Técnico Lisboa

University of Lisbon

Research Seminar in Information Security

(Prof. Paulo MATEUS)

Doctoral Program in Information Security

2023/2024 - 1st Semester

---

# Post-Quantum Security of the ("tweakable") Even-Mansour Cipher

**(Christian MAJENZ - November 23, 2023)**

---

**Report written by:**

- Rúben BARREIRO:
  - *ruben.andre.letra.barreiro@tecnico.ulisboa.pt*

Last updated: May 10, 2024

## 1 Motivation

The task of interest in this seminar is to address the post-quantum security of the ("tweakable") Even-Mansour Cipher through the combination of two previous experimental results, one of them on the security model of the plain Even-Mansour Cipher, as well as how to build variants of this cryptographic primitive considering also security models for the post-quantum setting [1].

As a motivation quick start, nowadays, we are concerned about the known quantum attacks against the currently used modern cryptography, and there are different types of such attacks. First of all, we have what we can call post-quantum attacks on Asymmetric Cryptography (or Public Key Cryptography) [2, 3], such as the ones performed by the well-known Shor's algorithm [4,5], that completely break some cryptographic primitives such as Rivest-Shamir-Adleman (RSA) [6] and other based on Elliptic Curves [7, 8] and Discrete Logarithm problem [9]. Then, we have other post-quantum attacks on Symmetric Cryptography [10], such as the ones performed by Grover's algorithm [11] and other related ones [12, 13], which do not break completely some of the currently used cryptographic primitives. However, Symmetric Cryptography is significantly impacted by these mentioned quantum algorithms since we can use these algorithms for (exhaustive) key search attacks against block ciphers or collision attacks against hash functions. On the other hand, once we know some specific cryptographic constructions very well, we can take classical attack strategies like differential and linear cryptoanalysis, and speed them up using Grover's algorithm [14] for finding keys or collisions, which degrade the security of these cryptographic protocols, at least in principle despite we still do not know how to quantify that degradation of security. Finally, we have an additional class of quantum attacks, which we can call beyond post-quantum attacks, where we make some assumptions that are not justifiable in the usual use case for a symmetric key cryptographic algorithm. In this case, we assume the key is secret but that an attacker can access some pairs of plaintexts and ciphertexts while using a Known-Plaintext Attack (KPA) and exploiting quantum computing capabilities. These attacks could allow us to completely break specific cryptographic schemes, such as the Even-Mansour cipher [15,16] and some other block cipher modes of operation, in an unrealistic attack model.

We are far less concerned about the security of Symmetric Cryptography against quantum attacks than the one for Asymmetric Cryptography, at least for pre-quantum Public Key Cryptography. However, we should still have a post-quantum security proof in the global settings for our security proofs.

## 2 ("Tweakable") Even-Mansour Cipher

The Even-Mansour cipher [15, 16] is arguably one of the simplest possible block cipher constructions, on which we assume there is a public permutation $P$ that everybody can evaluate and should have some specific reasonable cryptographic properties. The block cipher operation of this cryptographic primitive requires a secret key $k$, and it starts by applying an eXclusive OR (XOR) operation on that secret key $k$ and a message $x$. Then, it applies the permutation $P$ to the previous intermediate result, followed by a new XOR operation on it and on the secret key $k$ again. This cryptographic scheme is an essential minimal construction because we can see a cipher as an oracle box we can evaluate forward and backward. For this reason, we need to have secret key $k$ before and after the permutation since it is breakable otherwise. Despite the simplicity of this cryptographic scheme, it represents a similar key aspect to the one used for real-world standardized symmetric cryptographic lightweight schemes such as Elephant [17], Chaskey [18], and Minalpher [19]. Generally, we can describe the Even-Mansour cipher as follows. Given a public permutation $P : \{0,1\}^n \to \{0,1\}^n$ and a key $k \in \{0,1\}^n$, the cipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is defined as $E_k[P](x) = P(x \oplus k) \oplus k$. An illustrative sketch of the Even-Mansour cipher is given in the figure below:

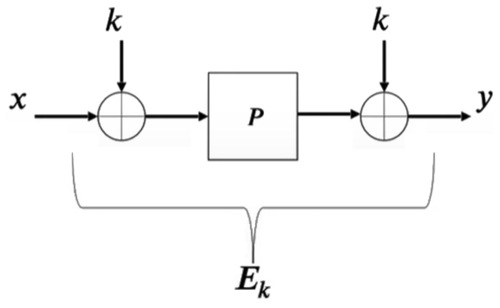


Figure 1: High-level illustration sketch of the Even-Mansour cipher.

Furthermore, we can make this cryptographic construction a little more complicated and versatile by adding a simple extra input to the block cipher,

which we usually call the "tweak" input $t$. This modification on the original block cipher should create a family of independent block ciphers dependent on this input $t$. Namely, for each different "tweak" $t$, we should obtain an independent block cipher while still only using one secret key $k$. Basically, in this new variant, we have a function $f(k, t)$ called the "tweak" function that fulfills few cryptographic properties, and for each random secret key, the output of this new block cipher is as good as the random and independent Even-Mansour keys used in the original block cipher. This feature can be helpful because, for some cipher mode operations, we can be concerned about length extension attacks, and we should apply a different operation in the last round of that cipher mode operation. In that case, we can use a different "tweak" input for the final round of the block cipher, for example. There are several applications of how to use this new "tweak" input $t$, and it is actually this new key ingredient of the ("tweakable") Even-Mansour cipher that the previously mentioned symmetric key encryption schemes, such as Elephant [17], Chaskey [18], and Minalpher [19] use in their constructions.
Generally, we can describe the "tweak" function used in the ("tweakable") Even-Mansour cipher as follows. It is represented by a mathematical function $f : \{0,1\}^{\ell_k} \times \mathcal{T} \to \{0,1\}^n$ so that for a random key $k \leftarrow \{0,1\}^{\ell_k}$, the outputs $f(k, t)$ for different $t \in \mathcal{T}$ are as good as independent Even-Mansour keys. An illustrative sketch of the ("tweakable") Even-Mansour cipher is given below:

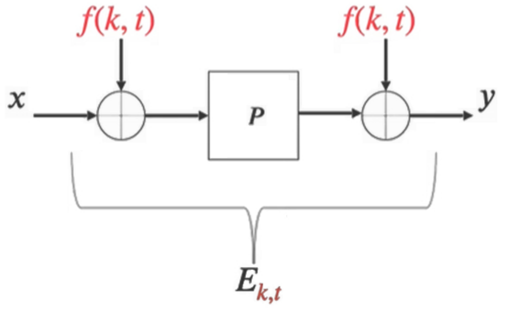


Figure 2: High-level illustration sketch of the ("tweakable") Even-Mansour cipher.

Then, we have an additional variant of this cryptographic primitive, which we know as the Even-Mansour Cipher with Key Expansion. Namely, this cryptographic primitive uses a shorter key $k$ of size $l$, which we expand using zero-padding of size $(n-l)$ and we permute the expanded bit string. Then, we apply the XOR operation with the input $x$ and the previous permuted result, permuting the output again before XORing it with the permutation of the shorter key expanded with zero-padding one last time. In this new variant, all the permutations used are the same permutations $P$ used for the original Even-Mansour cryptographic construction. However, we can conclude this cryptographic construction has both an advantage and a disadvantage. This cryptographic scheme is a bad idea because we always want to use different cryptographic primitives for distinct parts of a cryptographic construction. However, on the other hand, this same cryptographic scheme ends up being a great idea from a practical point of view because we can have different block lengths than the key lengths for some computational devices with Hardware acceleration for the permutation function $P$ if we only want to compute it. Generally, this Even-Mansour cipher with Key Expansion has a shorter key $k \in \{0,1\}^{\ell}$ and we can represent the respective cipher procedure $E$ as $E : \{0,1\}^{\ell} \times \{0,1\}^{n} \rightarrow \{0,1\}^{n}$. An illustrative sketch of this variant of Even-Mansour cipher with Key Expansion is given in the figure below:

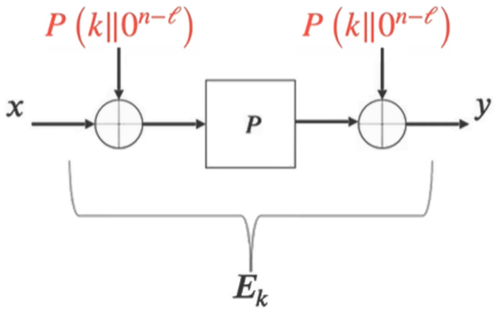
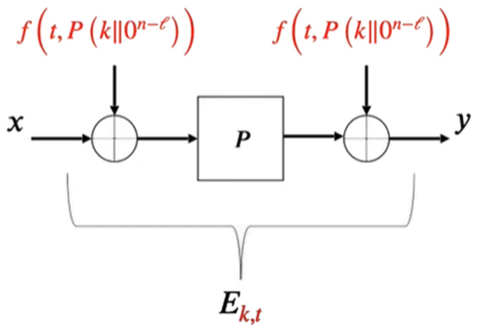


Figure 3: High-level illustration sketch of
the Even-Mansour cipher with Key Expansion.

Finally, we can combine the ("tweakable") Even-Mansour Cipher with the Even-Mansour Cipher with Key Expansion. Then, considering the key expansion and permutation for the first input of the "tweak function", we end up with a more complex "tweak function" $f\left(t, P\left(k||0^{(n-\ell)}\right)\right)$. This feature keeps the cryptographic construction modular despite using this "tweak" function inside it not being sufficient to prove any post-quantum security yet.

An illustrative sketch of the ("tweakable") Even-Mansour cipher with Key Expansion, as a combination of the previous schemes, is given below:



$f\left(t, P\left(k \| 0^{n-\ell}\right)\right)$
$f\left(t, P\left(k \| 0^{n-\ell}\right)\right)$
$x$
$P$
$y$
$E_{k,t}$

Figure 4: High-level illustration sketch of
the ("tweakable") Even-Mansour cipher with Key Expansion.

## 3 Attack Models

There are three attack models: the classical attack model, the post-quantum attack model (also called the $Q1$ attack model), and the beyond post-quantum model (also called the $Q2$ attack model) [20, 21]. We can reason and argue about which attack model is the right one. The classical attack model is a little optimistic nowadays, and the beyond post-quantum attack model assumes the global community has considerably large access to quantum resources, which is unlikely since most of the devices we use daily are computational but they are not quantum ones. In fact, these cryptographic primitives based on the Even-Mansour block cipher fit more in the Lightweight Cryptography field, where the use cases are precisely the ones comprising the ubiquitous and mobile devices we use daily, following the trend of the Internet of Things (IoT). For these reasons, it is more reasonable to follow the post-quantum attack model, which is the case where we can evaluate this public permutation primitive $P$ on a quantum computer, but we only allow the construction of the Even-Mansour block cipher to be evaluated classically.

## 3.1 Security of the Even-Mansour Cipher

For the security of the Even-Mansour Cipher, we start by idealizing the permutation $P$ as a uniformly random permutation. Now, we assume that some attacker makes a certain number of queries $q_P$ and $q_E$ on the public random permutation and the Even-Mansour Cipher, respectively, both in the forward and backward directions. For the classical attack model, we know that to break this block cipher classically, the product of the two numbers of queries must be in the order of $2^n$, i.e., $q_P \times q_E = \Omega(2^n)$ [15]. Once we consider the unrealistic beyond post-quantum attack model, we can completely break this cryptographic scheme in theory since we can apply Simon's algorithm [12] and attack this block cipher using only $\mathcal{O}(n)$ queries [22]. These observations make the security for the perspective of the post-quantum attack model much more frangible because we cannot argue without loss of generality that the attack performed is unitary once we are dealing with a mixture of oracles, where some of them have a quantum essence, and others have a classical one.

# 4 Experimental Results

For the experimental results, the author and his colleagues combined the previous results for ("tweakable") Even-Mansour Cipher and Even-Mansour Cipher with Key Expansion [1]. Namely, for an arbitrary Even-Mansour Cipher, we have the security bound given by the following expression:

$$10 \cdot 2^{-\frac{\hat{n}}{2}} \cdot (q_P \cdot \sqrt{q_E} + q_E \cdot \sqrt{q_P})$$

For the ("tweakable") Even-Mansour cipher $E[P]$ (with or without Key Expansion), any quantum adversary making at most $q_P$ quantum queries to $P$ and $q_E$ classical queries to an oracle $O$ has distinguishing advantage between $O = E$ and $O = R$ at most the security bound given above. Here, $R$ is an independent (family of) random permutation and $\hat{n} = n$ ($\hat{n} = \ell$ for the Key Expansion variant). Additionally, $q_P$ and $q_E$ are again the number of queries made to the public random permutation and the Even-Mansour Cipher, respectively, while $\hat{n}$ is the key length for all cases. For the Key Expansion variant case, the key length is equal to the shorter key length ($\hat{n} = \ell$). Otherwise, the key length is equal to the block length used ($\hat{n} = n$). This experimental result was shown before for plain Even-Mansour Cipher [23] and is interesting in several contexts. First, this result is suitable when considering adversaries that decide on all the classical queries ahead of time, which is a non-adaptive choice on where to query. On the other hand, this experimental result is also sound for the realistic setting, where the number of queries to the key expansion construction is much smaller than the number of queries to the public random permutation. This last result is tight assuming $q_E \ll q_P$ and corresponds to cryptographic attacks based on solving the claw-finding problem. The security bound expression is also the success probability of an oracle that tries to distinguish the block cipher from a random permutation, which matches the one from the Brassard-Horne-Tapp (BHT) [13] or "offline" Simon's [12, 24] algorithms. This security bound shows us we obtain the expected analog of the classical security for the key expansion case once we obtain the product of the two numbers of queries in the classical case, being equivalent to a collision [25, 26] or a claw-finding [27–29] attack. For the "tweakable" case, the success probability is the same as the key expansion case. Additionally, for a fixed "tweak" input and a random secret key, the output needs to be random, while for two fixed but distinct tweaks and a random secret key, the XOR operation of the outputs still needs to be random. For those cases, a universal hash function such as the Carter-Wegman scheme [30, 31] works as a good "tweak" function.

## 5 Proof Approach

Now, we can dive into the proof approach and the challenges that may occur when we try to prove the previous experimental results more in detail.

### 5.1 General Proof

In the classical setting, security proofs of Even-Mansour usually use what we can call global techniques. For example, these techniques may involve characterizing the probability that certain malicious and bad events happen that are defined globally for an entire transcript of queries. Another global technique commonly used is the $H$-coefficient technique, which also involves transcripts of queries, partitioning them into good and bad ones, and then comparing the expected results from the real world with the ideal world. In the quantum setting, we do not have transcripts of queries, and in particular, some techniques called Compressed Oracles can recover a kind of transcript of queries for the quantum setting. However, this technique is not available yet for random permutations. Therefore, what we can do is resort to a much "more primitive" technique, which turns out to be a more common technique in the quantum setting and that is referred to as the hybrid argument.

## 5.2 The Hybrid Argument

Now, we can illustrate what happens when an adversary tries to break the ("tweakable") Even-Mansour Cipher with Key Expansion, as shown below:

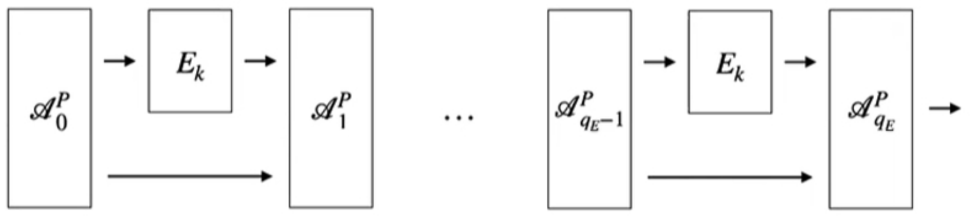


Figure 5: High-level illustration sketch of
the Hybrid Argument for an adversary having $(q_E + 1)$ stages.

Namely, we can partition the adversary's actions in $(q_E+1)$ stages, basically as the phases before, between, and after the classical queries made to the block cipher. Thus, each one of these stages makes any arbitrary number of queries to the public permutation primitive $P$, which can be quantum queries, but the sum of those queries assumes some upper bound given by $q_P$. The naive hybrid approach consists of us replacing successively the block cipher $E_k$ with the idealized object stage by stage, which is an independent random permutation $R$. However, if we replace the block cipher $E_k$ with independent random permutations $P$ for the first couple of stages, the adversary could notice we changed the original cryptographic scheme. Therefore, we need to keep the future of the cryptographic scheme consistent with its ideal past in some sense, which represents a challenge in this proof. Luckily, what we can do in this situation is postpone the consistency problem until it disappears.

For the proof exploring the hybrid argument, we can design and introduce specific objects called hybrids, which are denoted as $H$ and shown below:

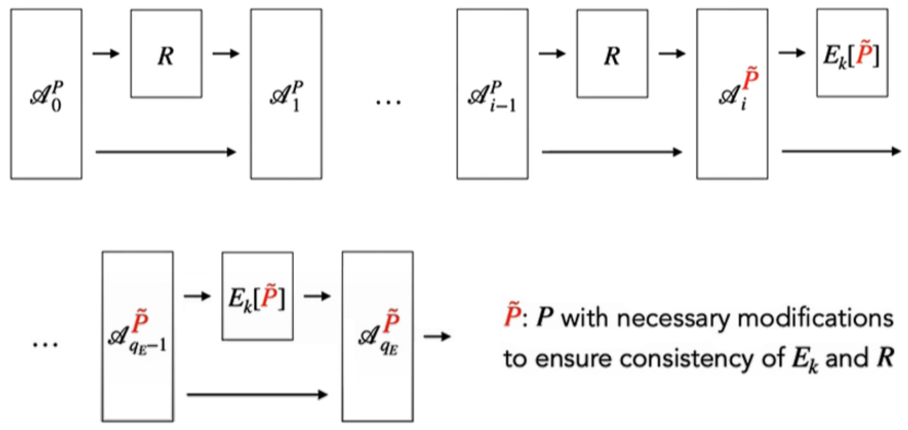

$\mathscr{A}_0^P$
$R$
$\mathscr{A}_1^P$
$\mathscr{A}_{i-1}^P$
$R$
$\mathscr{A}_i^{\tilde{P}}$
$E_k[\tilde{P}]$
$\mathscr{A}_{q_E-1}^{\tilde{P}}$
$E_k[\tilde{P}]$
$\mathscr{A}_{q_E}^{\tilde{P}}$
$\tilde{P}$: $P$ with necessary modifications to ensure consistency of $E_k$ and $R$

Figure 6: High-level illustration sketch of the Hybrid objects for the security proof.

More specifically, for the first $i$ stages, we use these independent random permutations $R$, and then, we switch to the Even-Mansour Cipher in the end. But now, we slightly change the public random permutations $P$ to ensure the consistency of the cipher $E_k$ and independent random permutations $R$ with the previous queries made to the original scheme, from which we obtain a new random permutation $\tilde{P}$. Namely, this modified random permutation $\tilde{P}$ blows up successively as the number of the first $i$ stages grows. However, there are also just a few applications for this modification, and in the end, in the last hybrid object, the effects of this modification disappear. Therefore, we do not have to worry about changing the public random permutations $P$ too much, once it works out for our general approach and how the proof proceeds.

## 6 Resampling Lemma Zoo

When we want to prove the indistinguishability property of these adjacent hybrid objects introduced before for this approach, we need to use some Resampling Lemmas, which we will describe in the following section in detail.

## 6.1 The Resampling Lemma for Functions

For simplicity reasons, we can start with the Resampling Lemma for random functions, where we have a uniformly random function $H$, which maps $n$ bits to other $n$ bits, and we have a two-stage algorithm $\mathcal{D}$ that plays a given interactive game with a challenger and a prover. This game starts with algorithm $\mathcal{D}$ interacting with some oracle to compute the random function $H$, outputting then some internal state. Then, the challenger samples a random bit $b$, jointly with two random bitstrings of size n denoted as $x^*$ and $y^*$. Now, we denote by $H_{(x^* \mapsto y^*)}$ the uniformly random function $H$ that we reprogram at the input $x^*$ to the output $y^*$. For $b = 0$, we do not perform any action, which gives us $H^0 = H$, and for $b = 1$, we reprogram the function $H$ at the input $x^*$ to the output $y^*$. Then, in the second phase of this algorithm, we get access to this possibly changed $H$ function that gets $x^*$ as input, and the prover needs to decide if the $H^b$ function is the $H$ function reprogrammed at this input or not, choosing a $b'$ accordingly. The difficulty of winning this game is that $x^*$ is sampled at random by the challenger. The prover can gather as much information about $H$ as possible for $b = 0$, but if it does not query at $x^*$, it has no idea what the output should be, and it cannot detect if the challenger changed the corresponding output. Finally, the author and their colleagues proved that this simple case with a uniform random function provides a very sharp bound on the probability of winning this game [32]. Namely, the probability of winning and distinguishing this game cannot be equal to 1 unless the prover makes a number of queries on the order of $2^n$. A simple illustration of this interactive proof game is given in the figure below:

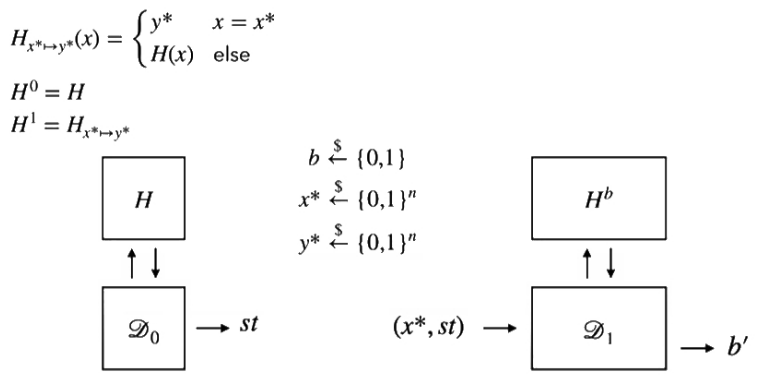

$H_{x^*\mapsto y^*}(x) = \begin{cases} y^* & x = x^* \\ H(x) & \text{else} \end{cases}$
$H^0 = H$
$H^1 = H_{x^*\mapsto y^*}$
$H$
$b \xleftarrow{\$} \{0,1\}$
$x^* \xleftarrow{\$} \{0,1\}^n$
$y^* \xleftarrow{\$} \{0,1\}^n$
$H^b$
$\mathscr{D}_0$
$st$
$(x^*, st)$
$\mathscr{D}_1$
$b'$

Figure 7: High-level illustration sketch of
the Interactive Proof Game for
the Resampling Lemma for Functions.

The bound for the probability of winning this interactive game is given as:

$$Pr[\mathcal{D}\ wins] \leq \frac{1}{2} + \frac{3}{2} \cdot \sqrt{q_0 \cdot 2^{-n}}$$

## 6.2 The Resampling Lemma for Permutations

For the case of permutations, we want to take a similar approach to what we follow for random functions. However, if we take a random input and reprogram the permutation to a random output, we are no longer dealing with a random permutation. Thus, keeping track of what is happening becomes extremely hard when we follow that approach using the hybrid argument. In the end, it is preferable to keep the permutation property and follow again a two-phase interactive game. In the first phase, also known as the learning phase, instead of resampling a new output, the challenger picks two inputs, $s_0$ and $s_1$, jointly with a random bit $b$ again. If $b = 0$, the challenger does not perform any action again. Otherwise, if $b = 1$, the challenger switches the outputs for these two picked inputs. Note that we know if $b = 0$, the challenger chooses the actual permutation we want to reprogram, and it picks $s_0$ and $s_1$ at random. Actually, the challenger has reprogrammed the output at the inputs $s_0$ and $s_1$ to a uniformly random output for $b = 0$ because it picked $s_0$ and $s_1$ uniformly random. Now, the challenger needs to finish the game, and as expected, the prover gets the two resembling inputs and tries to guess the bit $b$ used, similar to what we defined for the case of random functions. For this case, the probability of winning the game (and respective security bound) is essentially the same as in the case of random functions [33]. An illustration of this interactive challenge-proof game is given below:

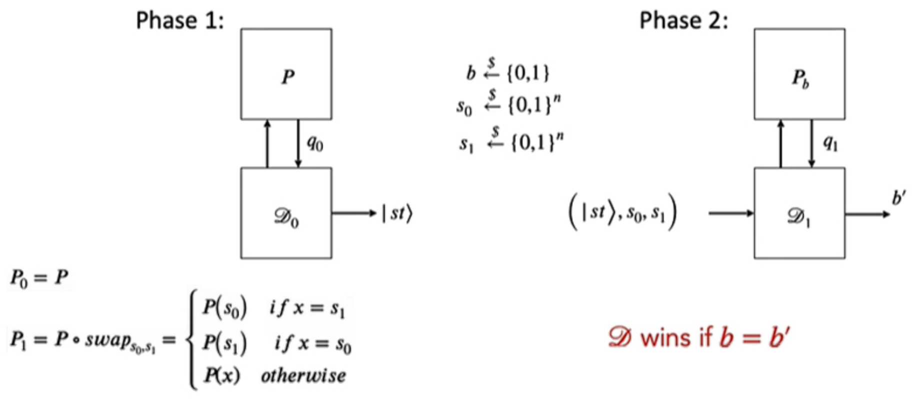

Phase 1:
$P$
$q_0$
$\mathscr{D}_0$
$|st\rangle$
Phase 2:
$b \xleftarrow{\$} \{0,1\}$
$s_0 \xleftarrow{\$} \{0,1\}^n$
$s_1 \xleftarrow{\$} \{0,1\}^n$
$P_b$
$q_1$
$(|st\rangle, s_0, s_1)$
$\mathscr{D}_1$
$b'$
$P_0 = P$
$P_1 = P \circ swap_{s_0,s_1} = \begin{cases} P(s_0) & if\ x = s_1 \\ P(s_1) & if\ x = s_0 \\ P(x) & otherwise \end{cases}$
$\mathscr{D}$ wins if $b = b'$

Figure 8: High-level illustration sketch of
the Interactive Proof Game for
the Resampling Lemma for Permutations.

The bound for the probability of winning this interactive game is given as:

$$Pr[\mathcal{D}\ wins] \leq \frac{1}{2} + 2 \cdot \sqrt{q_0 \cdot 2^{-n}}$$

## 6.3 The Resampling Lemma for Key Expansion

Finally, if we consider the Even-Mansour Cipher variant with the additional Key Expansion step, the proof for indistinguishability becomes much more complicated. In this case, we start again with the learning phase, but now we are more constrained on what we can do. Namely, we allow the adversary to output a distribution $\mathcal{D}$ and a parameter $\tau$. The adversary outputs the distribution $\mathcal{D}$, according to which we should sample the input $\hat{s}$ where we disorder with the uniformly random public permutation $P$. The adversary also outputs the parameter $\tau$, which represents a form of post-processing of the sample after expanding it through the public random permutation $P$. An illustration of this interactive challenge-proof game is given below:

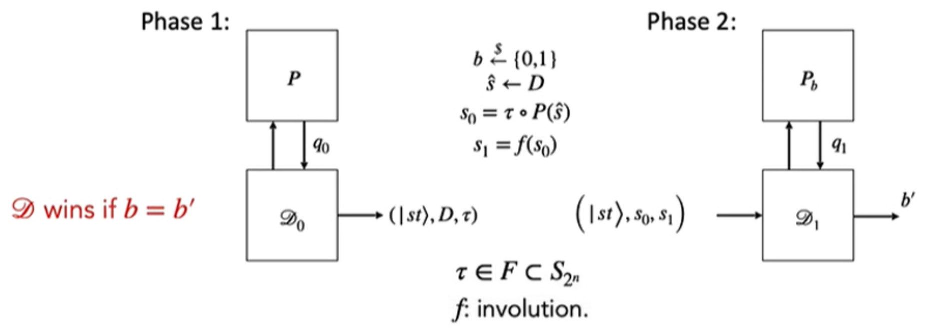


Figure 9: High-level illustration sketch of
the Interactive Proof Game for
the Resampling Lemma for Key Expansion.

Now, the adversary has a lot of choices regarding how the challenger should pose the challenge, and we need to choose $\tau$ from a subset of all permutations. Otherwise, the security bound will not be good enough, and the "tweak function" $f$ should represent an involution of pairs of $s_0$ and $s_1$. The prover should decide this pairing ahead of time. However, since we have this very restricted setting, then even though the sampling of the inputs involves the public random permutation $P$ itself and the adversarial actions, we still can obtain the following upper bound for the probability of winning the interactive proof game, which represents the security bound itself [1]:

$$Pr[\mathcal{D}\ wins] \leq \sqrt{\epsilon} \cdot \left(1 + \sqrt{q + \log\left(\frac{11 \cdot |F|}{\sqrt{\epsilon}}\right)}\right)$$

Where:

$$\epsilon = \max_{s} \Pr_{(s_0,s_1)\leftarrow\mathcal{D}} [s = s_0 \vee s = s_1]$$

In this security bound, the most complex part of the mathematical expression is on logarithmic order. At the same time, $\epsilon$ is the minimal entropy of the distribution $\mathcal{D}$ to sample $\hat{s}$ because it is the adversary that provides it.

## 7 Superposition Oracle Resampling

Now, to prove these resamplings for both random function and permutation cases, we need to introduce the notion of superposition oracle resampling [34].

### 7.1 Superposition Oracle Resampling for Functions

For the case of functions, we use these superposition oracles where we have a random function for the "vanilla" random oracle, and we have a superposition of functions for the superposition oracle. Namely, in this last oracle, for each function output, we have a separate quantum register that is initialized in the uniform superposition of states to represent the uniform function output. For the superposition oracle, instead of having this fixed query unitary as in the "vanilla" random oracle, we have a controlled unitary that reads out this appropriate quantum register and answers the query that way. Finally, resampling for the superposition oracle is very similar to the case of the "vanilla" random oracle, with random input and output, using as well the reprogramming step. However (and luckily), we can discard (or reset) the appropriate quantum register $F_{x^*}$ and initialize it in a new uniform quantum superposition of states, resulting now in our new resampling operation, since the classical resampling is just sample a fresh independent output for which we can use an independent uniform quantum superposition state. Then, we conclude this set of steps is sufficient for achieving the core argument of replacing this Oracle distinguishability task with a state discrimination task, where we need to discriminate the original quantum state of the oracle simulation from the one where we replaced this uniform quantum superposition. This resampling works because, after just a few queries, most of the quantum registers intuitively contain the initial quantum state. After all, the adversary has only looked at $q$ many inputs (and queries), even if it uses a quantum superposition of states to perform such query operations.

We now show a global overview of the main differences between the random function oracle and the superposition oracle for random functions below:

$H : \{0,1\}^n \to \{0,1\}^n$

| **Random Function Oracle** | **Quantum Superposition Oracle** [34] |
|---|---|
| For each $x \in \{0,1\}^n$: $H(x) \leftarrow \{0,1\}^n$ | For each $x \in \{0,1\}^n$: Initialize $n$-qubit quantum register $F_x$ in the quantum state $\lvert\phi_0\rangle = \lvert+\rangle^{\oplus n}$ |
| **Query Unitary**: $U_H\lvert x\rangle_X\lvert y\rangle_Y = \lvert x\rangle_X\lvert y \oplus H(x)\rangle_Y$ | **Query Unitary**: $O\lvert x\rangle_X = CNOT^{\oplus n}_{F_x:Y}$ |
| **Resampling at** $x^* : y^* \leftarrow \{0,1\}^n$, $H'(x) = \begin{cases} y^*, & \text{if } x = x^*. \\ H(x), & \text{otherwise.} \end{cases}$ | **Resampling at** $x^*$: • Discard the contents of the quantum register $F_{x^*}$; • Prepare the exact same quantum register $F_{x^*}$ in the quantum state $\lvert\phi_0\rangle$; |

Most quantum registers are in the quantum state $\lvert+\rangle^{\oplus n}$ after $q$ queries...

$\Rightarrow$ Distinguish quantum states resampled or not $\to$ Quantum State Discrimination (QSD)

Table 1: Global Overview of the main differences between the Random Function Oracle and Quantum Superposition Oracle.

## 7.2 Superposition Oracle Resampling for Permutations

For the case of permutations, we still use a superposition oracle similar to the one used for the case of functions. However, we initialize it in a uniform quantum superposition of all function tables for a permutation. The query unitary is the same as the case of functions, but now we need a more intricate way of resampling, where we choose two uniformly random inputs and swap their outputs. In this case, we cannot take the same action as for the case of functions and discard some registers, initializing them then from scratch, since we will not end up with a permutation in that case. So, the technique we adopt for this case is to prove the resampling lemma in the "Heisenberg Picture", and we show that the reprogramming operation essentially almost commutes with the superposition oracle calls of the adversary, at least if the input is honest. We show an overview of the differences between the random permutation oracle and the superposition oracle for permutations below:

$P \in S_{2^n}$

| **Random Permutation Oracle** | **Quantum Superposition Oracle** |
|---|---|
| $P \leftarrow S_{2^n}$ | Initialize $2^n \cdot n$-qubit quantum register $F$ in the quantum state $\lvert\phi_0\rangle = \frac{1}{\sqrt{2^n!}} \sum_{\pi \in S_{2^n}} \lvert\pi(0^n)\rangle \lvert\pi(0^{(n-1)}1)\rangle \ldots \lvert\pi(1^n)\rangle$ |
| **Query Unitary**: $U_P\lvert x\rangle_X \lvert y\rangle_Y = \lvert x\rangle_X \lvert y \oplus P(x)\rangle_Y$ | **Query Unitary**: $O\lvert x\rangle_X = CNOT^{\oplus n}_{F_x:Y}$ |
| **Resampling at** $(s_0, s_1) : b \leftarrow \{0,1\}$, $P'(x) = \begin{cases} s_{(i-b)}, & \text{if } x = s_i. \\ P(x), & \text{otherwise.} \end{cases}$ | **Resampling at** $(s_0, s_1)$: ??? |

Prove in "Heisenberg Picture" instead:

$$\left[Swap_{F_{s_0}F_{s_1}}, O_{XYF}(P_{s_0 s_1})_X\right] = 0,\ P_{s_0 s_1} = \mathbb{1} - \lvert s_0\rangle\langle s_0\rvert - \lvert s_1\rangle\langle s_1\rvert$$

Table 2: Global Overview of the main differences between the Random Permutation Oracle and Quantum Superposition Oracle.

### 7.2.1 Superposition Oracle Resampling for Permutations used in Key Expansion

Finally, we have the case of permutations used in the key expansion variant of the Even-Mansour Cipher, which is the most complicated case. In this case, we want to use the previously mentioned commutation result, where the resampling operation approximately commutes with the superposition oracle calls made by the adversary. However, note the inputs $s_0$ and $s_1$, which show up in this commutator equality, are sampled conditioned on some quantum measurement the adversary has made after querying this quantum superposition oracle. Therefore, the proof approach applied for the previous permutation case using a quantum superposition oracle does not work for this case since we cannot use a commutation result between operators applied in the past and conditioned on measurement outcomes we achieved after applying these same operators. Namely, what we do instead is to find some information inside the random permutation $P$ that represents independent random variables. Then, we can perform the old technique that analyzes what happens to the quantum superposition state and what this information inside the random permutation $P$ is. Namely, we partition the set of inputs into $2^{(n-1)}$ pairs, and for each pair $(s_0, s_1)$, if we have an order on the set of strings, then for each pair, we can check if the random permutation $P$ inverts that pair or not, according to this order. For example, if the input $s_0$ is smaller than the input $s_1$ according to the order, we check if the output of the random permutation $P$ on $s_1$ is smaller than the one of the random permutation $P$ on the input $s_0$. If that verification holds, we conclude that the random permutation $P$ inverted the order. Otherwise, we conclude that the random permutation $P$ did not invert the order. Therefore, we can claim the bits for each pair $(s_0, s_1)$ are independent in the quantum superposition oracle by having independent quantum registers with each of these bits. Now, there are quantum superpositions of them instead of simple random bits, which are always random whether the random permutation $P$ inverts this particular pair of inputs $(s_0, s_1)$ or not according to the order. Then, for these independent random variables, we have a set of quantum registers for the quantum superposition oracle similar to the random function case, and we can use the same techniques used in that case as demonstrated previously.

## 8 Outlook

In this seminar, the author showed how the research work it developed jointly with its colleagues proved the post-quantum security of the Even-Mansour Cipher and related symmetric cryptographic constructions. Additionally, the author also demonstrated that other lightweight cryptographic primitives, such as the mentioned Elephant [17] and Chaskey [18], use a generalized version of the Even-Mansour Cipher [15, 16], and to achieve their security proofs [1,33], we need to generalize them and change this key ingredient called resampling lemma. However, the security technique demonstrated in this seminar can handle them together with some resampling lemma zoology and quantum superposition oracles for security models with quantum adversaries.

## References

[1] Gorjan Alagic, Chen Bai, Jonathan Katz, Christian Majenz, and Patrick Struck. Post-Quantum Security of Tweakable Even-Mansour, and Applications, 2022.

[2] Ralph Merkle. Secure Communications Over Insecure Channels. *Commun. ACM*, 21(4):294–299, 1978.

[3] Whitfield Diffie and Martin Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[4] Peter Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[5] Peter Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[6] Ron Rivest, Adi Shamir, and Leonard Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[7] Victor Miller. Use of Elliptic Curves in Cryptography. In Hugh Williams, editor, *Advances in Cryptology — CRYPTO '85 Proceedings*, pages 417–426, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg.

[8] Neal Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[9] Taher Elgamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[10] Vincent Rijmen and Joan Daemen. Advanced Encryption Standard. *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, 19:22, 2001.

[11] Lov Grover. A Fast Quantum Mechanical Algorithm for Database Search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 212–219, New York, NY, USA, 1996. Association for Computing Machinery.

[12] Daniel Simon. On the Power of Quantum Computation. *SIAM Journal on Computing*, 26(5):1474–1483, 1997.

[13] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum Cryptanalysis of Hash and Claw-Free Functions. In Cláudio Lucchesi and Arnaldo Moura, editors, *LATIN'98: Theoretical Informatics*, pages 163–169, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[14] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Quantum Differential and Linear Cryptanalysis. *IACR Transactions on Symmetric Cryptology*, 2016(1):71–94, 2016.

[15] Shimon Even and Yishay Mansour. A Construction of a Cipher From a Single Pseudorandom Permutation. *J. Cryptology*, 10:151–162, 1997.

[16] Orr Dunkelman, Nathan Keller, and Adi Shamir. Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 336–354, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[17] Tim Beyne, Yu Chen, Christoph Dobraunig, and Bart Mennink. Dumbo, Jumbo, and Delirium: Parallel Authenticated Encryption for the Lightweight Circus. *IACR Transactions on Symmetric Cryptology*, 2020(S1):5–30, 2020.

[18] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers, 2014.

[19] Yu Sasaki, Yosuke Todo, Kazumaro Aoki, Yusuke Naito, Takeshi Sugawara, Yumiko Murakami, Mitsuru Matsui, and Shoichi Hirose. minalpher v1. *CAESAR Round*, 1, 2014.

[20] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking Symmetric Cryptosystems Using Quantum Period Finding. In *Advances in Cryptology – CRYPTO 2016*, pages 207–237, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[21] Mark Zhandry. How to Construct Quantum Random Functions. *J. ACM*, 68(5), 2021.

[22] Hidenori Kuwakado and Masakatu Morii. Security on the Quantum-Type Even-Mansour Cipher. In *2012 International Symposium on Information Theory and its Applications*, pages 312–316, 2012.

[23] Joseph Jaeger, Fang Song, and Stefano Tessaro. Quantum Key-Length Extension, 2021.

[24] Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and André Schrottenloher. Quantum Attacks Without Superposition Queries: The Offline Simon's Algorithm. In Steven Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 552–583, Cham, 2019. Springer International Publishing.

[25] Xiaoyun Wang, Dengguo Feng, Xuejia Lai, and Hongbo Yu. Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD, 2004.

[26] Marc Stevens. Fast Collision Attack on MD5, 2006.

[27] Whitfield Diffie and Martin Hellman. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *Computer*, 10(6):74–84, 1977.

[28] Shengyu Zhang. Promised and Distributed Quantum Search. In Lusheng Wang, editor, *Computing and Combinatorics*, pages 430–439, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[29] Seiichiro Tani. Claw Finding Algorithms Using Quantum Walk. *Theoretical Computer Science*, 410(50):5285–5297, 2009.

[30] Lawrence Carter and Mark Wegman. Universal Classes of Hash Functions. In *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing*, STOC '77, page 106–112, New York, NY, USA, 1977. Association for Computing Machinery.

[31] Mark Wegman and Lawrence Carter. New Hash Functions and Their Use in Authentication and Set Equality. *Journal of Computer and System Sciences*, 22(3):265–279, 1981.

[32] Alex Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight Adaptive Reprogramming in the QROM. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 637–667, Cham, 2021. Springer International Publishing.

[33] Gorjan Alagic, Chen Bai, Jonathan Katz, and Christian Majenz. Post-Quantum Security of the Even-Mansour Cipher, 2021.

[34] Mark Zhandry. How to Record Quantum Queries, and Applications to Quantum Indifferentiability, 2018.