The transcript of classical queries and responses is denoted by $T_{j+1} = \{(x_1, y_1), \ldots, (x_{j+1}, y_{j+1})\}$.
3. All quantum queries from $\mathcal{A}$ are answered using P_1,P_2. More formally, $\mathcal{D}$ defines $F(\alpha, t, x) := \mathsf{P}^t_\alpha(x)$, for $t \in \{1, -1\}$ and $\alpha \in \{1, 2\}$.
4. $\mathcal{D}$ defines the randomized algorithm $\mathcal{B}$, which takes a key k, computes a set B of input/output pairs to be reprogrammed so that $F^{(B)}(\alpha, t, x) = \mathsf{P}^t_{\alpha,T_{j+1},k}(x)$, for all α, t, x.

Phase 2. For all the quantum queries in the $(j+1)$-th stage:

1. $\mathcal{B}$ is run to generate the set B and $\mathcal{D}$ is given quantum access to F_b, where $F_0 = F(= \mathsf{P}^t_\alpha)$ and $F_1 = F^{(B)}(= \mathsf{P}^t_{\alpha,T_{j+1},k})$.
2. All quantum queries from $\mathcal{A}$ are answered using F_b.

Phase 3. For all the queries beyond and including $(j+2)$-th classical query:

1. $\mathcal{D}$ is given the randomness (specifically the key k here) used by $\mathcal{B}$ to generate B.

2. All classical queries from $\mathcal{A}$ are answered using $\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2]$.
3. All quantum queries from $\mathcal{A}$ are answered using F_1.
4. Finally, $\mathcal{D}$ returns whatever $\mathcal{A}$ returns.

It is clear that $\mathcal{D}$ is a valid ARP distinguisher.

- **Case $b = 0$:** Here, the oracle used by $\mathcal{D}$ during `Phase` 2 is $F_0 = F$. Consequently, the output distribution of $\mathcal{A}$ matches its output distribution in the hybrid experiment $\mathbf{H}_{j+1}$.
- **Case $b = 1$:** In this case, the oracle used by $\mathcal{D}$ during `Phase` 2 is $F_1 = F^{(B)}$. Therefore, the output distribution of $\mathcal{A}$ is identical to its output distribution in the experiment $\hat{\mathbf{H}}_j$.

Thus

$$| \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_{j+1}) \to 1] | = | \Pr[\mathcal{D} \to 1 \mid b = 1] - \Pr[\mathcal{D} \to 1 \mid b = 0] | .$$

To bound this quantity using Lemma 1, we need to bound ϵ ,i.e., the reprogramming probability. Also, we need a bound on the expected number of queries made by $\mathcal{D}$ in `Phase` 2 (when $F_0 = F$).

We bound the reprogramming probability ϵ, considering the definition of $\mathsf{P}_{\alpha,T_{j+1},k}$ and the fact that $F^{(B)}(\alpha, t, x) = \mathsf{P}^t_{\alpha,T_{j+1},k}(x)$. If we fix $\mathsf{P}_1, \mathsf{P}_2$ and T_{j+1}, then the oracle differs from the original one only on the set B_1. The probability that any given (α, t, x) is reprogrammed is at most the probability (over k) that it is in the set:

$$B_1 = \{(1, 1, (x_i \oplus k_1)), (1, -1, \mathsf{P}_1^{-1}(u_i)), (2, 1, (x_i \oplus k_2)), (2, -1, \mathsf{P}_2^{-1}(y_i \oplus u_i))\}_{i=1}^{j+1}.$$

Again, for KSoP21, we get the respective set by substituting $k_1 = k_2 = k$. Here, for any (α, t, x), reprogramming occurs only if (α, t, x) coincides with one of the $4(j + 1)$ reprogrammed points. So, the probability that (α, t, x) is reprogrammed is upper bounded by $4(j + 1)/2^n$. Hence, we get $\epsilon \leq 4(j + 1)/2^n$.

When $F_0 = F$, the expected number of queries by $\mathcal{D}$ during `Phase` 2 is precisely the expected number of queries by $\mathcal{A}$ in the $(j + 1)$-th stage, i.e., p_{j+1}. Hence, invoking Lemma 1, it follows that

$$| \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_{j+1}) \to 1] | \leq 2p_{j+1}\sqrt{\frac{4(j+1)}{2^n}}.$$

4.3 Proof of Lemma 6

For bounding the distinguishing advantage between $\mathbf{H}_j$ and $\hat{\mathbf{H}}_j$, we will introduce two intermediate hybrids called $\mathbf{\Gamma}_j$, $\mathbf{\Sigma}_j$. Now, we bound the distinguishing advantages between the following hybrids in pairs from top to bottom.

$$\mathbf{H}_j : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \quad \mathsf{P}_{1,T_j,k}, \quad \mathsf{P}_{2,T_j,k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_j,k}, \mathsf{P}_{2,T_j,k})^{q-j-1}}_{q-j-1 \text{ classical queries}}.$$

$$\mathbf{\Gamma}_j : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2], \mathsf{P}^{(1)}_{1,T_j,k}, \mathsf{P}_{2,T_j,k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2], \mathsf{P}^{(1)}_{1,T_j,k}, \mathsf{P}_{2,T_j,k})^{q-j-1}}_{q-j-1 \text{ classical queries}}.$$

$$\mathbf{\Sigma}_j : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2^{(1)}], \mathsf{P}^{(1)}_{1,T_j,k}, \mathsf{P}^{(1)}_{2,T_j,k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2^{(1)}], \mathsf{P}^{(1)}_{1,T_j,k}, \mathsf{P}^{(1)}_{2,T_j,k})^{q-j-1}}_{q-j-1 \text{ classical queries}}.$$

$$\hat{\mathbf{H}}_j : \underbrace{\mathsf{P}_1, \mathsf{P}_2, (\mathsf{F}, \mathsf{P}_1, \mathsf{P}_2)^j}_{j \text{ classical queries}}, \mathsf{F}, \quad \mathsf{P}_{1,T_{j+1},k}, \quad \mathsf{P}_{2,T_{j+1},k}, \underbrace{(\mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2], \mathsf{P}_{1,T_{j+1},k}, \mathsf{P}_{2,T_{j+1},k})^{q-j-1}}_{q-j-1 \text{ classical queries}}.$$

Let, $\mathcal{A}$ be an adversary of $\mathbf{H}_j$ and $\mathbf{\Gamma}_j$. From $\mathcal{A}$, we can construct an ARS distinguisher $\mathcal{D}$ as follows.

Phase 1. For all the queries up to (immediately before) the $(j+1)$-th classical query:

1. $\mathcal{D}$ is given quantum oracle access to permutation P_1 and $\mathcal{D}$ samples an independent random permutation $\mathsf{P}_2 \in \mathcal{P}(n)$ and uniform random function $\mathsf{F} \in \mathcal{F}(n,n)$.
2. All classical queries from $\mathcal{A}$ are answered using F. Let the resulting transcript of classical queries and responses be denoted by $T_j = \{(x_1, y_1), \ldots, (x_j, y_j)\}$.
3. All quantum queries from $\mathcal{A}$ are answered using P_1,P_2.

Phase 2. For all the queries beyond and including $(j+1)$-th classical query:

1. $\mathcal{D}$ receives $s_0, s_1 \in \{0,1\}^n$ and $\mathcal{D}$ has now quantum oracle access to $\mathsf{P}_1^{(b)}$, where $\mathsf{P}_1^{(0)} = \mathsf{P}_1$ and $\mathsf{P}_1^{(1)} = \mathsf{swap}_{\mathsf{P}_1(s_0),\mathsf{P}_1(s_1)} \circ \mathsf{P}_1$.
2. Let $\mathcal{D}$ receives the $(j+1)$-th classical query from $\mathcal{A}$ which is x_{j+1}.
3. - For KSoP: $\mathcal{D}$ sets $k_1 = x_{j+1} \oplus s_0$ and $k_2 = x_{j+1} \oplus s_1$, and $k = (k_1, k_2)$.
   - For $\mathsf{KSoP21}$, $\mathcal{D}$ sets $k_1 = x_{j+1} \oplus s_0$.
4. All classical queries from $\mathcal{A}$ are answered using $\mathsf{E}_k[\mathsf{P}_1^{(b)}, \mathsf{P}_2]$.
5. All quantum queries from $\mathcal{A}$ are answered using $\mathsf{P}_{1,T_j,k}^{(b)}$ and $\mathsf{P}_{2,T_j,k}$ correspondingly.
6. Finally $\mathcal{D}$ returns whatever $\mathcal{A}$ returns.

It is clear that $\mathcal{D}$ is a valid ARS distinguisher. If $b = 0$, then $\mathsf{P}_1^{(0)} = \mathsf{P}_1$, thus the output distribution of $\mathcal{A}$ is identical to its output distribution in $\mathbf{H}_j$. On the other hand, if $b = 1$, then the output of $\mathcal{A}$ is identically distributed to its output in $\mathbf{\Gamma}_j$. Hence, the distinguishing advantage between the hybrids $\mathbf{H}_j$ and $\mathbf{\Gamma}_j$ is equal to the distinguishing advantage of $\mathcal{D}$ in the resampling game. Using Lemma 2 we have

$$| \Pr[\mathcal{A}(\mathbf{H}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{\Gamma}_j) \to 1] | \leq 4\sqrt{\frac{p}{2^n}}. \tag{8}$$

Now, we bound the distinguishing advantage between the hybrids $\mathbf{\Gamma}_j$ and $\mathbf{\Sigma}_j$. Let $\mathcal{A}$ be an adversary between these two hybrids, we can construct an ARS distinguisher $\mathcal{D}$ as follows.

Phase 1. For all the queries up to (immediately before) the $(j+1)$-th classical query:

1. $\mathcal{D}$ is given quantum oracle access to permutation P_2 and $\mathcal{D}$ samples an independent random permutation $\mathsf{P}_1 \in \mathcal{P}(n)$ and a random function $\mathsf{F} \in \mathcal{F}(n,n)$.
2. All classical queries from $\mathcal{A}$ are answered using F. Let the resulting transcript of classical queries and responses be denoted by $T_j = \{(x_1, y_1), \ldots, (x_j, y_j)\}$.
3. All quantum queries from $\mathcal{A}$ are answered using P_1,P_2.

Phase 2. For all the queries beyond and including $(j+1)$-th classical query:

1. $\mathcal{D}$ receives $s_0, s_1 \in \{0,1\}^n$ and $\mathcal{D}$ has now quantum oracle access to $\mathsf{P}_2^{(b)}$, where $\mathsf{P}_2^{(0)} = \mathsf{P}_2$ and $\mathsf{P}_2^{(1)} = \mathsf{swap}_{\mathsf{P}_2(s_0),\mathsf{P}_2(s_1)} \circ \mathsf{P}_2$.
2. - For KSoP: $\mathcal{D}$ sets $k_1 = x_{j+1} \oplus s_0$ and $k_2 = x_{j+1} \oplus s_1$, and $k = (k_1, k_2)$.
   - For $\mathsf{KSoP21}$, $\mathcal{D}$ sets $k_1 = x_{j+1} \oplus s_0$.

   Then $\mathcal{D}$ reprogram the permutation P_1 to get $\mathsf{P}_1^{(1)}$.

3. All classical queries from $\mathcal{A}$ are answered using $\mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2^{(b)}]$.
4. All quantum queries from $\mathcal{A}$ are answered using $\mathsf{P}_{1,T_j,k}^{(1)}$ and $\mathsf{P}_{2,T_j,k}^{(b)}$ correspondingly.
5. Finally $\mathcal{D}$ returns whatever $\mathcal{A}$ returns.

Clearly, $\mathcal{D}$ is a valid ARS distinguisher. If $b = 0$, then $\mathsf{P}_2^{(0)} = \mathsf{P}_2$, thus the output distribution of $\mathcal{A}$ is identical to its output distribution in $\mathbf{\Gamma}_j$. On the other hand, if $b = 1$, then the output of $\mathcal{A}$ is identically distributed to its output in $\mathbf{\Sigma}_j$. Hence, the distinguishing advantage between the hybrids $\mathbf{\Gamma}_j$ and $\mathbf{\Sigma}_j$ is equal to the distinguishing advantage of $\mathcal{D}$ in the resampling game. Using Lemma 2 we have

$$| \Pr[\mathcal{A}(\mathbf{\Gamma}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{\Sigma}_j) \to 1] | \leq 4\sqrt{\frac{p}{2^n}}. \tag{9}$$

Now, consider the following three events for KSoP:

1. bad_1: $(s_1 \oplus k_1) \in \{x_1, \ldots, x_j\}$.
2. bad_2: $\mathsf{P}_1(s_1) \in \{u_1, \ldots, u_j\} \cup \{y_{j+1} \oplus u_1 \oplus y_1, \ldots, y_{j+1} \oplus u_j \oplus y_j\}$.
3. bad_3: In phase 2, $\mathcal{A}$ queries its classical oracle on $s_1 \oplus k_1$ i.e $s_1 \oplus k_1 \in \{x_{j+2}, \ldots, x_q\}$.

Remark 1. Observe that in KSoP, $(s_1 \oplus k_1) = (s_1 \oplus s_0 \oplus x_{j+1}) = (s_0 \oplus k_2)$.

Again, the above bad events are also considered for KSoP21, for which $k_1 = k_2 = k$. From the randomness of s_0 and s_1, we get that $\Pr[\mathsf{bad}_1] \leq j/2^n$. Similarly from the randomness of P_1, we get that $\Pr[\mathsf{bad}_2] \leq 2j/2^n$. Now for the event bad_3 consider the distinguisher $\mathcal{D}'$ that behaves identically to $\mathcal{D}$ in `Phase 1` and `Phase 2`, but then when $\mathcal{A}$ terminates outputs 1 iff event bad_3 occurred. When $b = 0$, the probability that bad_3 occurs is at most $(q - j)/2^n$. Since $\mathcal{D}'$ is a distinguisher for the resampling game, so, we get $\Pr[\mathsf{bad}_3] \leq (q - j)/2^n + 4\sqrt{p/2^n}$.

In Algorithm 1, we described $\mathbf{\Sigma}_j$ and $\mathbf{\Sigma}'_j$. Observe that, $\mathbf{\Sigma}_j$ and $\mathbf{\Sigma}'_j$ are identical until either bad_1 or bad_2 or bad_3 occur, and so we have

$$| \Pr[\mathcal{A}(\mathbf{\Sigma}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{\Sigma}'_j) \to 1] | \leq \Pr[\mathsf{bad}_1 \vee \mathsf{bad}_2 \vee \mathsf{bad}_3] \leq \frac{(q + 2j)}{2^n} + 4\sqrt{\frac{p}{2^n}}. \tag{10}$$

Now, we are arguing that $\mathbf{\Sigma}'_j$ is identical to $\hat{\mathbf{H}}_j$:

1. Up to (just before) the $(j + 1)$-th classical query $\mathbf{\Sigma}'_j$ is exactly identical to $\hat{\mathbf{H}}_j$.
2. In $\mathbf{\Sigma}'_j$, the response of the $(j + 1)$-th classical query, y_{j+1} is defined as:

$$y_{j+1} = \mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2^{(1)}](x_{j+1}) = \mathsf{P}_1^{(1)}(x_{j+1} \oplus k_1) \oplus \mathsf{P}_2^{(1)}(x_{j+1} \oplus k_2)$$

   Thus, considering the respective substitution of keys, we get:

$$y_{j+1} = \begin{cases} \mathsf{P}_1^{(1)}(s_0) \oplus \mathsf{P}_2^{(1)}(s_1) = \mathsf{P}_1(s_1) \oplus \mathsf{P}_2(s_0), & \text{for KSoP}, \\ \mathsf{P}_1^{(1)}(s_0) \oplus \mathsf{P}_2^{(1)}(s_0) = \mathsf{P}_1(s_1) \oplus \mathsf{P}_2(s_1), & \text{for KSoP21}. \end{cases}$$

   So, the value y_{j+1} is uniformly distributed.

3. For KSoP, in $\mathbf{\Sigma}'_j$, the oracles Q_1 and Q_2 are always equal to $\mathsf{P}_{1,T_{j+1},k}$ and $\mathsf{P}_{2,T_{j+1},k}$ respectively. If "$\mathsf{bad}_1 = \mathsf{true}$" or "$\mathsf{bad}_2 = \mathsf{true}$" , then this is immediate from line 12 of Algorithm 1. If bad_1 does not occur, then for $i = 1, \ldots, j$ we have $x_i \oplus k_1 \neq s_0$ (because, we assumed no redundant query) and $x_i \oplus k_1 \neq s_1$ (by definition of

Algorithm 1 Hybrid experiment $\mathbf{\Sigma}_j$. Including the boxed statements results in $\mathbf{\Sigma}'_j$.

1: $\mathsf{P}_1, \mathsf{P}_2 \xleftarrow{\$} \mathcal{P}(n)$, $\mathsf{F} \xleftarrow{\$} \mathcal{F}(n,n)$
2: Run $\mathcal{A}$ with quantum access to $\mathsf{P}_1, \mathsf{P}_2$ and classical access to F, until $\mathcal{A}$ makes its $(j+1)$-th classical query x_{j+1}
3: $T_j := \{(x_1, y_1), \ldots, (x_j, y_j)\}$
4: $s_0, s_1 \xleftarrow{\$} \{0,1\}^n$, $\mathsf{P}_1^{(1)} := \mathsf{P}_1 \circ \mathsf{swap}_{s_0,s_1}$, $\mathsf{P}_2^{(1)} := \mathsf{P}_2 \circ \mathsf{swap}_{s_0,s_1}$
5: Set the key
- For KSoP : $k_1 := s_0 \oplus x_{j+1}$, $k_2 := s_1 \oplus x_{j+1}$, $k := (k_1, k_2)$
- For $\mathsf{KSoP21}$: $k := s_0 \oplus x_{j+1}$.

6: $y_{j+1} := \mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2^{(1)}](x_{j+1})$
7: $Q_1 := \mathsf{P}^{(1)}_{1,T_j,k}$, $Q_2 := \mathsf{P}^{(1)}_{2,T_j,k}$
8: Give y_{j+1} to $\mathcal{A}$ as the answer to its $(j+1)$-th classical query
9: $T_{j+1} := \{(x_1, y_1), \ldots, (x_{j+1}, y_{j+1})\}$
10: $u_{j+1} := \mathsf{P}_1(s_1)$
11: **if** $\mathsf{bad}_1 := \mathsf{true}$, or $\mathsf{bad}_2 := \mathsf{true}$ **then**
12: $\boxed{Q_1 := \mathsf{P}_{1,T_{j+1},k} \text{ and } Q_2 := \mathsf{P}_{2,T_{j+1},k}}$
13: Continue running $\mathcal{A}$ with quantum access to Q_1, Q_2
(The remaining classical queries are answered as follows:)
14: $y := \mathsf{E}_k[\mathsf{P}_1^{(1)}, \mathsf{P}_2^{(1)}](x)$
15: **if** $\mathsf{bad}_3 := \mathsf{true}$, **then**
16: $\boxed{y := \mathsf{E}_k[\mathsf{P}_1, \mathsf{P}_2]}$
17: return y.

bad_1). Similarly, for $i = 1, \ldots, j$ we have $x_i \oplus k_2 \neq s_0$ and $x_i \oplus k_2 \neq s_1$. So, $\mathsf{P}_1^{(1)}(x_i \oplus k_1) = \mathsf{P}_1(x_i \oplus k_1)$ and $\mathsf{P}_2^{(1)}(x_i \oplus k_2) = \mathsf{P}_2(x_i \oplus k_2)$, for $i = 1, \ldots, j$. Thus

$$\overrightarrow{S}_{T_j,\mathsf{P}_1^{(1)},k} \stackrel{\text{def}}{=} \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}_1^{(1)}(x_i \oplus k_1), u_i} = \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}_1(x_i \oplus k_1), u_i} = \overrightarrow{S}_{T_j,\mathsf{P}_1,k}$$

and

$$\overrightarrow{S}_{T_j,\mathsf{P}_2^{(1)},k} \stackrel{\text{def}}{=} \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}_2^{(1)}(x_i \oplus k_2), (y_i \oplus u_i)} = \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}_2(x_i \oplus k_2), (y_i \oplus u_i)} = \overrightarrow{S}_{T_j,\mathsf{P}_2,k}.$$

The above analysis holds for $\mathsf{KSoP21}$ also considering $k_1 = k_2 = k$.

4. Therefore, for both KSoP and $\mathsf{KSoP21}$ we have

$$\begin{aligned} Q_1 = \mathsf{P}^{(1)}_{1,T_j,k} &= \overrightarrow{S}_{T_j,\mathsf{P}_1^{(1)},k} \circ \mathsf{P}_1^{(1)} = \overrightarrow{S}_{T_j,\mathsf{P}_1,k} \circ \mathsf{swap}_{\mathsf{P}_1(s_0),\mathsf{P}_1(s_1)} \circ \mathsf{P}_1 \\ &= \overrightarrow{S}_{T_j,\mathsf{P}_1,k} \circ \mathsf{swap}_{\mathsf{P}_1(x_{j+1} \oplus k_1),\mathsf{P}_1(s_1)} \circ \mathsf{P}_1 \\ &= \overrightarrow{S}_{T_j,\mathsf{P}_1,k} \circ \mathsf{swap}_{\mathsf{P}_1(x_{j+1} \oplus k_1),u_{j+1}} \circ \mathsf{P}_1 \\ &= \overrightarrow{S}_{T_{j+1},\mathsf{P}_1,k} \circ \mathsf{P}_1 \\ &= \mathsf{P}_{1,T_{j+1},k}. \end{aligned}$$

Also, for KSoP:

$$\begin{aligned}
Q_2 &= \mathsf{P}^{(1)}_{2,T_j,k} = \overrightarrow{S}_{T_j,\mathsf{P}^{(1)}_2,k} \circ \mathsf{P}^{(1)}_2 = \overrightarrow{S}_{T_j,\mathsf{P}_2,k} \circ \mathsf{swap}_{\mathsf{P}_2(s_0),\mathsf{P}_2(s_1)} \circ \mathsf{P}_2 \\
&= \overrightarrow{S}_{T_j,\mathsf{P}_2,k} \circ \mathsf{swap}_{\mathsf{P}_2(s_0),\mathsf{P}_2(x_{j+1}\oplus k_2)} \circ \mathsf{P}_2 \\
&= \overrightarrow{S}_{T_j,\mathsf{P}_2,k} \circ \mathsf{swap}_{(u_{j+1}\oplus y_{j+1}),\mathsf{P}_2(x_{j+1}\oplus k_2)} \circ \mathsf{P}_2 \\
&= \overrightarrow{S}_{T_{j+1},\mathsf{P}_2,k} \circ \mathsf{P}_2 \\
&= \mathsf{P}_{2,T_{j+1},k}.
\end{aligned}$$

and for KSoP21,

$$\begin{aligned}
Q_2 &= \mathsf{P}^{(1)}_{2,T_j,k} = \overrightarrow{S}_{T_j,\mathsf{P}^{(1)}_2,k} \circ \mathsf{P}^{(1)}_2 = \overrightarrow{S}_{T_j,\mathsf{P}_2,k} \circ \mathsf{swap}_{\mathsf{P}_2(s_0),\mathsf{P}_2(s_1)} \circ \mathsf{P}_2 \\
&= \overrightarrow{S}_{T_j,\mathsf{P}_2,k} \circ \mathsf{swap}_{\mathsf{P}_2(x_{j+1}\oplus k),\mathsf{P}_2(s_1)} \circ \mathsf{P}_2 \\
&= \overrightarrow{S}_{T_j,\mathsf{P}_2,k} \circ \mathsf{swap}_{\mathsf{P}_2(x_{j+1}\oplus k),(u_{j+1}\oplus y_{j+1})} \circ \mathsf{P}_2 \\
&= \overrightarrow{S}_{T_{j+1},\mathsf{P}_2,k} \circ \mathsf{P}_2 \\
&= \mathsf{P}_{2,T_{j+1},k}.
\end{aligned}$$

5. For KSoP, the classical query's response is always equal to $\mathsf{E}_k[\mathsf{P}_1,\mathsf{P}_2](x)$ in $\mathbf{\Sigma}'_j$. If "$\mathsf{bad}_3 = \mathsf{true}$" then obviously this happens. If bad_3 does not occur, then $x \neq s_1 \oplus k_1$ and $x \neq s_0 \oplus k_2$. Also because of the assumption of no redundant query, $x \neq s_0 \oplus k_1 = x_{j+1}$ and $x \neq s_1 \oplus k_2 = x_{j+1}$. So, $\mathsf{P}^{(1)}_1(x \oplus k_1) = \mathsf{P}_1(x \oplus k_1)$ and $\mathsf{P}^{(1)}_2(x \oplus k_2) = \mathsf{P}_2(x \oplus k_2)$.This analysis holds for KSoP21 also by considering $k_1 = k_2 = k$.

The above observations allow us to rewrite $\mathbf{\Sigma}'_j$ as an experiment equivalent to $\hat{\mathbf{H}}_j$. Thus using Eq. (10) we have

$$|\ \mathsf{Pr}[\mathcal{A}(\mathbf{\Sigma}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1]\ | \leq \frac{(q+2j)}{2^n} + 4\sqrt{\frac{p}{2^n}}. \tag{11}$$

Hence, using Eq. (8), Eq. (9) and Eq. (11) we have

$$\begin{aligned}
&|\mathsf{Pr}[\mathcal{A}(\mathbf{H}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1]\ | \\
&\leq |\ \mathsf{Pr}[\mathcal{A}(\mathbf{H}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\mathbf{\Gamma}_j) \to 1]\ | + |\ \mathsf{Pr}[\mathcal{A}(\mathbf{\Gamma}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\mathbf{\Sigma}_j) \to 1]\ | \\
&\qquad\qquad + |\ \mathsf{Pr}[\mathcal{A}(\mathbf{\Sigma}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1]\ | \\
&\leq 4\sqrt{\frac{p}{2^n}} + 4\sqrt{\frac{p}{2^n}} + \frac{(q+2j)}{2^n} + 4\sqrt{\frac{p}{2^n}} \\
&= \frac{(q+2j)}{2^n} + 12\sqrt{\frac{p}{2^n}}.
\end{aligned}$$

5 Post-Quantum Security of KSoP12

In this section, we formally state and prove the post-quantum security of the KSoP12 construction.

Theorem 2. *For any adversary $\mathcal{A}$ that makes q classical queries to* KSoP12*, i.e.,* $\mathsf{E}[\mathsf{P}]$ *and p quantum queries to* P *(and its inverse) in total, it holds that*

$$\mathbf{Adv}^{\mathtt{pq\text{-}prf}}_{\mathsf{KSoP12}}(\mathcal{A}) \leq \frac{4q^2}{2^n} + \frac{4p\sqrt{q} + 12q\sqrt{p}}{2^{n/2}}.$$

The proof follows closely the line of argument used in Theorem 1. As before, we consider a sequence of hybrid experiments indexed by j, denoted $(\mathbf{H}_j)$, in which the first j classical queries are answered using random functions. To hop between two consecutive games $\mathbf{H}_j$ and $\mathbf{H}_{j+1}$, we use an intermediate hybrid $\hat{\mathbf{H}}_j$. The analysis proceeds in two steps: we first apply the reprogramming lemma (Lemma 1) and subsequently invoke our extended resampling lemma for $r = 2$ swaps (Lemma 3). We retain all notation from the proof from Section 4.

First, we define the reprogrammed permutation corresponding to P. For any $i \in [j]$, we sequentially choose random values $u_i \in \{0,1\}^n \setminus (\mathcal{U} \cup \mathcal{Y} \cup \mathcal{P}_1 \cup \mathcal{P}_2)$, where $\mathcal{U} = \{u_1, \ldots, u_{i-1}\}$, $\mathcal{Y} = \{y_i \oplus u_1 \oplus y_1, \ldots, y_i \oplus u_{i-1} \oplus y_{i-1}\}$, $\mathcal{P}_1 = \{\mathsf{P}(x_1 \oplus k_1), \ldots, \mathsf{P}(x_i \oplus k_1)\}$, and $\mathcal{P}_2 = \{\mathsf{P}(x_1 \oplus k_2) \oplus y_i, \ldots, \mathsf{P}(x_i \oplus k_2) \oplus y_i\}$. Now, the corresponding reprogramming operator is defined as

$$\overrightarrow{S}_{T_j,\mathsf{P},k} \stackrel{\text{def}}{=} \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}(x_i \oplus k_1), u_i} \circ \mathsf{swap}_{\mathsf{P}(x_i \oplus k_2), (y_i \oplus u_i)} \,.$$

We define the reprogrammed permutation $\mathsf{P}_{T_j,k}$ by

$$\mathsf{P}_{T_j,k} = \overrightarrow{S}_{T_j,\mathsf{P},k} \circ \mathsf{P}.$$

Now, let $\mathbf{H}_0, \ldots, \mathbf{H}_q$ denote a sequence of $q+1$ hybrids, where $\mathbf{H}_j$ is defined as follows:

1. Sample $k = (k_1, k_2) \xleftarrow{\$} \{0,1\}^{2n}$.
2. Answer the first j classical queries of $\mathcal{A}$ using F and the quantum queries of $\mathcal{A}$ in the first j stages (which ends when the $(j+1)$-th classical query is made) using P.
3. Answer all subsequent classical queries from $\mathcal{A}$ using $\mathsf{E}_k[\mathsf{P}]$ and all subsequent quantum queries from $\mathcal{A}$ using $\mathsf{P}_{T_j,k}$.

Thus, we have

$$\mathbf{Adv}^{\mathtt{pq\text{-}prf}}_{\mathsf{KSoP12}}(\mathcal{A}) = \left| \Pr[\mathcal{A}(\mathbf{H}_0) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_q) \to 1] \right| . \tag{12}$$

Now for any $j \in \{0, \ldots, q\}$, $\mathbf{H}_j$ can be viewed as a string of oracle calls made by $\mathcal{A}$

$$\underbrace{\mathsf{P}, (\mathsf{F}, \mathsf{P})^j,}_{j \text{ classical queries}} \quad \underbrace{(\mathsf{E}_k[\mathsf{P}], \mathsf{P}_{T_j,k},)^{q-j}.}_{q-j \text{ classical queries}}$$

where each occurrence of F (res. $\mathsf{E}_k[\mathsf{P}]$) indicates a single classical query and each occurrence of P (res. $\mathsf{P}_{T_j,k}$) indicates the sequence of quantum primitive queries. To bound the distinguishing advantage between $\mathbf{H}_j$ and $\mathbf{H}_{j+1}$, we introduce an intermediate hybrid $\hat{\mathbf{H}}_j$ defined exactly as $\mathbf{H}_j$, except that the $(j+1)$-th classical query of $\mathcal{A}$ is answered using F rather than $\mathsf{E}_k[\mathsf{P}]$. Accordingly, all subsequent quantum queries are answered using the reprogrammed permutations $\mathsf{P}_{T_{j+1},k}$.

Now, we compare the following hybrids in pairs from top to bottom:

$$\mathbf{H}_j: \quad \underbrace{\mathsf{P}, (\mathsf{F}, \mathsf{P})^j,}_{j \text{ classical queries}} \quad \mathsf{E}_k[\mathsf{P}], \mathsf{P}_{T_j,k}, \quad \underbrace{(\mathsf{E}_k[\mathsf{P}], \mathsf{P}_{T_j,k})^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

$$\hat{\mathbf{H}}_j: \quad \underbrace{\mathsf{P}, (\mathsf{F}, \mathsf{P})^j,}_{j \text{ classical queries}} \quad \mathsf{F}, \quad \mathsf{P}_{T_{j+1},k}, \quad \underbrace{(\mathsf{E}_k[\mathsf{P}], \mathsf{P}_{T_{j+1},k})^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

$$\mathbf{H}_{j+1}: \quad \underbrace{\mathsf{P}, (\mathsf{F}, \mathsf{P})^j,}_{j \text{ classical queries}} \quad \mathsf{F}, \quad \mathsf{P}, \quad \underbrace{(\mathsf{E}_k[\mathsf{P}], \mathsf{P}_{T_{j+1},k})^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

In Lemma 7 and Lemma 8 , we establish the bounds on the distinguishability of $\mathbf{H}_j$ and $\hat{\mathbf{H}}_j$, as well as $\hat{\mathbf{H}}_j$ and $\mathbf{H}_{j+1}$.

Lemma 7. *For $j = 0, \ldots, q-1$, we have*

$$| \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_{j+1}) \to 1] | \leq 2p_{j+1}\sqrt{\frac{4(j+1)}{2^n}},$$

where p_{j+1} is the expected number of quantum queries $\mathcal{A}$ makes in the $(j+1)$-th stage of the game.

Lemma 8. *For $j = 0, \ldots, q-1$, we have*

$$| \Pr[\mathcal{A}(\mathbf{H}_j) \to 1] - \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] | \leq \frac{(2q+2j)}{2^n} + 8\sqrt{2}\sqrt{\frac{p}{2^n}},$$

where q and p are the expected number of classical and quantum queries made by $\mathcal{A}$.

Thus using these two lemmas and Eq. (12), we have

$$\begin{aligned}
\mathbf{Adv}^{\mathtt{pq\text{-}prf}}_{\mathsf{KSoP12}}(\mathcal{A}) =& | \Pr[\mathcal{A}(\mathbf{H}_0) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_q) \to 1] | \\
&\leq \sum_{j=0}^{q-1} \Big(| \Pr[\mathcal{A}(\mathbf{H}_j) \to 1] - \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] | \\
&\qquad\qquad + | \Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_{j+1}) \to 1] | \Big) \\
&\leq \sum_{j=0}^{q-1} \left(\frac{(2q+2j)}{2^n} + 8\sqrt{2}\sqrt{\frac{p}{2^n}} + 2p_{j+1}\sqrt{\frac{4(j+1)}{2^n}} \right) \\
&\leq \frac{4q^2}{2^n} + 8q\sqrt{2}\sqrt{\frac{p}{2^n}} + 4p\sqrt{\frac{q}{2^n}} \\
&\leq \frac{4q^2}{2^n} + \frac{4p\sqrt{q} + 12q\sqrt{p}}{2^{n/2}}.
\end{aligned}$$

To complete the rest of the proof, it remains to bound the distinguishing advantage between $\hat{\mathbf{H}}_j$ and $\mathbf{H}_{j+1}$, as well as between $\mathbf{H}_j$ and $\hat{\mathbf{H}}_j$. Proofs of the above two lemmas are deferred to Section A.1 and Section A.2.

6 Key Recovery Attack on KSoP and Its Siblings

In this section, we present Q1 key-recovery attacks on KSoP, and its variants, following the offline-Simon attack strategy of Bonnetain et al. [7]. In an offline-Simon attack, the adversary first gathers classical data from the target construction and then locally builds a quantum oracle with a hidden XOR-periodicity. The attack proceeds as follows:

Step 1. Split the secret key into two parts: one part is guessed using Grover search, and the other is recovered using Simon's algorithm in Q1.

Step 2. Define functions F and G such that $H(x) = F(\text{guess}, x) \oplus G(x)$ has a hidden period for the correct key guess.

Step 3. Collect classical evaluations of G and construct H locally in the Q1 setting.

Step 4. Use Grover search over the guessed key part and apply Simon's algorithm in Q1, to recover the hidden period.

Step 5. Combine the recovered information to reconstruct and verify the full secret key.

6.1 Key Recovery Attack on KSoP

Consider u such that $0 \leq u \leq n$. We express k_1 as $k_1^l||k_1^r$ where $k_1^l \in \{0,1\}^u$ and $k_1^r \in \{0,1\}^{n-u}$. Similarly, we express k_2 as $k_2^l||k_2^r$. Overall key $k = (k_1, k_2)$ is of size $2n$.

We consider two functions, F and G, which are constructed based on the KSoP encryption and permutations as follows:

$$F : \{0,1\}^{n-u} \times \{0,1\}^n \times \{0,1\}^u \to \{0,1\}^n,\ G : \{0,1\}^u \to \{0,1\}^n,$$

$$F(i_1, i_2, x) := \mathsf{P}_1(x||i_1) \oplus \mathsf{P}_2(x||0^{n-u} \oplus i_2),\ G(x) := \mathsf{KSoP}(x||0^{n-u})\,.$$

Observe that $F(i_1, i_2, x) \oplus G(x)$ has period k_1^l when $i_1 = k_1^r$ and $i_2 = (k_1^l \oplus k_2^l)||k_2^r$:

$$\begin{aligned}
&F(k_1^r, (k_1^l \oplus k_2^l)||k_2^r, x) \oplus G(x)\\
&= \mathsf{P}_1(x||k_1^r) \oplus \mathsf{P}_2(x||0^{n-u} \oplus (k_1^l \oplus k_2^l)||k_2^r) \oplus \mathsf{KSoP}(x||0^{n-u})\\
&= \mathsf{P}_1(x||k_1^r) \oplus \mathsf{P}_2(x \oplus k_1^l \oplus k_2^l||k_2^r) \oplus \mathsf{P}_1(x||0^{n-u} \oplus k_1) \oplus \mathsf{P}_2(x||0^{n-u} \oplus k_2)\\
&= \mathsf{P}_1(x||k_1^r) \oplus \mathsf{P}_2(x \oplus k_1^l \oplus k_2^l||k_2^r) \oplus \mathsf{P}_1(x \oplus k_1^l||k_2^r) \oplus \mathsf{P}_2(x \oplus k_2^l||k_2^r)\\
&= \mathsf{P}_1(x \oplus k_1^l||k_2^r) \oplus \mathsf{P}_2(x \oplus k_2^l||k_2^r) \oplus \mathsf{P}_1(x||k_1^r) \oplus \mathsf{P}_2(x \oplus k_1^l \oplus k_2^l||k_2^r)\\
&= F(k_1^r, (k_1^l \oplus k_2^l)||k_2^r, x \oplus k_1^l) \oplus G(x \oplus k_1^l)\,.
\end{aligned}$$

Now we can recover a part of the key $k = (k_1, k_2)$ using Grover's search and rest of it using Simon's algorithm in Q1 with a high probability by performing $D = \mathcal{O}(2^u)$ classical queries to KSoP and making $T = \mathcal{O}(n^3.2^{(2n-u)/2})$ time. The tradeoff is $DT^2 = \tilde{\mathcal{O}}(2^{2n})$, which balances at $T = D = \tilde{\mathcal{O}}(2^{2n/3})$. Besides, our attack requires $\mathcal{O}(n)$ classical memory and poly(n) many qubits.

The detailed description of the key-recovery attack is identical to the one used in [7]. For completeness, we provide the details below.

Remark 2. In [17], Sun et al. present key-recovery attacks on permutation-based PRFs, as classified by Chen et al. [8]. In particular, they provide a generic key-recovery attack against two-permutation-based PRFs with complexity $O(2^{2n/3})$. Although KSoP belongs to this category, we independently develop the above dedicated key-recovery attack tailored specifically to the KSoP construction.

6.1.1 Detailed Analysis of the KSoP Attack.

We first prepare the quantum state

$$|\psi_G\rangle = \bigotimes_{j=1}^{c} \sum_{x_j \in \{0,1\}^u} |x_j\rangle\,|G(x_j)\rangle\,.$$

To construct this state, we define a quantum oracle U_p. For each $p \in [2^{u-1}]$, we first obtain the value $G(p)$ by making a classical query to KSoP. Using this known value $G(p)$, we define the oracle U_p as follows:

$$|p\rangle \bigotimes_{j=1}^{c} |x_j\rangle\,|y\rangle \to \begin{cases} |p\rangle \bigotimes_{j=1}^{c} |x_j\rangle\,|y \oplus G(x_j)\rangle\,, & \text{if } x_1 = x_2 = \cdots = x_c = p,\\ |p\rangle \bigotimes_{j=1}^{c} |x_j\rangle\,|y\rangle\,, & \text{otherwise.}\end{cases}$$

We use Algorithm 2 to prepare the state $|\psi_G\rangle$. Then, we define another quantum oracle $O_F : |(i_1, i_2, x)\rangle \mapsto |F(i_1, i_2, x)\rangle$. Using this oracle, we apply Algorithm 3 to recover $k_1^r||(k_1^l \oplus k_2^l)||k_2^r$. Now, define the function $f : \{0,1\}^u \to \{0,1\}^n$, where

$$f(x) = F(k_1^r, (k_1^l \oplus k_2^l)||k_2^r, x) \oplus G(x).$$

We then apply Algorithm 4, namely Simon's algorithm in the Q1 setting. Under the assumption that $\ell = \mathcal{O}(n)$, the algorithm returns k_1^l, which is the period of the function f.

Algorithm 2 Preparation of $|\psi_G\rangle$

Input: Classical query access to KSoP and an integer c.
Output: The state $|\psi_G\rangle$.

1: Initialize the registers as
$$|0\rangle^{\otimes u} |0\rangle^{\otimes cu} |0\rangle^{\otimes cn} .$$
2: Apply the Hadamard transform $H^{\otimes cu}$ on the second register to create the uniform superposition
$$|\phi\rangle = \bigotimes_{j=1}^{c} \sum_{x_j \in \{0,1\}^u} |x_j\rangle |0\rangle .$$
3: **for** $p = 0, 1, \ldots, 2^u - 1$ **do**
4: Prepare the basis state $|p\rangle$ in the first register.
5: Obtain the value $G(p)$ using classical queries to KSoP.
6: Apply the oracle U_p on the state $|p\rangle \otimes |\phi\rangle$.
7: Reset the first register by uncomputing $|p\rangle$.
8: **return** $|\psi_G\rangle$.

Complexity Analysis. We now analyze the complexity of the algorithms.

Analysis of Algorithm 2: In Algorithm 2, the state $|\psi_G\rangle$ can be prepared using only a small number of quantum gates. Each implementation of U_p requires one classical query to compute $G(x)$. One evaluation of $G(x)$ can be obtained with $\mathcal{O}(1)$ evaluations of KSoP. Therefore, Algorithm 2 prepares $|\psi_G\rangle$ using $\mathcal{O}(2^u)$ classical queries and $\mathcal{O}(u + c(n + u))$ qubits.

Analysis of Algorithm 3: The analysis consists of an online phase and an offline phase. Step 2 belongs to the online phase, where $|\psi_G\rangle$ is prepared. As shown by Bonnetain et al. [7], the same state $|\psi_G\rangle$ can be reused in every Grover iteration. Steps 3-9 form the offline phase. In particular, steps 4-8 implement the *test* oracle:

$$|i\rangle |\psi_G\rangle |b\rangle \mapsto \begin{cases} |i\rangle |\psi_G\rangle |b \oplus 1\rangle , & \text{if } F(i,x) \oplus G(x) \text{ is periodic function,} \\ |i\rangle |\psi_G\rangle |b \oplus 0\rangle , & \text{otherwise,} \end{cases}$$

where $i \in \{0,1\}^{2n-u}$ and can also be written as $i = i_1 || i_2$ with $i_1 \in \{0,1\}^{n-u}$ and $i_2 \in \{0,1\}^n$. Overall, we make $2c$ quantum queries to the oracle $F(i,x)$. In addition, step 6 requires $\mathcal{O}(n^3)$ time for the computation. Hence, Algorithm 3 recovers $k_1^r || (k_1^l \oplus k_2^l) || k_2^r$ using $\mathcal{O}(2^u)$ classical queries and $\mathcal{O}\left((n^3 + 2c)2^{\frac{(2n-u)}{2}}\right)$ time.

Analysis of Algorithm 4: Here we apply Algorithm 4 to the function $f(x) = F(k_1^r, (k_1^l \oplus k_2^l) || k_2^r, x) \oplus G(x)$. This allows us to recover k_1^l using $\mathcal{O}(2^u)$ classical queries, $\mathcal{O}(u^3 + u)$ time, and $\mathcal{O}(u^2)$ qubits. According to Theorem 2 of [6], choosing $c = n + \tau + 1$ (where τ is a positive integer) is sufficient to compute the unique $k_1^r, (k_1^l \oplus k_2^l) || k_2^r$. By applying Simon's algorithm in the Q1 model to the function $F(k_1^r, (k_1^l \oplus k_2^l) || k_2^r, x) \oplus G(x)$, we can recover k_1^l using $\mathcal{O}(2^u)$ classical queries, $\mathcal{O}(u^3)$ time, and poly(u) qubits.

6.2 Key Recovery Attack on KSoP21

In this subsection, we discuss a key recovery attack on KSoP21 under the Q1 model. Again, we follow the strategy of Bonnetain et.al. Consider u such that $0 \leq u \leq n$. We express k

Algorithm 3 Recover $k_1^r||(k_1^l \oplus k_2^l)||k_2^r$ using classical queries

Input: Oracle $O_F : |i_1, i_2, x\rangle \mapsto |F(i_1, i_2, x)\rangle$ with $F : \{0,1\}^{2n-u} \times \{0,1\}^u \to \{0,1\}^n$, and classical query access to the KSoP construction, and an integer c.
Output: $k_1^r||(k_1^l \oplus k_2^l)||k_2^r$.

1: Initialize the state: $|0\rangle^{\otimes(2n-u)} |0\rangle^{\otimes cu} |0\rangle^{\otimes cn} |1\rangle$.
2: Apply Algorithm 2 to prepare

$$|\psi_G\rangle = \bigotimes_{j=1}^{c} \sum_{x_j \in \{0,1\}^u} |x_j\rangle |G(x_j)\rangle .$$

3: Apply $H^{\otimes(2n-u)} \otimes I_u^{\otimes c} \otimes I_n^{\otimes c} \otimes H$ to obtain $\sum_{i \in \{0,1\}^{2n-u}} |i\rangle \otimes |\psi_G\rangle |b\rangle$, where I_n is the identity operator on n qubits and $|b\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.
4: Query the oracle O_F exactly c times:

$$\sum_{i \in \{0,1\}^{2n-u}} |i\rangle \bigotimes_{j=1}^{c} \sum_{x_j \in \{0,1\}^u} |x_j\rangle |G(x_j) \oplus F(i, x_j)\rangle = \sum_{i \in \{0,1\}^{2n-u}} |i\rangle \otimes |\psi_{G \oplus F}\rangle |b\rangle .$$

5: Apply the transformation $I_{2n-u} \otimes H^{\otimes cu} \otimes I_n^{\otimes c} \otimes I_1$:

$$\sum_{i \in \{0,1\}^{2n-u}} |i\rangle \bigotimes_{j=1}^{c} \sum_{x_j, y_j \in \{0,1\}^u} (-1)^{x_j \cdot y_j} |y_j\rangle |G(x_j) \oplus F(i, x_j)\rangle |b\rangle .$$

6: Compute the dimension d of the vector space generated by $y_1, y_2, \ldots, y_c$. If $d = u$, set $r = 0$; otherwise, if $d < u$, set $r = 1$. Add r to b, and then uncompute both d and r to get

$$\sum_{i \in \{0,1\}^{2n-u}} |i\rangle \bigotimes_{j=1}^{c} \sum_{x_j, y_j \in \{0,1\}^u} (-1)^{x_j \cdot y_j} |y_j\rangle |G(x_j) \oplus F(i, x_j)\rangle |b \oplus r\rangle .$$

7: Apply the Hadamard gates again: $I_{2n-u} \otimes H^{\otimes cu} \otimes I_n^{\otimes c} \otimes I_1$.
8: Query the oracle O_F another c times: $\sum_{i \in \{0,1\}^{2n-u}} |i\rangle \otimes |\psi_G\rangle |b \oplus r\rangle$.
9: Perform $O(2^{(2n-u)/2})$ Grover iterations and measure the state to recover $k_1^r||(k_1^l \oplus k_2^l)||k_2^r$.

Algorithm 4 Simon's algorithm using classical queries

Input: Classical query access to the periodic function $f(x) : \{0,1\}^u \to \{0,1\}^n$
Output: s

1: Perform 2^u classical queries to $f(x)$ to prepare the quantum state

$$|\psi_f\rangle = \bigotimes_{j=1}^{\ell} \sum_{x\in\{0,1\}^u} |x\rangle\,|f(x)\rangle\,.$$

2: Perform Hadamard gates H^n on each state $|x\rangle$ to obtain

$$\bigotimes_{j=1}^{\ell} \sum_{x,v\in\{0,1\}^u} (-1)^{x\cdot v}\,|v\rangle\,|f(x)\rangle\,.$$

3: Measure all $|v\rangle$ registers to obtain ℓ vectors $v_1, v_2, \ldots, v_\ell$.
4: Compute the dimension

$$d = \dim\big(\mathrm{Span}(v_1, v_2, \ldots, v_\ell)\big).$$

If $d \neq u-1$, return $\perp$. If $d = u-1$, compute a nonzero vector

$$s \in \{0,1\}^u \quad \text{such that} \quad s \perp \mathrm{Span}(v_1, v_2, \ldots, v_\ell),$$

using classical linear algebra.

express as $k^l||k^r$ where $k^l \in \{0,1\}^u$ and $k^r \in \{0,1\}^{n-u}$. Overall key $k = (k^l||k^r)$ is of size n.

We consider two functions, F and G, which are constructed based on the KSoP21 encryption and permutations as follows:

$$F : \{0,1\}^{n-u} \times \{0,1\}^u \to \{0,1\}^n,\ G : \{0,1\}^u \to \{0,1\}^n,$$

$$F(i,x) := \mathsf{P}_1(x||i) \oplus \mathsf{P}_2(x||i),\ G(x) := \mathsf{KSoP21}(x||0^{n-u}).$$

Observe that $F(i,x) \oplus G(x)$ has period k^l when $i = k^r$:

$$\begin{aligned}
&F(k^r, x) \oplus G(x)\\
&= \mathsf{P}_1(x||k^r) \oplus \mathsf{P}_2(x||k^r) \oplus \mathsf{KSoP21}(x||0^{n-u})\\
&= \mathsf{P}_1(x||k^r) \oplus \mathsf{P}_2(x||k^r) \oplus \mathsf{P}_1((x\oplus k^l)||k^r) \oplus \mathsf{P}_2((x\oplus k^l)||k^r)\\
&= \mathsf{P}_1((x\oplus k^l)||k^r) \oplus \mathsf{P}_2((x\oplus k^l)||k^r) \oplus \mathsf{P}_1(x||k^r) \oplus \mathsf{P}_2(x||k^r)\\
&= F(k^r, x\oplus k^l) \oplus G(x\oplus k^l)\,.
\end{aligned}$$

Now we can recover a part of the key k using Grover's search and the rest of it using Simon's algorithm in Q1 with a high probability by performing $D = \mathcal{O}(2^u)$ classical queries to KSoP21 and making $T = \mathcal{O}(n^3 2^{(n-u)/2})$ time. The tradeoff is $DT^2 = \tilde{\mathcal{O}}(2^n)$, which balances at $T = D = \tilde{\mathcal{O}}(2^{n/3})$. Besides, our attack requires $\mathcal{O}(n)$ classical memory and poly(n) many qubits.

6.3 Key Recovery Attack on KSoP12

For full key recovery attack on KSoP12, we consider u such that $0 \le u \le n$ and we express k_1 as $k_1^l||k_1^r$ where $k_1^l \in \{0,1\}^u$ and $k_1^r \in \{0,1\}^{n-u}$. Similarly we express k_2 as $k_2^l||k_2^r$. Overall key $k = (k_1, k_2)$ is of size $2n$.

We consider two functions, F and G, which are constructed based on the $\mathsf{KSoP12}$ encryption and permutations as follows:

$$F : \{0,1\}^{n-u} \times \{0,1\}^n \times \{0,1\}^u \to \{0,1\}^n,\ G : \{0,1\}^u \to \{0,1\}^n,$$

$$F(i_1, i_2, x) := \mathsf{P}(x||i_1) \oplus \mathsf{P}(x||0^{n-u} \oplus i_2),\ G(x) := \mathsf{KSoP12}(x||0^{n-u})\,.$$

Observe that $F(i_1, i_2, x) \oplus G(x)$ has period k_1^l when $i_1 = k_1^r$ and $i_2 = (k_1^l \oplus k_2^l)||k_2^r$:

$$\begin{aligned}
&F(k_1^r, (k_1^l \oplus k_2^l)||k_2^r, x) \oplus G(x)\\
&= \mathsf{P}(x||k_1^r) \oplus \mathsf{P}(x||0^{n-u} \oplus (k_1^l \oplus k_2^l)||k_2^r) \oplus \mathsf{KSoP12}(x||0^{n-u})\\
&= \mathsf{P}(x||k_1^r) \oplus \mathsf{P}(x \oplus k_1^l \oplus k_2^l||k_2^r) \oplus \mathsf{P}(x||0^{n-u} \oplus k_1) \oplus \mathsf{P}(x||0^{n-u} \oplus k_2)\\
&= \mathsf{P}(x||k_1^r) \oplus \mathsf{P}(x \oplus k_1^l \oplus k_2^l||k_2^r) \oplus \mathsf{P}(x \oplus k_1^l||k_2^r) \oplus \mathsf{P}(x \oplus k_2^l||k_2^r)\\
&= \mathsf{P}(x \oplus k_1^l||k_2^r) \oplus \mathsf{P}(x \oplus k_2^l||k_2^r) \oplus \mathsf{P}(x||k_1^r) \oplus \mathsf{P}(x \oplus k_1^l \oplus k_2^l||k_2^r)\\
&= F(k_1^r, (k_1^l \oplus k_2^l)||k_2^r, x \oplus k_1^l) \oplus G(x \oplus k_1^l)\,.
\end{aligned}$$

Now we can recover a part of the key $k = (k_1, k_2)$ using Grover's search and rest of it using Simon's algorithm in Q1 with a high probability by performing $D = \mathcal{O}(2^u)$ classical queries to $\mathsf{KSoP12}$ and making $T = \mathcal{O}(n^3 2^{(2n-u)/2})$ time. The tradeoff is $DT^2 = \tilde{\mathcal{O}}(2^{2n})$, which balances at $T = D = \tilde{\mathcal{O}}(2^{2n/3})$. Besides, our attack requires $O(n)$ classical memory and poly(n) many qubits.

7 A Discussion on Keyed-Sum of r-many Permutations

In this section, we consider the Keyed Sum of r-many Permutations with identical key. In line with the nomenclature we considered till now, we call this $\mathsf{KSoPr1}$. Let $\mathsf{P}_1, \ldots, \mathsf{P}_r \in \mathcal{P}(n)$ be r independent random permutations. The $\mathsf{KSoPr1}$ construction is defined as,

$$\mathsf{KSoPr1}_k[\mathsf{P}_1, \ldots, \mathsf{P}_r] := \mathsf{P}_1(x \oplus k) \oplus \cdots \oplus \mathsf{P}_r(x \oplus k).$$

Theorem 3. *For any adversary $\mathcal{A}$ that makes q classical queries to $\mathsf{KSoPr1}$, i.e., $\mathsf{E}[\mathsf{P}_1, \ldots, \mathsf{P}_r]$ and p quantum queries combined to $\mathsf{P}_1, \ldots, \mathsf{P}_r$ (and their inverse) in total, it holds that*

$$\mathbf{Adv}^{\mathtt{pq\text{-}prf}}_{\mathsf{KSoPr1}}(\mathcal{A}) = \mathcal{O}\left(\frac{q^2 r}{2^n} + \frac{(p\sqrt{rq} + rq\sqrt{p})}{2^{n/2}}\right).$$

Now, we discuss a brief proof sketch for the above theorem. Let $\mathcal{A}$ be a distinguisher and the goal of $\mathcal{A}$ is to distinguish between the *real world* ($\mathsf{E}[\mathsf{P}_1, \ldots, \mathsf{P}_r]$, P_1,…,P_r) and the *ideal world* ($\mathsf{F}, \mathsf{P}_1, \ldots, \mathsf{P}_r$), where F is a uniform random function. In the real world (resp. ideal world), the adversary $\mathcal{A}$ has *classical* access to $\mathsf{E}[\mathsf{P}_1, \ldots, \mathsf{P}_r]$ (resp. F). In both the worlds, the adversary has *quantum* access to the permutations $\mathsf{P}_1, \ldots, \mathsf{P}_r$ in both the forward and inverse directions.

Let T_j represent the list of the first j classical construction queries. To analyse the security of $\mathsf{KSoPr1}$ using the hybrid reprogramming technique, we need to define the corresponding reprogramming operators. For any $i \in \{1, \ldots, r-1\}$, we consider a set of random variables $\{u_{i,1}, \ldots, u_{i,j}\}$ and define the reprogramming operators

$$S_{T_j, \mathsf{P}_i, k} := \prod_{\ell=1}^{j} \mathsf{swap}_{\mathsf{P}_i(x_\ell \oplus k), u_{i,\ell}},\ S_{T_j, \mathsf{P}_r, k} := \prod_{\ell=1}^{j} \mathsf{swap}_{\mathsf{P}_r(x_\ell \oplus k), (y_\ell \oplus u_{1,\ell} \oplus \cdots \oplus u_{r-1,\ell})}.$$

The reprogrammed permutations are defined as

$$\mathsf{P}_{i,T_j,k} := S_{T_j, \mathsf{P}_i, k} \circ \mathsf{P}_i, \forall i \in [1, \ldots, r-1] \text{ and } \mathsf{P}_{r,T_j,k} := S_{T_j, \mathsf{P}_r, k} \circ \mathsf{P}_r.$$

Note that we need to sample the values of the random variables $u_{i,j}$, maintaining the permutation compatibility of the reprogrammed versions. Analogously, following the proof in Section 4, we consider the following steps.

Step 1: For any $j \in 0, \ldots, q$, define the hydrids $\mathbf{H}_j$, respectively define intermediate hybrids $\hat{\mathbf{H}}_j$.

Step 2: For $j = 0, \ldots, q-1$, we determine the distinguishing advantage between $\hat{\mathbf{H}}_j$ and $\mathbf{H}_{j+1}$ by applying Lemma 1.

Step 3: For $j = 0, \ldots, q-1$, we determine the distinguishing advantage between $\mathbf{H}_j$ and $\hat{\mathbf{H}}_j$. While distinguishing $\mathbf{H}_j$ and $\hat{\mathbf{H}}_j$, for every $i \in \{1, \ldots, r\}$, we need to introduce hybrids $\Gamma_{\mathsf{i,j}}$ (to play the ARS-game for the permutation P_i). Clearly, the distinguishing advantage between $\mathbf{H}_j$ & $\Gamma_{\mathsf{1,j}}$ can be determined by applying Lemma 2. Similarly for each $i \in \{1, \ldots, r-1\}$, the distinguishing advantage between $\Gamma_{\mathsf{i,j}}$ & $\Gamma_{\mathsf{i+1,j}}$ is also determined by applying Lemma 2.

Step 4: For $j = 0, \ldots, q-1$, considering proper bad events, we determine the distinguishing advantage between $\Gamma_{\mathsf{r,j}}$ and $\hat{\mathbf{H}}_j$.

Thus, by carefully carrying out the steps and finding distinguishing advantages between respective hybrids, as described above, and by repeating the same strategy employed in the proof of Section 4, one can derive the security bound.

8 Conclusion

In this paper, we studied the post-quantum security of the keyed sum of permutations construction and its related variants, including the single-permutation and single-key versions. We proved that all these constructions achieve $n/3$-bit security in the Q1 model. In addition, we presented a key-recovery attack on KSoP21 with data and time complexity $O(2^{n/3})$, thereby establishing its tight post-quantum security. For KSoP and KSoP12, we showed that key-recovery attacks can be carried out with data and time complexity $O(2^{2n/3})$. Determining tight security bounds for these two constructions remains an open and interesting problem. We also believe that our tool for handling resampling at multiple points of a single underlying permutation can be applied to obtain the security bounds for SoEM12. Another promising direction is the analysis of the post-quantum security of the domain-separated single-key, single-permutation variant of KSoP.

Acknowledgement. We would like to thank the anonymous reviewers of Asiacrypt 2026 and ToSC 2026/2027 for helpful remarks which improved the paper.

References

[1] Alagic, G., Bai, C., Katz, J., Majenz, C.: Post-quantum security of the even-mansour cipher. IACR Cryptol. ePrint Arch. **2021/1601** (2021), https://eprint.iacr.org/2021/1601

[2] Alagic, G., Bai, C., Katz, J., Majenz, C.: Post-quantum security of the even-mansour cipher. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 458–487. Springer (2022), https://doi.org/10.1007/978-3-031-07082-2_17

[3] Alagic, G., Bai, C., Katz, J., Majenz, C., Struck, P.: Post-quantum security of tweakable even-mansour, and applications. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I. Lecture Notes in Computer Science, vol. 14651, pp. 310–338. Springer (2024), https://doi.org/10.1007/978-3-031-58716-0_11

[4] Bai, C., Esmaili, M., Mantri, A.: Quantum security analysis of the key-alternating ciphers. Cryptology ePrint Archive, Paper 2025/945 (2025), https://eprint.iacr.org/2025/945

[5] Basak, J., Bhaumik, R., Chauhan, A.K., Jejurikar, R., Jha, A., Roy, A., Schrottenloher, A., Talnikar, S.: Post-quantum security of key-alternating feistel ciphers. In: Hanaoka, G., Yang, B. (eds.) Advances in Cryptology - ASIACRYPT 2025 - 31st International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, VIC, Australia, December 8-12, 2025, Proceedings, Part I. Lecture Notes in Computer Science, vol. 16245, pp. 446–478. Springer (2025), https://doi.org/10.1007/978-981-95-5018-0_15

[6] Bonnetain, X.: Tight bounds for simon's algorithm. In: Longa, P., Ràfols, C. (eds.) Progress in Cryptology - LATINCRYPT 2021 - 7th International Conference on Cryptology and Information Security in Latin America, Bogotá, Colombia, October 6-8, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12912, pp. 3–23. Springer (2021), https://doi.org/10.1007/978-3-030-88238-9_1

[7] Bonnetain, X., Hosoyamada, A., Naya-Plasencia, M., Sasaki, Y., Schrottenloher, A.: Quantum attacks without superposition queries: The offline simon's algorithm. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11921, pp. 552–583. Springer (2019), https://doi.org/10.1007/978-3-030-34578-5_20

[8] Chen, Y.L., Lambooij, E., Mennink, B.: How to build pseudorandom functions from public random permutations. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11692, pp. 266–293. Springer (2019), https://doi.org/10.1007/978-3-030-26948-7_10

[9] Grilo, A.B., Hövelmanns, K., Hülsing, A., Majenz, C.: Tight adaptive reprogramming in the QROM. In: Tibouchi, M., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13090, pp. 637–667. Springer (2021), https://doi.org/10.1007/978-3-030-92062-3_22

[10] Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Miller, G.L. (ed.) Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996. pp. 212–219. ACM (1996), https://doi.org/10.1145/237814.237866

[11] Jaeger, J., Song, F., Tessaro, S.: Quantum key-length extension. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part I. Lecture Notes in

Computer Science, vol. 13042, pp. 209–239. Springer (2021), https://doi.org/10.1007/978-3-030-90459-3_8

[12] Li, Z., Fan, S., Gao, F., Hao, Y., Sun, H., Hu, X., Li, D.: Quantum attacks on sum of even-mansour construction utilizing online classical queries. EPJ Quantum Technology **12**(1), 67 (2025), https://doi.org/10.1140/epjqt/s40507-025-00374-x

[13] Shiba, R., Iwata, T.: Multi-key security in the quantum world: Revisiting tweakable even-mansour and FX. IACR Trans. Symmetric Cryptol. **2026**(2) (2026), https://doi.org/10.46586/tosc.abksudsuurh6

[14] Shiba, R., Iwata, T.: Post-quantum security of tweakable key-alternating feistel ciphers in the multi-key setting. Cryptology ePrint Archive, Paper 2026/1312 (2026), https://eprint.iacr.org/2026/1312

[15] Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: 35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994. pp. 124–134. IEEE Computer Society (1994), https://doi.org/10.1109/SFCS.1994.365700

[16] Sibleyras, F., Todo, Y.: Keyed sum of permutations: A simpler rp-based PRF. In: Rosulek, M. (ed.) Topics in Cryptology - CT-RSA 2023 - Cryptographers' Track at the RSA Conference 2023, San Francisco, CA, USA, April 24-27, 2023, Proceedings. Lecture Notes in Computer Science, vol. 13871, pp. 573–593. Springer (2023), https://doi.org/10.1007/978-3-031-30872-7_22

[17] Sun, H., Gao, F., Xu, R., Li, D., Li, Z., Zhang, K.: Quantum key-recovery attacks on permutation-based pseudorandom functions. IEEE Internet Things J. **12**(17), 35692–35704 (2025), https://doi.org/10.1109/JIOT.2025.3579024

[18] Tan, Y., Gao, J., Li, X.: Post-quantum security of the sum of even-mansour. Cryptology ePrint Archive, Paper 2025/2274 (2025), https://eprint.iacr.org/2025/2274

Supplementary Materials

A Deferred Proofs

A.1 Proof of Lemma 7

Recall, we can have the following string of oracle call representations for the two hybrids.

$$\hat{\mathbf{H}}_j: \underbrace{\mathsf{P}, (\mathsf{F}, \mathsf{P})^j,}_{j \text{ classical queries}} \quad \mathsf{F}, \quad \mathsf{P}_{T_{j+1},k}, \underbrace{(\mathsf{E}_k[\mathsf{P}], \mathsf{P}_{T_{j+1},k})^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

$$\mathbf{H}_{j+1}: \underbrace{\mathsf{P}, (\mathsf{F}, \mathsf{P})^j,}_{j \text{ classical queries}} \quad \mathsf{F}, \quad \mathsf{P}, \quad \underbrace{(\mathsf{E}_k[\mathsf{P}], \mathsf{P}_{T_{j+1},k})^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

It is clear that the two hybrids are identical, except for the $(j+1)$-th stages, where $\hat{\mathbf{H}}_j$ uses reprogrammed versions of the permutation P, and $\mathbf{H}_{j+1}$ uses the original permutation P. We will show that the distinguishing advantage of these two hybrids can be captured by the *reprogramming* bounds of the permutations.

More formally, given the adversary $\mathcal{A}$, we construct an ARP distinguisher $\mathcal{D}$ as follows (note that Phases $1, 2, 3$ correspond to Phases $1, 2, 3$ in the ARP game respectively).

Phase 1. For all the queries up to and including the $(j+1)$-th classical query:

1. $\mathcal{D}$ samples a uniform random permutation P from $\mathcal{P}(n)$ and a uniform random function F from $\mathcal{F}(n,n)$.
2. All classical queries from $\mathcal{A}$ are answered using F. Let the transcript of classical queries and responses be denoted by $T_{j+1} = \{(x_1, y_1), \ldots, (x_{j+1}, y_{j+1})\}$.
3. All quantum queries from $\mathcal{A}$ are answered using P. More formally, $\mathcal{D}$ defines $F(t,x) := \mathsf{P}^t(x)$, for $t \in \{1, -1\}$.
4. $\mathcal{D}$ defines the randomized algorithm $\mathcal{B}$, which takes a key k, computes a set B of input/output pairs to be reprogrammed so that $F^{(B)}(t,x) = \mathsf{P}^t_{T_{j+1},k}(x)$, for all t, x.

Phase 2. For all the quantum queries in the $(j+1)$-th stage:

1. $\mathcal{B}$ is run to generate the set B and $\mathcal{D}$ is given quantum access to F_b, where $F_0 = F(= \mathsf{P}^t)$ and $F_1 = F^{(B)}(= \mathsf{P}^t_{T_{j+1},k})$.
2. All quantum queries from $\mathcal{A}$ are answered using F_b.

Phase 3. For all the queries beyond and including $(j+2)$-th classical query:

1. $\mathcal{D}$ is given the randomness (specifically the key k here) used by $\mathcal{B}$ to generate B.
2. All classical queries from $\mathcal{A}$ are answered using $\mathsf{E}_k[\mathsf{P}]$.
3. All quantum queries from $\mathcal{A}$ are answered using F_1.
4. Finally, $\mathcal{D}$ returns whatever $\mathcal{A}$ returns.

It is clear that $\mathcal{D}$ is a valid ARP distinguisher. If $b = 0$ (i.e., $\mathcal{D}$'s oracle in Phase 2 is $F_0 = F$), then the output distribution of $\mathcal{A}$ is identical to its output distribution in $\mathbf{H}_{j+1}$. On the other hand, if $b = 1$ (i.e., $\mathcal{D}$'s oracle in Phase 2 is $F_1 = F^{(B)}$), then the output of $\mathcal{A}$ is identically distributed to its output in $\hat{\mathbf{H}}_j$. It follows that $|\Pr[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{H}_{j+1}) \to 1]$ is equal to the distinguishing advantage of $\mathcal{D}$ in the reprogramming game. To bound this quantity using Lemma 1, we bound the reprogramming probability ϵ and the expected number of queries made by $\mathcal{D}$ in Phase 2 (when $F_0 = F$).

The reprogramming probability ϵ can be bounded using the definition of $\mathsf{P}_{T_{j+1},k}$ and the fact that $F^{(B)}(t,x) = \mathsf{P}^t_{T_{j+1},k}(x)$. Fixing P and T_{j+1}, the probability that any given (t,x) is reprogrammed is at most the probability (over k) that it is in the set:

$$B_1 = \{(1,(x_i \oplus k_1)),(-1,\mathsf{P}^{-1}(u_i)),(1,(x_i \oplus k_2)),(-1,\mathsf{P}^{-1}(y_i \oplus u_i))\}_{i=1}^{j+1}.$$

Here, for any (t,x), reprogramming occurs only if (t,x) coincides with one of the $4(j+1)$ reprogrammed points. So, the probability that (t,x) is reprogrammed is upper bounded by $4(j+1)/2^n$. Hence, we get $\epsilon \leq 4(j+1)/2^n$.

The expected number of queries made by $\mathcal{D}$ in `Phase 2` when $F_0 = F$ is equal to the expected number of queries made by $\mathcal{A}$ in its $(j+1)$-th stage, which is p_{j+1}. Thus, by Lemma 1, we have

$$|\ \mathsf{Pr}[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\mathbf{H}_{j+1}) \to 1]\ | \leq 2p_{j+1}\sqrt{\frac{4(j+1)}{2^n}}.$$

A.2 Proof of Lemma 8

For bounding the distinguishing advantage between $\mathbf{H}_j$ and $\hat{\mathbf{H}}_j$, we will introduce an intermediate hybrid called $\mathbf{\Sigma}_j$. Now, we bound the distinguishing advantages between the following hybrids in pairs from top to bottom.

$$\mathbf{H}_j: \underbrace{\mathsf{P},(\mathsf{F},\mathsf{P})^j,}_{j \text{ classical queries}} \mathsf{E}_k[\mathsf{P}],\quad \mathsf{P}_{T_j,k},\quad \underbrace{(\mathsf{E}_k[\mathsf{P}],\mathsf{P}_{T_j,k})^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

$$\mathbf{\Sigma}_j: \underbrace{\mathsf{P},(\mathsf{F},\mathsf{P})^j,}_{j \text{ classical queries}} \mathsf{E}_k[\mathsf{P}^{(1)}],\ \mathsf{P}^{(1)}_{T_j,k},\ \underbrace{(\mathsf{E}_k[\mathsf{P}^{(1)}],\mathsf{P}^{(1)}_{T_j,k})^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

$$\hat{\mathbf{H}}_j: \underbrace{\mathsf{P},(\mathsf{F},\mathsf{P})^j,}_{j \text{ classical queries}} \mathsf{F},\quad \mathsf{P}_{T_{j+1},k},\ \underbrace{(\mathsf{E}_k[\mathsf{P}],\mathsf{P}_{T_{j+1},k},)^{q-j-1}.}_{q-j-1 \text{ classical queries}}$$

Let, $\mathcal{A}$ be an adversary of $\mathbf{H}_j$ and $\mathbf{\Sigma}_j$. From $\mathcal{A}$, we can construct an ARS distinguisher $\mathcal{D}$ as follows.

Phase 1. For all the queries up to (immediately before) the $(j+1)$-th classical query:

1. $\mathcal{D}$ is given quantum oracle access to permutation P and $\mathcal{D}$ samples a uniform random function F from $\mathcal{F}(n,n)$.
2. All classical queries from $\mathcal{A}$ are answered using F. Let the resulting transcript of classical queries and responses be denoted by $T_j = \{(x_1,y_1),\ldots,(x_j,y_j)\}$.
3. All quantum queries from $\mathcal{A}$ are answered using P.

Phase 2. For all the queries beyond and including $(j+1)$-th classical query:

1. $\mathcal{D}$ receives $s_0,s_1,s_2,s_3 \in \{0,1\}^n$ and $\mathcal{D}$ has now quantum oracle access to $\mathsf{P}^{(b)}$, where $\mathsf{P}^{(0)} = \mathsf{P}$ and $\mathsf{P}^{(1)} = \mathsf{swap}_{\mathsf{P}(s_0),\mathsf{P}(s_1)} \circ \mathsf{swap}_{\mathsf{P}(s_2),\mathsf{P}(s_3)} \circ \mathsf{P}$.
2. Let $\mathcal{D}$ receives the $(j+1)$-th classical query from $\mathcal{A}$ which is x_{j+1}.
3. $\mathcal{D}$ sets $k_1 = x_{j+1} \oplus s_0$ and $k_2 = x_{j+1} \oplus s_2$, and $k = (k_1,k_2)$.
4. All classical queries from $\mathcal{A}$ are answered using $\mathsf{E}_k[\mathsf{P}^{(b)}]$.
5. All quantum queries from $\mathcal{A}$ are answered using $\mathsf{P}^{(b)}_{T_j,k}$.
6. Finally $\mathcal{D}$ returns whatever $\mathcal{A}$ returns.

Algorithm 5 Hybrid experiment $\mathbf{\Sigma}_j$. Including the boxed statements results in $\mathbf{\Sigma}'_j$.

$\mathsf{P} \xleftarrow{\$} \mathcal{P}(n)$, $\mathsf{F} \xleftarrow{\$} \mathcal{F}(n,n)$
Run $\mathcal{A}$ with quantum access to P and classical access to F, until $\mathcal{A}$ makes its $(j+1)$-th classical query x_{j+1}
$T_j := \{(x_1, y_1), \ldots, (x_j, y_j)\}$
$s_0, s_1, s_2, s_3 \xleftarrow{\$} \{0,1\}^n$, $\mathsf{P}^{(1)} := \mathsf{P} \circ \mathsf{swap}_{s_0,s_1} \circ \mathsf{swap}_{s_2,s_3}$.
$k_1 := s_0 \oplus x_{j+1}$, $k_2 := s_1 \oplus x_{j+1}$, $k := (k_1, k_2)$
$y_{j+1} := \mathsf{E}_k[\mathsf{P}^{(1)}](x_{j+1})$
$Q := \mathsf{P}^{(1)}_{T_j,k}$.
Give y_{j+1} to $\mathcal{A}$ as the answer to its $(j+1)$-th classical query
$T_{j+1} := \{(x_1, y_1), \ldots, (x_{j+1}, y_{j+1})\}$
$u_{j+1} := \mathsf{P}(s_1)$
if $\mathsf{bad}_1 := \mathsf{true}$, or $\mathsf{bad}_2 := \mathsf{true}$ **then**
    $\boxed{Q := \mathsf{P}_{T_{j+1},k}}$
Continue running $\mathcal{A}$ with quantum access to Q
(The remaining classical queries are answered as follows:)
$y := \mathsf{E}_k[\mathsf{P}^{(1)}](x)$
if $x = (s_1 \oplus k_1) \vee x = (s_3 \oplus k_2)$ **then**
    $\mathsf{bad}_3 := \mathsf{true}$
    $\boxed{y := \mathsf{E}_k[\mathsf{P}]}$
return y.

It is clear that $\mathcal{D}$ is a valid ARS distinguisher. If $b = 0$, then $\mathsf{P}^{(0)} = \mathsf{P}$, thus the output distribution of $\mathcal{A}$ is identical to its output distribution in $\mathbf{H}_j$. On the other hand, if $b = 1$, then the output of $\mathcal{A}$ is identically distributed to its output in $\mathbf{\Sigma}_j$. Hence, the distinguishing advantage between the hybrids $\mathbf{H}_j$ and $\mathbf{\Sigma}_j$ is equal to the distinguishing advantage of $\mathcal{D}$ in the resampling game. Using our extended resampling lemma, i.e., Lemma 3 for $r = 2$, we have

$$| \Pr[\mathcal{A}(\mathbf{H}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{\Sigma}_j) \to 1] | \leq 4\sqrt{2}\sqrt{\frac{p}{2^n}}. \tag{13}$$

Now, consider the following three events:

1. bad_1: $(s_1 \oplus k_1) \in \{x_1, \ldots, x_j\}$ or $(s_3 \oplus k_2) \in \{x_1, \ldots, x_j\}$.
2. bad_2: $\mathsf{P}(s_1) \in \{u_1, \ldots, u_j\} \cup \{y_{j+1} \oplus u_1 \oplus y_1, \ldots, y_{j+1} \oplus u_j \oplus y_j\}$.
3. bad_3: In phase 2, $\mathcal{A}$ queries its classical oracle on $s_1 \oplus k_1$ or on $s_3 \oplus k_2$ i.e $s_1 \oplus k_1 \in \{x_{j+2}, \ldots, x_q\}$ or $s_3 \oplus k_2 \in \{x_{j+2}, \ldots, x_q\}$.

Since, $s_1 \oplus k_1 = s_1 \oplus s_0 \oplus x_{j+1}$ and $s_3 \oplus k_2 = s_3 \oplus s_2 \oplus x_{j+1}$ are uniformly distributed, $\Pr[\mathsf{bad}_1] \leq 2j/2^n$. Similarly from the randomness of P, we get that $\Pr[\mathsf{bad}_2] \leq 2j/2^n$. Now for the event bad_3 consider the distinguisher $\mathcal{D}'$ that behaves identically to $\mathcal{D}$ in `Phase` 1 and `Phase` 2, but then when $\mathcal{A}$ terminates outputs 1 iff event bad_3 occurred. When $b = 0$, the probability that bad_3 occurs is at most $2(q-j)/2^n$. Since $\mathcal{D}'$ is a distinguisher for the resampling game, we get $\Pr[\mathsf{bad}_3] \leq 2(q-j)/2^n + 4\sqrt{2}\sqrt{p/2^n}$.

In Algorithm 5, we described $\mathbf{\Sigma}_j$ and $\mathbf{\Sigma}'_j$. Observe that, $\mathbf{\Sigma}_j$ and $\mathbf{\Sigma}'_j$ are identical until either bad_1 or bad_2 or bad_3 occur, and so we have

$$| \Pr[\mathcal{A}(\mathbf{\Sigma}_j) \to 1] - \Pr[\mathcal{A}(\mathbf{\Sigma}'_j) \to 1] | \leq \Pr[\mathsf{bad}_1 \vee \mathsf{bad}_2 \vee \mathsf{bad}_3] \leq \frac{(2q+2j)}{2^n} + 4\sqrt{2}\sqrt{\frac{p}{2^n}}. \tag{14}$$

Now, we are arguing that $\mathbf{\Sigma}'_j$ is identical to $\hat{\mathbf{H}}_j$:

1. Up to (just before) the $(j+1)$-th classical query $\boldsymbol{\Sigma}'_j$ is exactly identical to $\hat{\mathbf{H}}_j$.
2. In $\boldsymbol{\Sigma}'_j$, the response of the $(j+1)$-th classical query, y_{j+1} is defined as

$$\begin{aligned} y_{j+1} = \mathsf{E}_k[\mathsf{P}^{(1)}](x_{j+1}) &= \mathsf{P}^{(1)}(x_{j+1} \oplus k_1) \oplus \mathsf{P}^{(1)}(x_{j+1} \oplus k_2) \\ &= \mathsf{P}^{(1)}(s_0) \oplus \mathsf{P}^{(1)}(s_2) = \mathsf{P}(s_1) \oplus \mathsf{P}(s_3). \end{aligned} \tag{15}$$

   So, the value y_{j+1} is uniformly distributed.
3. In $\boldsymbol{\Sigma}'_j$, the oracle Q is always equal to $\mathsf{P}_{T_{j+1},k}$. When bad_1 occurs this is immediate (since then Q is set to $\mathsf{P}_{T_{j+1},k}$ in line 13). If "$\mathsf{bad}_1 = \mathsf{true}$" or "$\mathsf{bad}_2 = \mathsf{true}$", then for $i = 1, \ldots, j$ we have $x_i \oplus k_1 \neq s_0$ and $x_i \oplus k_1 \neq s_1$ (where the former is because $x_{j+1} \oplus k_1 = s_0$ but $x_i \neq x_{j+1}$ by assumption, and the latter is by definition of bad_1). Similarly, for $i = 1, \ldots, j$ we have $x_i \oplus k_2 \neq s_3$ and $x_i \oplus k_2 \neq s_2$. So, $\mathsf{P}^{(1)}(x_i \oplus k_1) = \mathsf{P}(x_i \oplus k_1)$ and $\mathsf{P}^{(1)}(x_i \oplus k_2) = \mathsf{P}(x_i \oplus k_2)$, for $i = 1, \ldots, j$. Thus

$$\begin{aligned} \overrightarrow{S}_{T_j,\mathsf{P}^{(1)},k} &\stackrel{\text{def}}{=} \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}^{(1)}(x_i \oplus k_1),u_i} \circ \mathsf{swap}_{\mathsf{P}^{(1)}(x_i \oplus k_2),(u_i \oplus y_i)} \\ &= \prod_{i=1}^{j} \mathsf{swap}_{\mathsf{P}(x_i \oplus k_1),u_i} \circ \mathsf{swap}_{\mathsf{P}(x_i \oplus k_2),(u_i \oplus y_i)} \\ &= \overrightarrow{S}_{T_j,\mathsf{P},k}. \end{aligned}$$

   Therefore,

$$\begin{aligned} Q = \mathsf{P}^{(1)}_{T_j,k} &= \overrightarrow{S}_{T_j,\mathsf{P}^{(1)},k} \circ \mathsf{P}^{(1)} \\ &= \overrightarrow{S}_{T_j,\mathsf{P},k} \circ \mathsf{swap}_{\mathsf{P}(s_0),\mathsf{P}(s_1)} \circ \mathsf{swap}_{\mathsf{P}(s_2),\mathsf{P}(s_3)} \circ \mathsf{P} \\ &= \overrightarrow{S}_{T_j,\mathsf{P},k} \circ \mathsf{swap}_{\mathsf{P}(x_{j+1} \oplus k_1),\mathsf{P}(s_1)} \circ \mathsf{swap}_{\mathsf{P}(x_{j+1} \oplus k_2),\mathsf{P}(s_3)} \circ \mathsf{P} \\ &= \overrightarrow{S}_{T_j,\mathsf{P},k} \circ \mathsf{swap}_{\mathsf{P}(x_{j+1} \oplus k_1),u_{j+1}} \circ \mathsf{swap}_{\mathsf{P}(x_{j+1} \oplus k_2),(u_{j+1} \oplus y_{j+1})} \circ \mathsf{P} \\ &= \overrightarrow{S}_{T_{j+1},\mathsf{P},k} \circ \mathsf{P} \\ &= \mathsf{P}_{T_{j+1},k}. \end{aligned}$$

4. In $\boldsymbol{\Sigma}'_j$, the response from oracle $\mathcal{O}(x)$ is always equal to $\mathsf{E}_k[\mathsf{P}](x)$. When bad_2 occurs, this is immediate. But if bad_3 does not occur then $x \neq s_1 \oplus k_1$ and $x \neq s_0 \oplus k_2$; we also know that $x \neq s_3 \oplus k_2 = x_{j+1}$ and $x \neq s_2 \oplus k_2 = x_{j+1}$ by assumption. So, $\mathsf{P}^{(1)}(x \oplus k_1) = \mathsf{P}(x \oplus k_1)$ and $\mathsf{P}^{(1)}(x \oplus k_2) = \mathsf{P}(x \oplus k_2)$.

Syntactically rewriting $\boldsymbol{\Sigma}'_j$ using the above observations yields an experiment that is identical to $\hat{\mathbf{H}}_j$. Thus using Eq. (10) we have

$$|\,\mathsf{Pr}[\mathcal{A}(\boldsymbol{\Sigma}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1]\,| \leq \frac{(2q+2j)}{2^n} + 4\sqrt{2}\sqrt{\frac{p}{2^n}}. \tag{16}$$

Hence, using Eq. (13) and Eq. (16) we have:

$$\begin{aligned} &|\mathsf{Pr}[\mathcal{A}(\mathbf{H}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1]\,| \\ &\leq |\,\mathsf{Pr}[\mathcal{A}(\mathbf{H}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\boldsymbol{\Sigma}_j) \to 1]\,| + |\,\mathsf{Pr}[\mathcal{A}(\boldsymbol{\Sigma}_j) \to 1] - \mathsf{Pr}[\mathcal{A}(\hat{\mathbf{H}}_j) \to 1]\,| \\ &\leq 4\sqrt{2}\sqrt{\frac{p}{2^n}} + \frac{(2q+2j)}{2^n} + 4\sqrt{2}\sqrt{\frac{p}{2^n}}. \\ &= \frac{(2q+2j)}{2^n} + 8\sqrt{2}\sqrt{\frac{p}{2^n}}. \end{aligned}$$