Evaluating the Security of Merkle-Damgård Hash Functions and Combiners in Quantum Settings

Zhenzhen Bao[1,3], Jian Guo[1], Shun Li[1,2], and Phuong Pham[1]

[1] School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore.
{guojian,shun.li,guozhen.liu}@ntu.edu.sg, pham0079@e.ntu.edu.sg
[2] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China.
[3] Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China.
zzbao@tsinghua.edu.cn

Abstract. In this work, we evaluate the security of Merkle-Damgård (MD) hash functions and their combiners (XOR and concatenation combiners) in quantum settings. Two main quantum scenarios are considered, including the scenario where a substantial amount of cheap quantum random access memory (qRAM) is available and where qRAM is limited and expensive to access. We first convert a rich set of known tools invented for generic attacks in the classical setting to quantum versions. That includes Joux's multi-collision, expandable message, diamond structure, and interchange structure. With these basic tools in hand, we then present generic quantum attacks on the MD hash functions and hash combiners, and carefully analyze the complexities under both quantum scenarios. The considered securities are fundamental requirements for hash functions, including the resistance against collision, (second-)preimage, and herding attacks. The results are consistent with the conclusions in the classical setting, that is, the considered resistances of the MD hash functions and their combiners are far less than ideal, despite the significant differences in the expected security bounds between the classical and quantum settings. Particularly, the generic attacks can be improved significantly using quantum computers under both scenarios. These results serve as an indication that classical hash constructions require careful security re-evaluation before being deployed to the post-quantum cryptography schemes.

Keywords: Merkle-Damgård, Hash Combiner, XOR, Concatenation, Quantum, Generic Attack

1 Introduction

In light of recent and projected progress in building quantum computers [18, 21], more and more quantum algorithms have recently been applied to cryptanalysis against classical cryptography systems to assess their security strength against quantum computers. In the past, most if not all crypto-systems were designed to resist attacks by conventional computers taking advantage of the limited computation power the real world may possess in the classical setting. In other words, these crypto-systems are only computationally secure, not information theoretically secure, under conventional computers. However, quantum computers have significant advantage of speedup computing (a.k.a. quantum supremacy) over conventional ones, which results in completely broken of some crypto-systems, and others with security strength weakened. For instance, Shor's factoring algorithm [31] is a powerful quantum algorithm to factorize an integer M in polynomial time with respect to the bit length of M, which can be used to break all current RSA standards and many other public-key crypto-systems. Therefrom, public-key crypto-systems have attracted a lot of attention from the research community and government agencies, e.g., the ongoing effort by NIST on post-quantum cryptography standardization [30]. On the other hand for symmetric-key cryptography, Grover's search algorithm [19] is able to find a marked data in an unstructured database of size N in just $O(\sqrt{N})$ time, vs. $O(N)$ for brute-force search in classical setting. This generally reduces the security strength in bits by half of most keyed symmetric-key crypto-systems, e.g., the secret key of AES-128 can be recovered within a complexity of roughly 2^{64} vs. 2^{128} in the classical setting by brute-force search.

In this paper, we re-assess the fundamental security properties, i.e., collision, preimage, and second-preimage resistance, of some hash constructions that have existed for long in the classical setting of the real world, under some quantum settings. We focus on *iterated* hash functions, in particular those following the Merkle-Damgård construction (MD) [12, 29], where a single compression function is called iteratively in order to extend the input domain from a fixed length to arbitrary length and the digest length is the same as that of internal state as for most of the standards like MD5, SHA-1, and SHA-2.

The security of hash constructions has been well studied in the classical setting in the past few decades. For Merkle-Damgård construction, it is known that the collision resistance of the hash function can be reduced to that of the underlying compression function [12, 29]. The existence of multi-collisions was formally introduced by Joux [24] in 2004, and the first generic second-preimage was found by Kelsey and Schneier [26] in 2005 and later improved by Andreeva *et al.* [3, 5]. Herding attack was introduced first by Kelsey and Kohno [25] in 2006 and was later improved by Andreeva *et al.* [4]. It is noted that second-preimage attacks and herding attacks are all utilizing collisions and hence complexities are well above birthday bound.

In the quantum setting, the security of these hash constructions has also received some investigations. In [34], Zhandry proved that the Merkle-Damgård construction with ideal (cannot be distinguished from a random oracle) underlying compression function cannot be distinguished from a random oracle with more than negligible advantage. In [20], Hosoyamada and Yasuda proved that Merkle-Damgård construction with Davies-Meyer (DM-mode) compression function is quantum one-way function, and the lower bound of the number of queries required by preimage attacks is $O(2^{n/2})$ — that given by the generic Grover's search algorithm. It is reckoned in [10] that similar proof to that in [20] could be done also with the MatyasMeyerOseas (MMO) mode compression function. These works provide provable security lower bound for the Merkle-Damgård constructions in quantum settings. Yet, the rich set of tools invented in previous work to do generic attacks, which provide security upper bound, on Merkle-Damgård hash constructions in classical settings still remain to be fully exploited in quantum settings.

Besides the single hash functions, we also re-evaluate the security of hash combiners in quantum settings. We focus on two typical hash combiners, *i.e.*, the concatenation combiner and the exclusive-or (XOR) combiner. Given two (independent) hash functions $\mathcal{H}_1$ and $\mathcal{H}_2$, the concatenation combiner returns $\mathcal{H}_1(M)\|\mathcal{H}_2(M)$, and the XOR combiner returns $\mathcal{H}_1(M) \oplus \mathcal{H}_2(M)$. In practice, people may wonder whether we can combine existing hash functions to achieve long term security instead of replacing existing infrastructure to new ones (in SSL v3 [17] and TLS 1.0/1.1 [13, 14], MD5 and SHA-1 were combined in various ways, including concatenation combiner and XOR combiner [16]). The main purpose of hash combiners might be to achieve *security amplification*, *i.e.*, the hash combiner offers higher security strength than its component hash functions, or to achieve *security robustness*, *i.e.*, the hash combiner remains secure as long as at least one of its component hash functions is secure. We know from the results of previous cryptanalyses that in the classical setting, the hash combiners are not as secure as expected (e.g., guarantee its security if either underlying hash function remains secure, or as secure as a single ideal hash function). Concretely, the attacks on XOR combiners by Leurent and Wang [28] in 2015 and on concatenation combiners by Dinur [15] in 2016 showed surprising weaknesses, which either contradicts the intended purposes of security robustness or security amplification. These results were then improved and summarized by Bao *et al.* in [6, 7]. However, some techniques used in previous cryptanalyses of hash combiners in the classical setting cannot be directly accelerated using quantum computers (e.g., those attacks on combiners exploiting properties of random functional graphs). Whereas generic attack is accelerated in the quantum setting, that is, the security upper bound of an ideal hash function is lower. Thus, the broken primitives (e.g., the investigated hash combiners) in the classical setting might be unbroken (no better attacks than the most generic attack) in the quantum setting. So, we investigate this question and aim to provide references.

1.1 Our Contributions

In this paper, we port most of the important and generic attacks in the classical settings against Merkle-Damgård construction and hash combiners, make adjustments of the attack algorithms whenever necessary, and carefully evaluate the complexities in the quantum setting. Table 1 summarizes detailed complexities. Surprisingly, most of the (second-)preimage attacks and herding attacks in the classical setting still constitute valid attacks in the quantum setting.

Target	Property	CS		Scenario $\mathcal{R}_1$		Scenario $\mathcal{R}_2$			Reference
		CTime	CMem	QTime	QMem	QTime	QMem	CMem	
$\mathcal{H}$	Collision	$2^{n/2}$	$O(1)$	$2^{n/3}$	$2^{n/3}$	$2^{2n/5}$	$O(n)$	$2^{n/5}$	[9, 11]
	Preimage	2^{n}	$O(1)$	$2^{n/2}$	$O(n)$	$2^{n/2}$	$O(n)$	$O(1)$	[19]
	2nd Preimage	$2^{n/2}$ [26]	$2^{n/2}$	$2^{n/3}$	$2^{n/3}$	$2^{3n/7}$	$O(n)$	$2^{3n/7}$	Sect. 4.2
	Herding	$2^{2n/3}$ [25]	$2^{n/3}$	$2^{3n/7}$	$2^{3n/7}$	$2^{11n/23}$	$O(n)$	$2^{7n/23}$	Sect. 4.3
$\mathcal{H}_1 \oplus \mathcal{H}_2$	Collision	$2^{n/2}$	$O(1)$	$2^{n/3}$	$2^{n/3}$	$2^{2n/5}$	$O(n)$	$2^{n/5}$	[9, 11]
	Preimage	$2^{11n/18}$ [6]	$2^{11n/18}$	$2^{10n/21}$	$2^{n/3}$	$2^{52n/105}$	$2^{n/7}$	$2^{n/5}$	Sect. 5.1
	2nd Preimage	$2^{11n/18}$ [6]	$2^{11n/18}$	$2^{10n/21}$	$2^{n/3}$	$2^{52n/105}$	$2^{n/7}$	$2^{n/5}$	Sect. 5.1
	Herding	$2^{2n/3}$ [4]	$2^{n/3}$	$2^{4n/9}$	$2^{n/3}$	$2^{24n/49}$	$2^{n/7}$	$2^{n/5}$	Sect. 5.3
$\mathcal{H}_1 \\| \mathcal{H}_2$	Collision	$2^{n/2}$ [24]	$O(n)$	$2^{n/3}$	$2^{n/3}$	$2^{3n/7}$	$2^{n/7}$	$2^{n/5}$	Sect. 5.2
	Preimage	2^{n} [24]	$O(n)$	$2^{n/2}$	$2^{n/3}$	$2^{n/2}$	$O(n)$	$2^{n/5}$	Sect. 5.2
	2nd Preimage	$2^{25n/34}$ [6]	$2^{25n/34}$	$2^{n/2}$	$2^{n/3}$	$2^{n/2}$	$O(n)$	$2^{n/5}$	Sect. 5.2
	Herding	$2^{2n/3}$ [4]	$2^{n/3}$	$2^{4n/9}$	$2^{n/3}$	$2^{24n/49}$	$2^{n/7}$	$2^{n/5}$	Sect. 5.3

CS: Classical Setting QTime: Quantum Time
QMem: Quantum Memory CMem: Classical Memory

Table 1: Security status of Merkle-Damgånd hash functions and hash combiners (polynomial factors are ignored for exponential complexities)

The attacks in quantum settings are divided into two scenarios, depending on whether cheaply accessible quantum random access memory is available or not, and they are named Scenario $\mathcal{R}_1$ and Scenario $\mathcal{R}_2$. Scenario $\mathcal{R}_1$ refers qRAM supporting access in constant time regardless of the size of the memory, while it costs $O(R)$ time for each access to quantum memory of size $O(R)$ and also linear time for each access to classical memory in Scenario $\mathcal{R}_2$.

This article is organized as follows. In the next Section 2, we introduces some basic notions and algorithms used in quantum computation. Section 4 and 5 are the demonstration of several attacks on Merkle-Damgård structures and hash combiners. Section 6 concludes the results and presents some open problems. We revise some important techniques for our attack belong with the quantum version of these techniques in Section 3.

2 Basic Quantum Algorithms for Collision and Search

In this section, we briefly introduce hash functions, hash combiners, qRAM, and quantum algorithms used throughout this paper.

2.1 Merkle-Damgård Hash Construction

Define $\mathcal{H}$ for a cryptographic hash function that maps arbitrarily long messages to an n bit digest, *i.e.*, $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$. Like most iterated hash functions, to hash a message M, the Merkle-Damgård (MD) construction first pads and splits the message bits into message blocks of fixed length (*e.g.*, b bits), *i.e.*, $M = m_1 \| m_2 \| \cdots \| m_L$, where the last message block m_L comprises the bit encoding of the original message length. Then, starting from a public initial value $IV = x_0$, the message block with the intermediate state is hashed by the same compression function H iteratively, *i.e.*, $x_i = h(x_{i-1}, m_i)$ for $i = 1, \ldots, L$ (see Fig 1). In the quantum setting, the MD hash functions are proven to be quantum one-way functions [20], while other security properties remain largely un-exploited in the quantum setting.

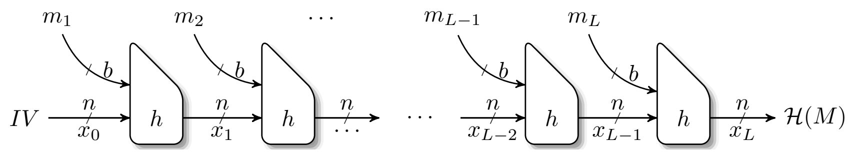


Figure 1: Merkle-Damgård hash function

The XOR combiner and concatenation combiner based hash functions following MD structure are demonstrated in the following figures.

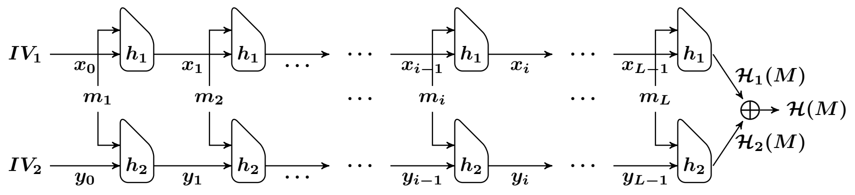


Figure 2: The XOR combiner

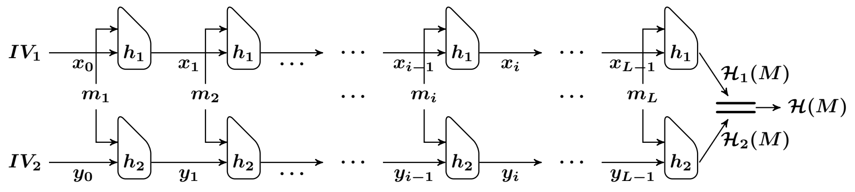


Figure 3: The concatenation combiner

2.2 QRAM

Quantum random access memory (qRAM) can be considered as a quantum counterpart of random access memory (RAM) from the classical setting, which allows accessing (read or write) the elements in memory with constant time regardless of storage size. There are two types of qRAM: *quantum-accessible classical memory* (QRACM), which allows to access the classical data in quantum superpositions, and *quantum-accessible quantum memory* (QRAQM), where the data is stored in quantum memory. Suppose that we want to store a list of data (classical or quantum) $D = (x_0, x_1, \cdots, x_{2^k-1})$, where x_i is an n-bit data. Then the *qRAM* for accessing data D is constructed as a quantum gate and defined via a unitary operator $U_{\text{qRAM}}(D)$ by

$$U_{\text{qRAM}}(D) : |i\rangle |y\rangle \mapsto |i\rangle |y \oplus x_i\rangle$$

Property	Classical Setting		Quantum Setting				
	CTime	CMem	QTime	qRAM	CMem	Optimal	Reference
Collision	$2^{n/2}$	$O(1)$	$2^{n/3}$	$2^{n/3}$	-	YES	Scenario $\mathcal{R}_1$ [9, 33]
			$2^{2n/5}$	$O(n)$	$2^{n/5}$	unknown	Scenario $\mathcal{R}_2$ [11]
Preimage	2^n	$O(1)$	$2^{n/2}$	$O(n)$	$O(1)$	YES	Scenario $\mathcal{R}_2$ [19, 32]

Table 2: Comparison of security upper bounds of ideal hash functions in classical and quantum settings (polynomial factors are ignored for exponential complexities).

where $i \in \{0,1\}^k$ and y is an n-bit value. Since qRAM is a powerful model with requirement of specific physical architecture, many quantum algorithms take advantage of it to reduce time complexity, such as the algorithm for collision search [9] requires QRACM and element distinctness [2] requires QRAQM. Though qRAM is still a controversial issue, it is essential to evaluate the cryptography systems in the scenario that qRAM is big and cheap to access (we will call this quantum model as Scenario $\mathcal{R}_1$). On the other hand, a relatively more realistic model is to assume that qRAM is costly and accessing to R quantum qubit memory costs $O(R)$ time as in [11, 22] (we will call this quantum model as Scenario $\mathcal{R}_2$). We will analyze the complexities of our attacks in both Scenario $\mathcal{R}_1$ and Scenario $\mathcal{R}_2$ with respective optimal choices of attack parameters.

2.3 Grover's Search Algorithm

The quantum algorithm for searching a marked point in a database is firstly introduced by Grover in [19]. In 1999, Zalka [32] proved that Grover's algorithm is optimal for the searching problem. It considers the following problem.

Problem 1. Let F be a Boolean function, $F : \{0,1\}^n \to \{0,1\}$. Suppose that there is only one x such that $F(x) = 1$. Then, find x.

In the classical setting, the number of queries to find x is approximately 2^n, while Grover's algorithm can find x by making only $O(\sqrt{2^n} = 2^{n/2})$ queries. That is, in the quantum setting, the time complexity for the database search problem is quadratic faster than the classical ones. Due to the optimality of the algorithm, the $2^{n/2}$ complexity is the tight security level of preimage resistance of hash functions in quantum setting, as summarized in Table 2.

Some variants of Problem 1 involve the general case with $|\{x : F(x) = 1\}| = 2^t$. Then, with high probability, Grover's algorithm returns x after making $O(\sqrt{2^n/2^t})$ quantum queries to F.

2.4 Quantum Collision Finding Algorithms

Brassard, Høyer, and Tapp in [9] first introduced a quantum algorithm (so-called BHT algorithm) to find a collision for a (2-to-1) random function in time $O(2^{n/3})$ and $O(2^{n/3})$ quantum queries, with an additional assumption that quantum random access memory (qRAM) is available. Subsequently, Zhandry in [33] extended this result to any random function with the size of the domain at least the square root of the size of the codomain, which is more relevant for hash functions or permutations in cryptographic settings. It considers the following problem.

Problem 2. Let $H : \{0,1\}^n \to \{0,1\}^n$ be a random function. Find x and x' such that $H(x) = H(x')$.

In the classical setting, finding collisions of a random function in range $\{0,1\}^n$ can be done after making $O(2^{n/2})$ queries, following the Birthday Paradox. While the BHT algorithm makes use of Grover's algorithm to find a collision in $O(2^{n/3})$ queries. Due to the optimality of the algorithm, $2^{n/3}$ is also the tight security level of the collision resistance of hash functions, in Scenario $\mathcal{R}_1$. In this paper, we consider the situation where qRAM is available, and the BHT algorithm can be applied efficiently for the collision finding problem of hash functions.

Scenario $\mathcal{R}_2$. In this situation, each lookup operation within the memory of size $O(2^{n/3})$ costs $O(2^{n/3})$ time, hence resulting in an inefficient algorithm even slower than the birthday attack. Chailloux et al. [11] proposed an efficient algorithm (denoted by CNS) to find a collision of hash function in time $\tilde{O}(2^{2n/5})$ with a quantum computer of $O(n)$ qubits, but large classical memory of size $\tilde{O}(2^{n/5})$.

2.5 Quantum Walk Algorithm for the Element Distinctness Problem

In the quantum setting, it is proven in [1] that the number of quantum queries for solving this problem is at least $O(N^{2/3})$. Up to now, only one algorithm, named as the *quantum walk algorithm* proposed in [2] reaches this bound. Recall this quantum walk algorithm for the following problem.

Problem 3. Given a set $S = \{x_1, x_2, ..., x_N\}$, does it exist i, j such that $1 \leq i < j \leq N$ and $x_i = x_j$? If yes, return i, j.

The element distinctness problem cannot be solved by an algorithm more efficiently than a brute force approach in the classical setting. This is because, only after $O(N)$ queries and sorting can one find two elements of the same value in a set of N elements. The Ambainis's quantum walk algorithm makes $O(N^{2/3})$ queries and requires $O(N^{2/3}\log N)$ qubits memory.

Scenario $\mathcal{R}_2$. The Ambainis's quantum walk algorithm for element distinctness problem can work efficiently and better than other algorithms in the scenario where the qRAM is available and it costs constant time to access qRAM gates (*i.e.*, Scenario $\mathcal{R}_1$). Very recently, to tackle with the situation that qRAM is not cheap and accessing R qubits quantum memory costs $O(R)$ operators or quantum gates, Jaques and Schrottenloher in [22] improved the quantum walk algorithm for golden collision problem (a more general case of the element distinctness problem), there the new algorithm requires $O(N^{6/7})$ computations and $O(N^{2/7})$ quantum memory, without using the qRAM. More explicitly, the assumption on the memory model in the quantum walk algorithm in [22] is that quantum memory is costly to access but free to maintain, which seems more realistic than Scenario $\mathcal{R}_1$. Thus, in this paper, when discussing the complexities of the presented attacks that calling a quantum walk algorithm in Scenario $\mathcal{R}_2$, we follow this assumption.

3 Collision-Search-Based Tools and Their Quantum Versions

In this section, we introduce several collision-search-based tools commonly used in generic attacks in classical settings. For each of them, we discuss how to transform it into a tool in quantum settings and re-evaluate the complexity. In the sequel, we denote by $\mathcal{H}$ an MD hash function, h for its compression function, and h^* for arbitrary times of iteration on h.

3.1 Multi-Collision (MC [24]).

Joux in [24] proposes an efficient way to obtain a large set of messages mapping a starting state to a common ending state on iterated hash functions, which is known as Joux's multi-collisions.

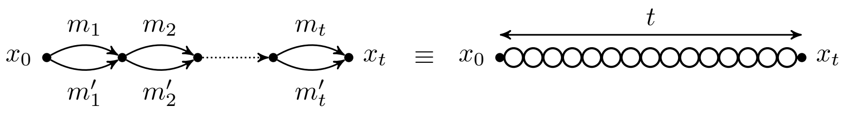


Figure 4: Multi-collision and its condensed representation in R.H.S. [23]

Multi-Collision (MC) in Quantum Settings. In Scenario $\mathcal{R}_1$, the t birthday attacks for finding t collisions to build a 2^t-$\mathcal{M}_{\texttt{MC}}$ can be done by calling t times of BHT algorithm. As a result, the total complexity, which is $t \cdot 2^{n/2}$ in the classical setting, is $t \cdot 2^{n/3}$ in the quantum setting. The quantum counterpart of building a 2^t-$\mathcal{M}_{\texttt{MC}}$ is given in Algorithm 1.

The complexity of Algorithm 1 is dominated by calling the BHT algorithm t times; hence, it requires

Algorithm 1: Building a 2^t-Joux's MC in Quantum Settings

```
Require: Given an oracle of the compression hash function $h$, an initial value $x_0$ and qRAM.
1. Initialize the data structure $\mathcal{M}_{\texttt{MC}}$ to store pairs of message blocks.
2. For $i = 1, ..., t$:
   (a) Start a BHT algorithm by querying $2^{n/3}$ message blocks $m'_j$ to the oracle of $h$, sort according to the second
       entry and store all the pairs in list $L$, if $L$ contains a collision, output the collision immediately.
       Store all pairs $(m'_j, h(x_{i-1}, m'_j))$ in $L$ to qRAM.
       Construct the oracle: $F : \{0,1\}^n \to \{0,1\}$ by defining $F(m) = 1$ if and only if there exist $(m'_j, h(x_{i-1}, m'_j))$
       in qRAM such that $h(x_{i-1}, m'_j) = h(x_{i-1}, m)$ and $m'_j \neq m$.
   (b) In the BHT algorithm, apply the Grover's search algorithm using oracle $F$:
       i. Initialize the state of the Grover's search to be the uniform superposition of $2^n$ messages;
       ii. After running about $\frac{\pi}{4} \cdot 2^{n/3}$ Grover steps, measure the state and return a pair of message blocks
           $(m_i, m'_i)$ such that $h(x_{i-1}, m_i) = h(x_{i-1}, m'_i)$.
   (c) Obtain $x_i = h(x_{i-1}, m_i)$, append $(m_i, m'_i)$ to $\mathcal{M}_{\texttt{MC}}$.
3. Output $(x_t, \mathcal{M}_{\texttt{MC}})$.
```

$O(t \cdot 2^{n/3})$ quantum queries, $O(t \cdot 2^{n/3})$ computations, and $O(2^{n/3})$ qRAM.

In Scenario $\mathcal{R}_2$, we can replace the BHT algorithm with the algorithm in [11], which requires $O(2^{2n/5})$ computations and $O(2^{n/5})$ classical memory. Then, the resulted quantum algorithm 1 requires $O\left(t \cdot 2^{2n/5}\right)$ quantum queries and $O(2^{n/5})$ classical memory.

Note that this quantum version of the Joux's multi-collision will be used in building more complex structures (interchange structure in Sect. 3.4), and in the presented preimage attacks (Sect. 5.1 and 5.2).

3.2 Expandable Message (EM [26]).

Kelsey and Schneier in [26] invented the *expandable message*, which is similar to Joux's multi-collision. By generating t collisions with pairs of message fragments of length $(1, 2^i + 1)$ for $i \in \{0, 1, \ldots, t-1\}$, one can get 2^t colliding messages whose lengths cover the range of $[t, t + 2^t - 1]$ (see Fig. 5). The complexity is of $2^t + t \cdot 2^{n/2}$ computations. This expandable message can be used to bypass the Merkle-Damgård strengthening and carry out a long message second-preimage attack on MD with roughly $2^n/L$ computations for a given challenge of L blocks.

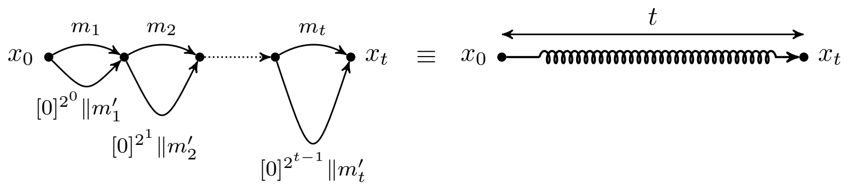


Figure 5: Expandable message and its condensed representation in R.H.S. [23]

Expandable Message (EM) in Quantum Settings. Since the main idea of building a 2^t-expandable message is finding the collision between a message of a single block and a message of length $2^i + 1$ for $0 \le i \le t-1$, this step can be done by applying the BHT algorithm in quantum setting. Similar to finding collisions in quantum setting for building Joux's multi-collision, for each i, we calculate the hash value x^*_{i-1} of message $[0]^{2^i}$ from the hash value x_{i-1} , and find a pair of message blocks (m_i, m'_i) such that $h(x_{i-1}, m_i) = h(x^*_{i-1}, m'_i) = x_i$. Then the constructing a message of length $s \in [t, t + 2^t - 1]$ step is proceeded in the same way as in the classical setting, as we look at the decomposition of $s - t$ in t-bit binary base. We select the long message $[0]^{2^i}\|m'_i$ in the iteration i if the i-th LSB of $s - t$ is equal to 1, otherwise, we select the single block message m_i instead. The complexity of this quantum algorithm is different from classical expandable message algorithm just by the collision search step; hence, it is of $2^t + t \cdot 2^{n/3}$ quantum computations in Scenario $\mathcal{R}_1$, or of $2^t + t \cdot 2^{2n/5}$ quantum computations using CNS algorithm in Scenario $\mathcal{R}_2$.

This quantum version of the expandable message will be used in the presented quantum second-preimage attack on the MD hash function (Sect. 4.2).

3.3 Diamond Structure (DS [25]).

Kelsey and Kohno in [25] invented the diamond structure. Similar to Joux's multi-collisions and Kelsey and Schneier's expandable message, diamond is also a kind of multi-collision. The difference is that, instead of mapping a single starting state to a final state in the form of sequential chain, a 2^t-diamond maps a set of 2^t starting states to a common final state in the form of a complete binary tree (see Fig. 6). Blackburn in [8] pointed out that the construction method and its complexity provided in [25] have a flaw, and offered a more rigorous analysis and construction method. The method in [8] requires $O(\sqrt{t} \cdot 2^{\frac{(n+t)}{2}})$ message blocks and $n \cdot \sqrt{t} \cdot 2^{\frac{(n+t)}{2}}$ computations, and will be converted into quantum method later in this section. Kortelainen and Kortelainen in [27] presented another method for constructing the diamond structure. The new method could reduce the message requirement to $O(2^{\frac{(n+t)}{2}})$. However, it becomes more intricate by separating the procedure into jumps, phases, and steps. During different phases and steps, different number of new messages are added and old messages are recycled, which makes the phases and steps more dynamic and the workloads are not balanced compared with previous methods.

Diamond is originally used in herding attacks on hash functions [25]. In [3, 4, 5], Andreeva *et al.* exploited the diamond structure to develop generic second-preimage attacks on Dithered hash function and Hash-Twice. Besides, the diamond structure was also used to device a second-preimage attack on Merkle-Damgård hash function with shorter messages than that in the long-message second-preimage attack in [26].

Diamond Structure (DS) in Quantum Settings. We adapt the construction method in [8] to build the diamond in the quantum setting. The framework is to build the complete binary tree of the diamond with given 2^t states as the leaves, layer by layer. The following description in Algorithm 2 focuses on the construction from one layer to the next, and takes the first two layers for example.

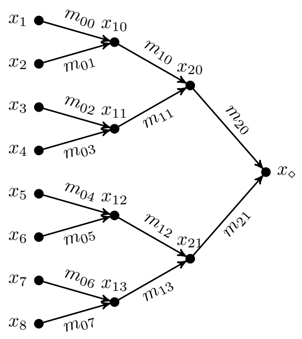


Figure 6: A 2^3-diamond

In Algorithm 2, according to [8], to find a perfect matching[4] in $\mathcal{G}$, the probability p for each pair of vertices being connected by an edge should be no less than $(\ln 2^t)/2^t \approx t\cdot 2^{-t}$. So, for each state x_i, the required number of other states that can lead to a collision with x_i is t. At this condition, we repeat Grover's algorithm t times for each state in Step 2. Then, the probability for each pair of (x_i, x_j) being mapped to a collision is $p \approx (t\cdot(L\cdot S))/2^n$. That requires $p \approx (t\cdot(L\cdot S))/2^n = (t\cdot 2^{n-t})/2^n$. That is, $L\cdot S \approx 2^{n-t}$. Let $\begin{cases} L = t^{d_1}\cdot 2^{\ell}, \\ S = t^{d_2}\cdot 2^{s}, \end{cases}$ then $\begin{cases} d_1 + d_2 = 0, \\ \ell + s = n - t. \end{cases}$ To balance the complexity of Step 1 and Step 2, we set $2^t\cdot t\cdot\sqrt{S} = 2^t\cdot L$, that is, $\begin{cases} 2 + d_2 = 2d_1, \\ s = 2\ell. \end{cases}$ Accordingly, we have $\begin{cases} d_1 = 2/3, \ \ \ell = (n-t)/3 \\ d_2 = -2/3, \ \ s = 2(n-t)/3. \end{cases}$ Therefore, $\begin{cases} L = t^{2/3}\cdot 2^{(n-t)/3} \\ S = t^{-2/3}\cdot 2^{2(n-t)/3}. \end{cases}$ As a conclusion, using the above method in Scenario $\mathcal{R}_1$, the total time complexity for building t layers of a 2^t-diamond is $O(t^{2/3}\cdot 2^{(n+2t)/3})$, and memory complexity is $O(t^{2/3}\cdot 2^{(n+2t)/3})$ qRAM.

In Scenario $\mathcal{R}_2$, the time complexity to find a collision is of $(2^{n-t})^{2/5}$ computations. Therefore, building a 2^t-diamond structure requires $O(t^{2/3}\cdot 2^t\cdot 2^{2(n-t)/5}) = O(t^{2/3}\cdot 2^{(2n+3t)/5})$ computations, with $O(t^{2/3}\cdot 2^t\cdot 2^{(n-t)/5}) = O(t^{2/3}\cdot 2^{(n+4t)/5})$ classical memory.

This quantum version of the diamond structure will be used in the presented quantum herding attack on the MD hash function (Sect. 4.3) and the quantum herding attack on combiners (Sect. 5.3).

3.4 Interchange Structure (IS [28]).

Leurent and Wang in [28] invented the interchange structure, which is used to devise a preimage attack on the XOR combiner. The interchange structure contains a set of messages $\mathcal{M}_{\texttt{IS}}$ and two sets of states $\mathcal{A}$ and $\mathcal{B}$, such that for any pair of states $(A_i, B_j \mid A_i \in \mathcal{A},\ B_j \in \mathcal{B})$, one can pick a message M from $\mathcal{M}_{\texttt{IS}}$ such that $A_i = \mathcal{H}_1(IV_1, M)$ and $B_i = \mathcal{H}_2(IV_2, M)$. To build a 2^t-interchange structure (with 2^t states for each hash function), one can cascade $2^{2t} - 1$ building modules named switches. The effect of a switch is that a state in one computation chain of one hash function can make pair with two states in two computation chains of the other hash function. A switch can be built using multi-collisions and the birthday attack (see Fig. 7a). The total complexity to build a 2^t-interchange structure is of $\tilde{O}(2^{2t+n/2})$ computations.

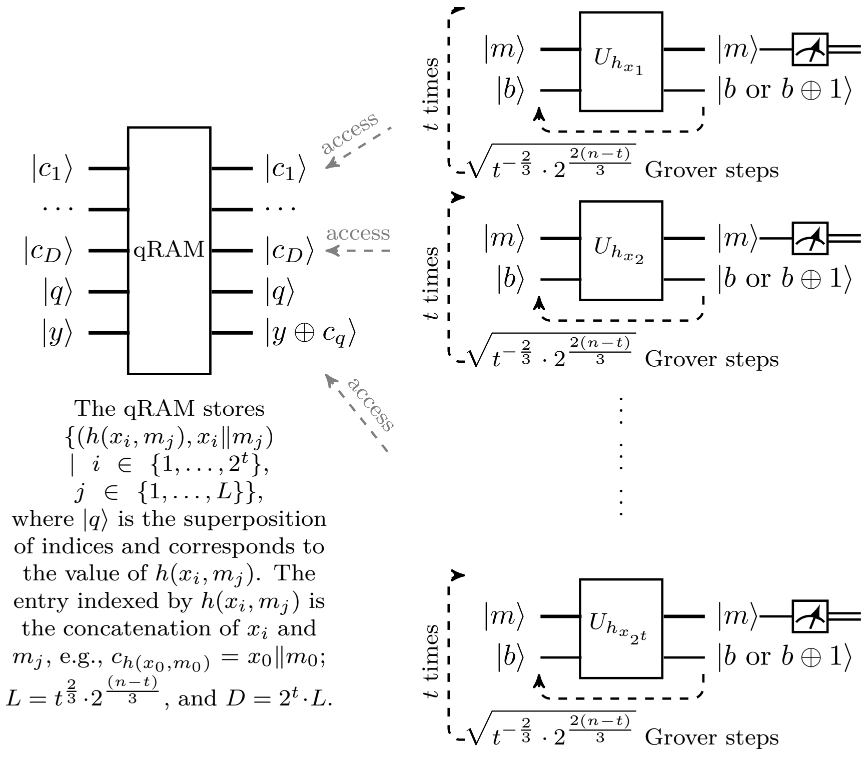
[4] In graph $\mathcal{G}$, if there exists a set of edges, no two of which share a vertex, then the set of edges is called a matching. M is a maximum matching in $\mathcal{G}$ if no matching in $\mathcal{G}$ contains more edges than M does. If matching M in $\mathcal{G}$ contains every vertex, then M is called a perfect matching. Our goal here, is to find a perfect matching in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, of which the vertex set is $\mathcal{V} = \{x_1, \ldots, x_{2^t}\}$ and $(x_i, x_j) \in \mathcal{E}$ if x_i and x_j generate an obtained collision.

Algorithm 2: Building One Layer of Diamond in Quantum Settings

Denote by $X = \{x_1, \dots, x_{2^t}\}$ the set of states at the first layer of the diamond.
Let $M = \{m_1, \dots, m_L\}$ be a space of L message blocks.

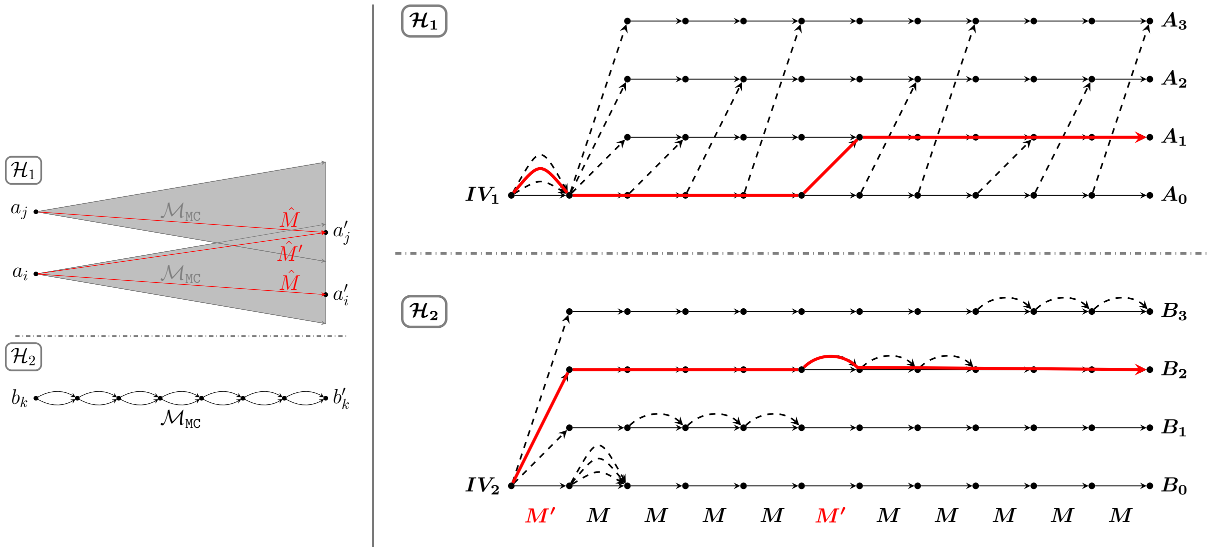
1. For each $x_i \in X$, compute a table $T_i = \{(h(x_i, m_j), (x_i \| m_j)) \mid m_j \in M\}$ with the hash values as the index and the concatenation of x_i and m_j as the contents of entries. Combine all 2^t tables into a single table without merging and store in the qRAM.
2. For each $x_i \in X$, repeat the following t times:
   (a) Let $M' = \{m'_1, \dots, m'_S\}$ be a new space of S message blocks.
   (b) Initialize the uniform superposition state of message blocks in space M'.
   (c) Use Grover's algorithm to search for a collision between the set of hash values $\{h(x_i, m'_j) \mid m'_j \in M'\}$ and the set of $2^t \cdot L$ values store in the qRAM. Each step in Grover's algorithm, after accessing qRAM, it returns $|b \oplus 1\rangle$ if $h(x_i, m)$ collides with some element in qRAM, otherwise, it returns $|b\rangle$.

   With the collisions obtained by the quantum computation, construct the associated graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, of which the vertex set is $\mathcal{V} = \{x_1, \dots, x_{2^t}\}$ and $(x_i, x_j) \in \mathcal{E}$ if x_i and x_j generate an obtained collision. This is same as did in the last part of Step 2 of the method in the classical setting (see [8]).
3. Find the perfect matching contained in $\mathcal{G}$ same as Step 3 of the method in the classical setting (see [8]).


$|c_1\rangle$
$\cdots$
$|c_D\rangle$
$|q\rangle$
$|y\rangle$
qRAM
$|c_1\rangle$
$|c_D\rangle$
$|q\rangle$
$|y \oplus c_q\rangle$
access
access
access
The qRAM stores $\{(h(x_i, m_j), x_i \| m_j) \mid i \in \{1, \dots, 2^t\}, j \in \{1, \dots, L\}\}$, where $|q\rangle$ is the superposition of indices and corresponds to the value of $h(x_i, m_j)$. The entry indexed by $h(x_i, m_j)$ is the concatenation of x_i and m_j, e.g., $c_{h(x_0,m_0)} = x_0 \| m_0$; $L = t^{\frac{2}{3}} \cdot 2^{\frac{(n-t)}{3}}$, and $D = 2^t \cdot L$.
$|m\rangle$
$|b\rangle$
$U_{h_{x_1}}$
$U_{h_{x_2}}$
$U_{h_{x_{2^t}}}$
$|b \text{ or } b \oplus 1\rangle$
t times
$\sqrt{t^{-\frac{2}{3}} \cdot 2^{\frac{2(n-t)}{3}}}$ Grover steps

Interchange Structure (IS) in Quantum Settings. The interchange structure starts with building a single switch, which is constructed by building a $2^{n/2}$-Joux's multi-collision for the hash function $\mathcal{H}_2$ and finding a collision between the hash value of $\mathcal{H}_1$ from different states (a_i, a_j) and some pair of message $(\hat{M}, \hat{M}')$. These two steps can be replaced by the quantum algorithm for building Joux's multi-collisions and the quantum walk algorithm for the element distinctness problem. The quantum algorithm for building a single switch is described as follows in Algorithm 3.

In Scenario $\mathcal{R}_1$, the complexity of Algorithm 3 is dominated by the building a multi-collision in Step 1, since Step 2 requires $O((2^{n/2+1})^{2/3}) = O(2^{n/3})$ quantum computations and $O(2^{n/3})$ quantum



(a) Building a switch **(b)** Interchange structure

Figure 7: Interchange structure and its building block

Algorithm 3: Building a Single Switch in Quantum Settings

1. Use the quantum Joux's multi-collision algorithm to build a set $\mathcal{M}_{\text{MC}}$ of $2^{n/2}$ messages for h_2^* that link the starting state b_k to the same state b'_k, *i.e.*, $\forall M \in \mathcal{M}_{\text{MC}}$, $h_2^*(b_k, M) = b'_k$.
2. Use a quantum walk algorithm to find a collision in the set of $2^{n/2+1}$ elements which are $h_1^*(a_i, M)$ and $h_1^*(a_j, M)$ for all the messages M in $\mathcal{M}_{\text{MC}}$. With high probability (constant), the algorithm return a pair of messages denoted as (M_i, M'_i) that $h_1^*(a_i, M_i) = h_1^*(a_j, M'_i)$.
3. Use the message M_i to compute the missing chains: $b'_j = h_2^*(b_j, M_i)$, $a'_j = h_1^*(a_j, M_i)$. With high probability, all the chains reach distinct values; if not, restart the algorithm with a new multi-collision.

memory. Hence, Algorithm 3 requires $O\left(\frac{n}{2} \cdot 2^{n/3}\right)$ quantum queries to the compression functions, $O\left(\frac{n}{2} \cdot 2^{n/3}\right)$ quantum time and $O(2^{n/3})$ quantum memory.

In Scenario $\mathcal{R}_2$, Step 1 needs $O\left(\frac{n}{2} \cdot 2^{2n/5}\right)$ quantum computations and $O(2^{n/5})$ classical memory, but when it comes to Step 2, the number of computations is higher, that is, $O((2^{n/2+1})^{6/7} = O(2^{3n/7})$ quantum computations and $O((2^{n/2})^{2/7}) = O(2^{n/7})$ quantum memory. Therefore, in this model, the time complexity for Algorithm 3 to build a single switch is of $O(2^{3n/7})$.

The framework for building a 2^t-interchange structure in quantum setting is the same as in the classical setting. One builds the required $2^{2t} - 1$ switches as the following: first, build a single switch from (a_0, b_0) to each of (a_0, b_k); then, for each k, build switches from (a_0, b_k) to all (a_j, b_k) for all $j = 0, \dots, 2^t - 1$. To reach the chain (a_j, b_k) from (a_0, b_0), we first find the switch to jump from (a_0, b_0) to (a_0, b_k) in the first step, then find the switch to jump from (a_0, b_k) to (a_j, b_k) in the second step. Then the complexity to build an interchange structure is $O\left(\frac{n}{2} \cdot 2^{2t+n/3}\right)$ for both quantum queries and time and $O(2^{n/3})$ quantum memory in Scenario $\mathcal{R}_1$, or $O(2^{2t+3n/7})$ and $O(2^{n/5})$ classical memory, $O(2^{n/7})$ quantum memory in Scenario $\mathcal{R}_2$.

This quantum version of the interchange structure will be used in the presented quantum preimage attack on the XOR-combiners (Sect. 5.1).

4 Security of Merkle-Damgård Structure in Quantum Settings

In this section, we explicate baselines for the security of Merkle-Damgård hash functions with respect to basic requirements in quantum settings, considering both Scenario $\mathcal{R}_1$ and Scenario $\mathcal{R}_2$. That includes the resistance against multi-collision, preimage, second-preimage attacks, and herding attacks.

4.1 Multi-Collision Attack

For the multi-collision attack on the Merkle-Damgård structure, as has been introduced in Sect. 3, following Joux's method and using BHT algorithm for each collision search, finding 2^t-collisions requires $O(t \cdot 2^{n/3})$ quantum computations and $O(2^{n/3})$ qRAM in Scenario $\mathcal{R}_1$. Since the time complexity to find a collision of any hash function is $O(2^{n/3})$ in Scenario $\mathcal{R}_1$, we can see that, same as in the classical setting, the quantum security of MD structure against multi-collision attack is only polynomial higher than the collision resistance of its compression function. In Scenario $\mathcal{R}_2$, 2^t-collisions of an MD hash function can be obtained by combining Joux's method and CNS algorithm with time complexity $O(t \cdot 2^{2n/5})$ and requires $O(2^{n/5})$ classical memory.

4.2 Preimage and Second-Preimage Attack

For an n-bit hash function, a security upper bound with respect to (second-) preimage attack in the quantum setting is directly provided by a plain Grover's algorithm, that is $O(2^{n/2})$ quantum computations. Thus, only attacks with complexity lower than the Grover's search algorithm can be seen as successful attacks. For the preimage resistance of MD hash construction, we cannot achieve better attacks than a plain Grover's search on an ideal hash. For the second-preimage resistance of MD hash construction, basing on the long-message second-preimage attack in [26], one can launch a quantum attack with the complexity lower than the generic Grover's attack.

Given message M_{target} of length $2^k + k + 1$, the goal is to find a second-preimage whose hash value is equal to that of the M_{target}. The quantum attack is described in Algorithm 4.

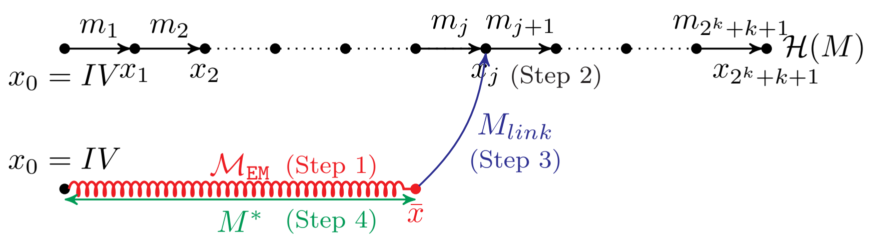
Attack in Scenario $\mathcal{R}_1$. The total complexity includes the complexity to build the expandable message with $2^k + k \cdot 2^{n/3}$ computations, $O(2^k)$ evaluations of compression function to compute the intermediate hash values of M_{target} and $\pi/4 \cdot 2^{(n-k)/2}$ evaluations to find M_{link}. Therefore, the total workload to find a second-preimage for a given message of length $2^k + k + 1$ is $2^{k+1} + k \cdot 2^{n/3} + \pi/4 \cdot 2^{(n-k)/2}$ quantum computations. Since the complexity of this attack in the classical setting is about $k \cdot 2^{n/2+1} + 2^k + 2^{n-k+1}$, the quantum version speeds up the attacks in classical setting when the given message is of length less than $2^{n/2}$.

The best-case Complexity. The minimum attack complexity is achieved when $\frac{n}{3} = \frac{n-k}{2}$, *i.e.*, $k = \frac{n}{3}$. Therefore, the second-preimage attack for a long message of length $O(2^{n/3})$ requires $O(n \cdot 2^{n/3})$ quantum computations and $O(2^{n/3})$ quantum memory. This complexity is only higher than that of the collision attack by BHT algorithm by a polynomial factor.

Attack in Scenario $\mathcal{R}_2$. The set of expandable messages can be built with $2^k + k \cdot 2^{2n/5}$ quantum computations, using $O(2^{n/5})$ classical memory. In Step 2, we store 2^k intermediate hash values of M_{target} to classical memory. In Step 3, different from using the Grover's algorithm as in Scenario $\mathcal{R}_1$, we apply the multi-target preimage search algorithm in [11] to search for message block M_{link}. The other steps do not change in this model, then the total work can be done in time $2^{k+1} + k \cdot 2^{2n/5} + 2^{n/2-k/6} + 2^k$.

Algorithm 4: Second-Preimage Attack on MD Hash in Quantum Settings

```
1. Build a set of expandable messages to cover the whole range of $[k, k+2^k-1]$ using the quantum algorithm
   as described in Sect. 3.2. Denote this set by $\mathcal{M}_{\mathtt{EM}}$, and the hash value after processing expandable message in
   $\mathcal{M}_{\mathtt{EM}}$ by $\bar{x}$.
2. Let $x_0 = IV$, $M_{target} = m_1\|m_2\|\cdots\|m_{2^k+k+1}$.
   Compute $x_i = h(x_{i-1}, m_i)$ for $i$ from 1 to $2^k+k+1$.
   This step is to compute $2^k$ intermediate hash values of $M_{target}$ and store results $x_{k+1}\dots x_{2^k+k+1}$ to qRAM.
3. Use Grover's algorithm to find a message block to link the iterated hash value of expandable message to one
   of the intermediate hash values of $M_{target}$, i.e. find $M_{link}$ such that $h(\bar{x}, M_{link}) = x_j$ for some $j$. Since the
   probability of the appearance of $M_{link}$ is $2^{k-n}$, we proceed $\pi/4 \cdot 2^{(n-k)/2}$ Grover steps before measure the
   superposition state to get $M_{link}$.
4. Find a message $M^*$ of length $j-1$ in $\mathcal{M}_{\mathtt{EM}}$.
5. Return the second-preimage $M^*\|M_{link}\|m_{j+1}\|\cdots\|m_{2^k+k+1}$
```



The best-case Complexity. The best-case complexity of this attack in Scenario $\mathcal{R}_2$ is achieved when $k = \dfrac{n}{2} - \dfrac{k}{6}$, *i.e.*, $k = \dfrac{3n}{7}$. The optimal time complexity is $O(2^{3n/7})$, with classical memory of size $O(2^{3n/7})$.

4.3 Herding Attack

The herding attack on Merkle-Damgård constructions is first introduced in [25], which is a special form of chosen-target preimage attack. In this attack, an adversary chooses a public hash value h_T, and then, she is challenged with a prefix P. Her goal is to find a suffix S such that $h_T = H(P\|S)$. Since h_T is chosen by the adversary, she can specifically choose it after some pre-computations, such as the root value of a diamond structure built on the compression function h.

In the following, we extend the classical herding attack to its quantum version, which uses the diamond structure and Grover's search algorithm. The attacker first builds the diamond structure by a quantum computer. In this diamond structure, from any of the intermediate hash values, she can produce a message which leads to the same final hash value h_T. She then publicizes h_T to commit. After receiving the challenged message P, the attacker applies Grover's algorithm to find a suffix message block S. The detailed attack is described in Algorithm 5.

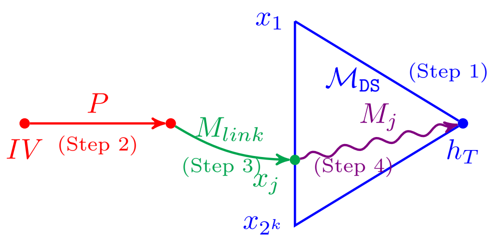
Attack in Scenario $\mathcal{R}_1$. The total complexity of the herding attack is $k^{2/3} \cdot 2^{(n+2k)/3} + 2^{(n-k)/2}$ quantum computations, with $O(k^{2/3} \cdot 2^{(n+2k)/3})$ quantum memory.

The best-case Complexity. The best complexity is achieved when $\dfrac{n+2k}{3} = \dfrac{n-k}{2}$, *i.e.* $k = \dfrac{n}{7}$, which results in the optimal $\tilde{O}(2^{3n/7})$ quantum computations.

Attack in Scenario $\mathcal{R}_2$. In this model, the 2^k-diamond structure can be built with time complexity of $O(k^{2/3} \cdot 2^{(2n+3k)/5})$; and the search of M_{link} can be done by using multi-target preimage algorithm

Algorithm 5: Herding Attack on MD Hash in Quantum Settings

1. Build the diamond structure using the quantum algorithm describe in Sect. 3.3: from 2^k starting hash values $D = \{x_i\}_{i=1}^{2^k}$ to the root value h_T. This step can be done in $O(k^{2/3} \cdot 2^{(n+2k)/3})$ computations. Commit h_T and publicize it.
2. Receive the challenged prefix: P.
3. Find a linking message: apply Grover's algorithm to search for a single block message M_{link} such that the value $h(P \| M_{link})$ collides with some value x_j in D. This step can be done in $O(2^{(n-k)/2})$ quantum queries and returns M_{link}.
4. Produce the message: $M = P \| M_{link} \| M_j$ where M_j is a sequence of message blocks linking x_j to h_T following the diamond structure built before.



with time complexity of $O(2^{n/2-k/6})$. Then the total complexity is $O(k^{2/3} \cdot 2^{(2n+3k)/5} + 2^{n/2-k/6})$ quantum computations, with $O(k^{2/3} \cdot 2^{(n+4k)/5})$ classical memory.

The best-case Complexity. The optimal time complexity is achieved when $\frac{2n+3k}{5} = \frac{n}{2} - \frac{k}{6}$, *i.e.*, $k = \frac{3n}{23}$, which results in $\tilde{O}(2^{11n/23})$ time and $\tilde{O}(2^{7n/23})$ classical memory.

5 Security of Hash Combiners in Quantum Settings

In this section, we present quantum attacks on hash combiners. For preimage, second-preimage, and herding attacks, the ideal quantum security are all $2^{n/2}$ (resp. 2^n) for XOR (resp. concatenation) combiners, which are bounded by attacks directly using Grover's search algorithm. For collision attack, the ideal quantum security bound is $2^{n/3}$ (resp. $2^{2n/3}$) for XOR (resp. concatenation) combiners, which is provided by the BHT's algorithm.

In the following, we present a quantum preimage attack on XOR combiners, which provides updated security upper bound in quantum settings for its resistance against (second-) preimage attack. We then present quantum collision, (second-) preimage attacks, and herding attacks on concatenation combiners. We note that, the presented herding attack on concatenation combiners applies to XOR combiners as well.

In the sequel, we denote by $\mathcal{H}_1$ and $\mathcal{H}_2$ the underlying hash functions, h_1 and h_2 their compression functions, and h_1^* and h_2^* the arbitrary times of iterations of h_1 and h_2, respectively.

5.1 Preimage Attack on XOR Combiners in Quantum Settings

In this section, we extend the preimage attack on XOR combiners in [28] to its quantum version. Let V denote the target value. The goal is to find a message M such that $\mathcal{H}_1(M) \oplus \mathcal{H}_2(M) = V$. The framework of the attack in the quantum setting is the same as that in the classical setting, which can be described as follows, and also detailed in Algorithm 6.

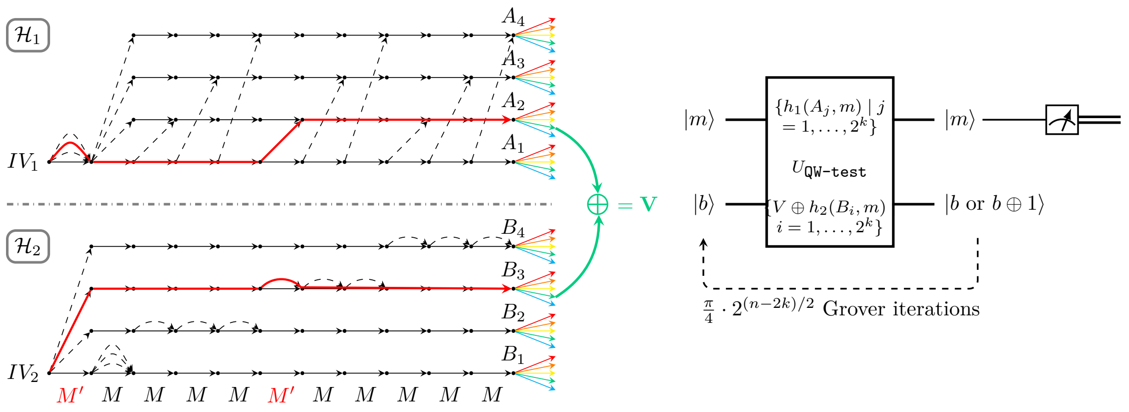
1. Build an interchange structure starting from the initialization vectors (IV_1, IV_2) and ending up with two sets of terminal states $\mathcal{A} = \{A_j \mid j = 1, \ldots, 2^k\}$ and $\mathcal{B} = \{B_i \mid i = 1, \ldots, 2^k\}$.

2. Launch a meet-in-the-middle procedure between the two sets $\mathcal{A}$ and $\mathcal{B}$, to find a message block m, and a state $A_{j^*} \in \mathcal{A}$ and a state $B_{i^*} \in \mathcal{B}$, such that $h_1(A_{j^*}, m) = V \oplus h_2(B_{i^*}, m)$.
This procedure contains two levels of iteration. The outer level of iteration is on the message block m, and the inner level of iteration is on the pairs of states $(A_i, B_j) \in \mathcal{A} \times \mathcal{B}$ under a fixed value of m. In the quantum version, the inner lever of iteration is implemented using a quantum walk algorithm, and the outer level of iteration is implemented using Grover's search algorithm.

The details of the quantum algorithm is described in Algorithm 6.

Algorithm 6: Preimage attack on XOR combiners in Quantum Settings

1. Build a 2^k-interchange structure using the quantum algorithm described in Sect. 3.4. This structure starts with IV_1 and IV_2 and ends with two ending point sets $\{A_j | j = 1 \cdots 2^k\}$ and $\{B_i | i = 1 \cdots 2^k\}$, so that for any state pair (A_j, B_i), we can easily find a message linking from starting points to it.
2. For each message block m, let $F(m)$ be the indicator function that $F(m) = 1$ if there exist a pair (A_{j^*}, B_{i^*}) in the two sets of ending points such that $h_1(A_{j^*}, m) = V \oplus h_2(B_{i^*}, m)$, and $F(m) = 0$ otherwise. To calculate $F(m)$, we use the quantum walk algorithm to find a collision between the two sets $\{A'_j = h_1(A_j, m) | j = 1 \cdots 2^k\}$ and $\{B'_i = V \oplus h_2(B_i, m) | i = 1 \cdots 2^k\}$. Denote this step by $U_{\texttt{QW-test}}$ and the ancillary qubit to indicate the value of F by $|b\rangle$.
3. Use Grover's algorithm to find a message block m^* satisfying $F(m^*) = 1$ in the space of 2^{n-2k} message blocks. Since the probability of finding a match between the above two sets is 2^{2k-n}, it requires performing about $\frac{\pi}{4} \cdot 2^{(n-2k)/2}$ Grover's steps.
4. Return $M = M^* \| m^*$ where M^* is the message mapping (IV_1, IV_2) to (A_{j^*}, B_{i^*}) corresponding to the hash values of $\mathcal{H}_1$ and $\mathcal{H}_2$.


$\mathcal{H}_1$
A_4
A_3
A_2
A_1
IV_1
$\oplus$ = V
$\mathcal{H}_2$
B_4
B_3
B_2
B_1
IV_2
M' M M M M M' M M M M M
$|m\rangle$
$\{h_1(A_j, m) \mid j = 1, \ldots, 2^k\}$
$U_{\texttt{QW-test}}$
$\{V \oplus h_2(B_i, m) \mid i = 1, \ldots, 2^k\}$
$|m\rangle$
$|b\rangle$
$|b$ or $b \oplus 1\rangle$
$\frac{\pi}{4} \cdot 2^{(n-2k)/2}$ Grover iterations

Attack in Scenario $\mathcal{R}_1$. Since the evaluation of $F(m)$ is performed during Grover's algorithm, the total computational complexity is the multiplication of the complexity of evaluating $F(m)$ and the total number of Grover's steps plus the complexity of building the interchange structure. It requires approximately $\frac{n}{2} \cdot 2^{2k+n/3}$ quantum computations to build a 2^k-interchange structure, $(2 \cdot 2^k)^{2/3}$ quantum computations to find a collision between the two sets of 2^{k+1} elements, and $\frac{\pi}{4} \cdot 2^{n-2k}$ iterations in Grover's algorithm. Then, the total workload required is

$$\frac{n}{2} \cdot 2^{2k+n/3} + 2^{2(k+1)/3} \cdot \frac{\pi}{4} \cdot 2^{(n-2k)/2} \approx \frac{n}{2} \cdot 2^{2k+n/3} + 2^{n/2-k/3}.$$

The best-case Complexity. The minimum complexity of the quantum preimage attack on XOR combiners based on the interchange structure can be achieved by selecting a message block that makes two parts of the complexity equal. When n is large enough that $\frac{n}{2}$ is negligible compared to $2^{n/3}$, we select the parameter k such that

$$2k + \frac{n}{3} = \frac{n}{2} - \frac{k}{3},$$

i.e., $k = \frac{n}{14}$. This results in a total complexity $O(2^{10n/21})$, which is slightly faster than Grover's algorithm.When n is small, we choose the value of k such that

$$\log_2 n - 1 + 2k + \frac{n}{3} = \frac{n}{2} - \frac{k}{3},$$

i.e., $k = \frac{3}{7} \cdot \left(\frac{n}{6} + 1 - \log n\right)$. For the attack to be faster than Grover's, it requires $k > 0$ and the value of n should be large enough to satisfy $\frac{n}{6} + 1 - \log_2 n > 0$, *e.g.*, $n \geq 20$.

Attack in Scenario $\mathcal{R}_2$. As analyzed in Sect. 3.4, the complexity of building a 2^k-interchange structure in this situation is $O(2^{2k+3n/7})$ time, $O(2^{n/5})$ classical memory and $O(2^{n/7})$ quantum memory. Step 3 of Algorithm 6 can be done after $O\left(2^{6(k+1)/7} \cdot \frac{\pi}{4} \cdot 2^{(n-2k)/2}\right) = O(2^{n/2-k/7})$ evaluations, since the quantum walk to search a collision in a set of 2^{k+1} elements requires $O(2^{6(k+1)/7})$ computations and $O(2^{k/7})$ quantum memory. Combined with the complexity of Step 1, the total computational complexity is $O(2^{2k+3n/7} + 2^{n/2-k/7})$.

The best-case Complexity. Choosing k such that $2k + \frac{3n}{7} = \frac{n}{2} - \frac{k}{7}$, *i.e.*, $k = \frac{n}{30}$ can minimize the time complexity of the preimage attack on XOR combiners to $2^{52n/105}$.

5.2 Collision Attack, Preimage Attack, and Second-Preimage attack on Concatenation Combiners in Quantum Settings

In this section, we present the collision attack and preimage attack on concatenation combiners in the quantum setting, which are directly converted from the classical attacks in [24]. Both quantum attacks use the quantum algorithm for building the Joux's multi-collision (refer to Sect. 3.1) and the quantum walk algorithm (refer to Sect. 2.5) for finding a collision from a set, which is different from the classical method by brute-force search.

Collision Attack. Here we introduce the quantum collision attack, which aims to find a pair of message blocks (M, M') such that $\mathcal{H}_1(M)\|\mathcal{H}_2(M) = \mathcal{H}_1(M')\|\mathcal{H}_2(M')$. The collision attack follows two steps:

Step 1: Apply Algorithm 1 to build $2^{n/2}$-Joux's multi-collision for the first compression hash function. Denote this set by $\mathcal{M}_{\mathtt{MC}}$. This step can be done in $O\left(\frac{n}{2} \cdot 2^{n/3}\right)$ time complexity.

Step 2: Apply quantum walk algorithm to find a collision of the second hash function in a set of $2^{n/2}$ message blocks constructed from $\mathcal{M}_{\mathtt{MC}}$. This step can be done in $O\left((2^{n/2})^{2/3}\right) = O(2^{n/3})$ time.

The time complexities of the two steps are balanced at $\tilde{O}(2^{n/3})$, using $O(2^{n/3})$ quantum memory. In Scenario $\mathcal{R}_2$, Step 1 can be done in $O\left(\frac{n}{2} \cdot 2^{2n/5}\right)$ time, using $O(2^{n/5})$ classical memory; Step 2 can be done in $O\left((2^{n/2})^{6/7}\right) = O(2^{3n/7})$ time, using $O(2^{n/7})$ memory. The total time and classical memory complexities under this scenario are $O(2^{3n/7})$ and $O(2^{n/5})$, respectively.

Preimage Attack. Let $V_1 \| V_2$ be a prefix of $2n$ bits. The goal of a preimage attack is to find a message M such that the concatenation of the outputs of the hash functions, $\mathcal{H}_1$ and $\mathcal{H}_2$ acting on M, is equal to V, *i.e.,* $\mathcal{H}_1(M) \| \mathcal{H}_2(M) = V_1 \| V_2$. We can directly generate a quantum attack based on Grover's algorithm to search for M in a space of 2^{2n} message blocks. With high probability, there exists one message M that satisfies the above condition; this attack require approximately $\pi/4 \cdot 2^n$ Grover steps to find M. This attack is considered as a generic quantum attack on any ideal hash construction of $2n$ output bits.

To devise a more efficient attack on concatenation combiners of MD hashes than the above most generic attack, we extend the attack in [24] to its quantum version. That is, we first build a 2^n-Joux's multi-collision for the first hash function $\mathcal{H}_1$ by Algorithm 1, and denote this set by $\mathcal{M}_{\mathtt{MC}}$. All messages in $\mathcal{M}_{\mathtt{MC}}$ have the same hash value as x. From the hash value x, we find a message block m among 2^n message blocks so that $h(x, m) = V_1$. This step can be done by Grover's algorithm in $O(2^{n/2})$ time. For the hash function $\mathcal{H}_2$, search M_1 from the set $\mathcal{M}_{\mathtt{MC}}$ such that $\mathcal{H}_2(M_1 \| m) = V_2$. Since the cardinality of $\mathcal{M}_{\mathtt{MC}}$ is 2^n, it is expected there is at least one such message M_1. This step can be done by Grover's algorithm searching in the space of messages in $\mathcal{M}_{\mathtt{MC}}$ with time complexity $O(2^{n/2})$. Therefore, the total workload required is $O(n \cdot 2^{n/3} + 2^{n/2}) = O(2^{n/2})$, using $O(2^{n/3})$ quantum memory in Scenario $\mathcal{R}_1$. In Scenario $\mathcal{R}_2$, the time complexity of a quantum attack does not change so much, which is $O(n \cdot 2^{2n/5} + 2^{n/2}) = O(2^{n/2})$; because it is dominated by the searching step, in which we can simply replace the quantum memory by a classical memory of size $O(2^{n/5})$. This attack exponentially speeds up the plain quantum attack using Grover's search, and also exponentially improves the classical attack, of which the time complexity is $O(2^n)$.

Compared to the quantum preimage attack on one MD hash function of n bits, the attack on concatenated combiners only require a constant factor of more evaluations.

Second-Preimage Attack. Since the second-preimage attack can be implied from the preimage attack, the complexity is similar to the preimage attack.

5.3 Herding Attack on Concatenation Combiners in Quantum Settings

The quantum herding attack on a single MD hash function has been introduced in Sect. 4.3. In this section, we adapt the quantum herding attack to concatenation of two MD hashes. The framework of the attack follows that of the classical attack in [4], in which the main idea is that by constructing a multi-collision $\mathcal{M}_{\mathtt{MC}}$ for $\mathcal{H}_1$ one can use the messages in $\mathcal{M}_{\mathtt{MC}}$ to builds a diamond structure for $\mathcal{H}_2$. The high level description of the attack is as follows, which is also illustrated by the figure in Algorithm 7.

1. **Phase 1 - off-line precomputation.**
   (a) Build a succeeding of three structures for $\mathcal{H}_1$: a diamond structure $\mathcal{M}_{\mathtt{DS}1}$, a Joux's multi-collision $\mathcal{M}_{\mathtt{MC}s}$, and a Joux's multi-collision $\mathcal{M}_{\mathtt{MC}\ell}$. Terminate at a state T_1.
   (b) Build one structure for $\mathcal{H}_2$: a diamond structure $\mathcal{M}_{\mathtt{DS}2}$ using the messages in $\mathcal{M}_{\mathtt{MC}\ell}$. Terminate at a state T_2.
   (c) Commit $T_1 \| T_2$ to the public.
2. **Phase 2 - on-line.**
   (a) Being challenged by a prefix P, compute the two intermediate states $x_P = h_1^*(IV_1, P)$ and $y_P = h_2^*(IV_2, P)$.
   (b) For $\mathcal{H}_1$, find a message block m^* mapping x_P to one of the leaf states x_j of $\mathcal{M}_{\mathtt{DS}1}$, retrieve the message fragment S_1 mapping x_j to the root of $\mathcal{M}_{\mathtt{DS}1}$. For $\mathcal{H}_2$, compute $y_T = h_2^*(IV_2, P \| m^* \| S_1)$

(c) For $\mathcal{H}_2$, find a message fragment S_2 in $\mathcal{M}_{\mathtt{MC}s}$ mapping y_T to one of the leaf states y_i of $\mathcal{M}_{\mathtt{DS}2}$. Retrieve the message fragment S_3 mapping y_i to the root of $\mathcal{M}_{\mathtt{DS}2}$, which is T_2.

(d) Response with $M = P\|m^*\|S_1\|S_2\|S_3$.

Attack in Scenario $\mathcal{R}_1$. The details of the quantum herding attack is presented in Algorithm 7.

Algorithm 7: Herding Attack on Concatenation Combiners in Quantum Settings

Phase 1 - off-line precomputation.

(a) Build a diamond $\mathcal{M}_{\mathtt{DS}1}$ for $\mathcal{H}_1$, which starts from 2^k states $D_1 = \{x_i\}_1^{2^k}$ and are all mapped to the root value x_T. That can be done using the quantum algorithm in Sect. 3.3. From the hash value x_T, build a 2^{n-k}-Joux's multi-collision $\mathcal{M}_{\mathtt{MC}s}$, in which all messages map x_T to a state x_{M_0}. Continue to build a $2^{nk/2}$-Joux's multi-collision (consists of k fragments and each fragment is of length $n/2$) on $\mathcal{H}_1$ from the starting state x_{M_0} and mapping to the state T_1, and denote it by $\mathcal{M}_{\mathtt{MC}\ell}$. Denote the terminal states of each of the k fragments of $\mathcal{M}_{\mathtt{MC}\ell}$ by x_{M_i} for i from 1 to k (note that $x_{M_k} = T_1$).

(b) Build a diamond $\mathcal{M}_{\mathtt{DS}2}$ for $\mathcal{H}_2$, which starts from 2^k states $D_2 = \{y_i\}_1^{2^k}$. The messages used to building $\mathcal{M}_{\mathtt{DS}2}$ are all chosen from the set $\mathcal{M}_{\mathtt{MC}\ell}$. For example, the messages mapping the first layer of 2^k states to the 2^{k-1} states in $\mathcal{M}_{\mathtt{DS}2}$ are chosen from the set of $2^{n/2}$ messages in the first fragment of $\mathcal{M}_{\mathtt{MC}\ell}$ mapping x_{M_0} to x_{M_1}. To build the next layer from D_2, use the quantum walk algorithm to find a collision in the set of $2^{n/2}$ messages for pairs of states in D_2, with $O(2^{n/3})$ quantum computations. Repeats this step until reaching a root T_2 for $\mathcal{M}_{\mathtt{DS}2}$. Note that, the building method for $\mathcal{M}_{\mathtt{DS}2}$ is different from the quantum algorithm describe in Sect. 3.3. That is because, the messages should be selected from the set $\mathcal{M}_{\mathtt{MC}\ell}$, which is limited. Therefore, building the diamond structure $\mathcal{M}_{\mathtt{DS}2}$ costs $O(2^k \cdot 2^{n/3}) = O(2^{(n+3k)/3})$ computations.

(c) Commit $T_1\|T_2$ to the public.

Phase 2 - on-line. Being challenged with a prefix P, proceed as follows.

(a) Compute the two intermediate states $x_P = h_1^*(IV_1, P)$ and $y_P = h_2^*(IV_2, P)$.

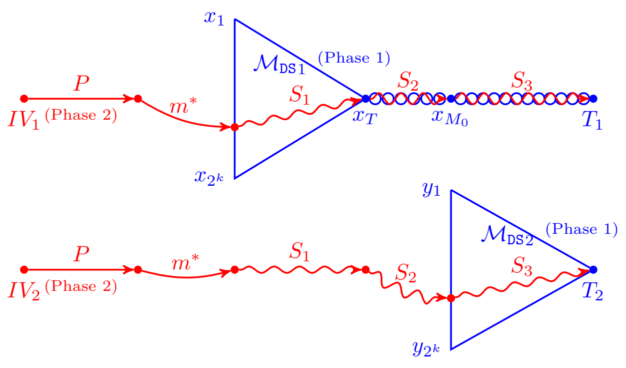
(b) Search for a message block m^* that maps x_P to one of the leaf states x_j of $\mathcal{M}_{\mathtt{DS}1}$. This is done by using Grover's algorithm, which accesses the quantum oracle of h_1 to find m^* in $O(2^{(n-k)/2})$ steps.

(c) Retrieve the message S_1 in $\mathcal{M}_{\mathtt{DS}1}$ that maps x_j to the root. Compute $y_T = h_2^*(IV_2, P\|m^*\|S_1)$.

(d) Search for a message fragment S_2 among $\mathcal{M}_{\mathtt{MC}s}$ that maps y_T to one of the leaf states y_i of $\mathcal{M}_{\mathtt{DS}2}$. This is done by using Grover's algorithm again.

(e) Retrieve the message fragment S_3 in $\mathcal{M}_{\mathtt{DS}2}$ that maps y_i to the root, which is T_2. Due to the way of construction of $\mathcal{M}_{\mathtt{DS}2}$ in Phase 1, for $\mathcal{H}_1$, the message fragment S_3 also maps the starting state of $\mathcal{M}_{\mathtt{MC}\ell}$ to T_1.

(f) Response with $M = P\|m^*\|S_1\|S_2\|S_3$.



The time complexity of the precomputation phase includes that of building $n-k+nk/2$ Joux's-multicollision and that of building the two diamond structures $\mathcal{M}_{\mathtt{DS}1}$ and $\mathcal{M}_{\mathtt{DS}2}$. Therefore, the total time complexity is $O\left((n-k+(nk)/2)\cdot 2^{n/3} + 2^{(n+3k)/3}\right) = O(2^{(n+3k)/3})$. The online phase is done in time complexity $O(2^{(n-k)/2})$. Therefore, the total work is done in $O(2^{(n+3k)/3} + 2^{(n-k)/2})$ quantum time, with $O(2^{n/3})$ quantum memory since the quantum memory can be re-used after each iteration.

THE BEST-CASE COMPLEXITY. To minimize the time complexity of the herding attack on concatenated hashes, we choose the length of the message which balances the time complexity of two phases, *i.e.*, $(n+3k)/3 = (n-k)/2$. Then with the value $k = n/9$, the attack is optimized with $O(2^{4n/9})$ time, while a naive quantum herding attack using Grover's algorithm requires about $O(2^n)$ quantum computations to search for the suffix S.

Attack in Scenario $\mathcal{R}_2$. The difference in this model lies in the complexity of building $n-k+(nk)/2$ Joux's multi-collisions, building the diamonds, and finding message fragments linking from collision values to one of starting points of diamonds. To build $\mathcal{M}_{\mathtt{DS2}}$, we need 2^k iterations to merge pairs of hash values into one. This step costs $O(2^k \cdot (2^{n/2})^{6/7}) = O(2^{3n/7+k})$ computations and $O(2^{n/7})$ quantum memory size. Then, the precomputation phase is of $O\left((n-k+(nk)/2)\cdot 2^{2n/5} + 2^{3n/7+k}\right) = O(2^{3n/7+k})$ time complexity, using $O(2^{n/5})$ classical memory, and $O(2^{n/7})$ quantum memory size.

THE BEST-CASE COMPLEXITY. The best attack is achieved when k satisfies $3n/7 + k = n/2 - k/6$, *i.e.*, $k = 3n/49$. It gives the time complexity of $O(2^{24n/49})$.

6 Conclusions

In this paper, we studied the security of various constructions of hash functions in quantum settings with respect to important attacks: collision attacks, (second-) preimage attacks, and herding attacks. We analyzed the complexities of these attacks under different quantum memory models.The results show that our attacks in both models have better time complexity than that of the generic attacks by directly applying Grover's algorithm, and exponentially reduce both time and memory complexities compared to the classical attacks. The cryptanalysis results of hash combiners in quantum settings is consistent with that in the classical setting, that is, the security of most hash combiners are not as high as commonly expected, and can be even lower than that of a single underlying hash function. These results serve as an indication that, to achieve long-term security to the post-quantum era, current symmetric-key crypto-systems require careful security re-evaluation or even re-design before being adopted by post-quantum cryptography schemes.

Acknowlegements. This research is partially supported by Nanyang Technological University in Singapore under Start-up Grant 04INS000397C230, and Ministry of Education in Singapore under Grants RG91/20 and MOE2019-T2-1-060. This version of the contribution has been accepted for publication at NSS 2022, after peer review (when applicable) but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections.The Version of Record is available online at: `https://doi.org/10.1007/978-3-031-23020-2_39`. Use of this Accepted Version is subject to the publishers Accepted Manuscript terms of use.

References

1. Aaronson, S., Shi, Y.: Quantum lower bounds for the collision and the element distinctness problems. Journal of the ACM (JACM) 51(4), 595–605 (2004)
2. Ambainis, A.: Quantum walk algorithm for element distinctness. SIAM Journal on Computing 37(1), 210–239 (2007)
3. Andreeva, E., Bouillaguet, C., Dunkelman, O., Fouque, P.A., Hoch, J.J., Kelsey, J., Shamir, A., Zimmer, S.: New Second-Preimage Attacks on Hash Functions. Journal of Cryptology 29(4), 657–696 (Oct 2016)
4. Andreeva, E., Bouillaguet, C., Dunkelman, O., Kelsey, J.: Herding, Second Preimage and Trojan Message Attacks beyond Merkle-Damgård. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009: 16th Annual International Workshop on Selected Areas in Cryptography. LNCS, vol. 5867, pp. 393–414. Springer, Heidelberg, Germany, Calgary, Alberta, Canada (Aug 13–14, 2009)

5. Andreeva, E., Bouillaguet, C., Fouque, P.A., Hoch, J.J., Kelsey, J., Shamir, A., Zimmer, S.: Second Preimage Attacks on Dithered Hash Functions. In: Smart, N.P. (ed.) Advances in Cryptology – EUROCRYPT 2008. LNCS, vol. 4965, pp. 270–288. Springer, Heidelberg, Germany, Istanbul, Turkey (Apr 13–17, 2008)
6. Bao, Z., Dinur, I., Guo, J., Leurent, G., Wang, L.: Generic attacks on hash combiners. Journal of Cryptology pp. 1–82 (2019)
7. Bao, Z., Wang, L., Guo, J., Gu, D.: Functional Graph Revisited: Updates on (Second) Preimage Attacks on Hash Combiners. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology – CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 404–427. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 20–24, 2017)
8. Blackburn, S.R., Stinson, D.R., Upadhyay, J.: On the Complexity of the Herding Attack and Some Related Attacks on Hash Functions. Cryptology ePrint Archive, Report 2010/030 (2010), http://eprint.iacr.org/2010/030
9. Brassard, G., Høyer, P., Tapp, A.: Quantum cryptanalysis of hash and claw-free functions. In: Latin American Symposium on Theoretical Informatics. pp. 163–169. Springer (1998)
10. Canteaut, A., Duval, S., Leurent, G., Naya-Plasencia, M., Perrin, L., Pornin, T., Schrottenloher, A.: Saturnin: a suite of lightweight symmetric algorithms for post-quantum security. IACR Trans. Symmetric Cryptol. 2020(S1), 160–207 (2020), https://doi.org/10.13154/tosc.v2020.iS1.160-207
11. Chailloux, A., Naya-Plasencia, M., Schrottenloher, A.: An Efficient Quantum Collision Search Algorithm and Implications on Symmetric Cryptography. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology – ASIACRYPT 2017, Part II. LNCS, vol. 10625, pp. 211–240. Springer, Heidelberg, Germany, Hong Kong, China (Dec 3–7, 2017)
12. Damgård, I.: A Design Principle for Hash Functions. In: Brassard, G. (ed.) Advances in Cryptology – CRYPTO'89. LNCS, vol. 435, pp. 416–427. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 20–24, 1990)
13. Dierks, T., Allen, C.: The TLS protocol version 1.0. RFC 2246, 1–80 (1999), https://doi.org/10.17487/RFC2246
14. Dierks, T., Rescorla, E.: The transport layer security (TLS) protocol version 1.1. RFC 4346, 1–87 (2006), https://doi.org/10.17487/RFC4346
15. Dinur, I.: New Attacks on the Concatenation and XOR Hash Combiners. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part I. LNCS, vol. 9665, pp. 484–508. Springer, Heidelberg, Germany, Vienna, Austria (May 8–12, 2016)
16. Fischlin, M., Lehmann, A., Wagner, D.: Hash Function Combiners in TLS and SSL. In: Pieprzyk, J. (ed.) Topics in Cryptology – CT-RSA 2010. LNCS, vol. 5985, pp. 268–283. Springer, Heidelberg, Germany, San Francisco, CA, USA (Mar 1–5, 2010)
17. Freier, A.O., Karlton, P., Kocher, P.C.: The secure sockets layer (SSL) protocol version 3.0. RFC 6101, 1–67 (2011), https://doi.org/10.17487/RFC6101
18. Google: Google Quantum Computing, https://research.google/teams/applied-science/quantum/
19. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996)
20. Hosoyamada, A., Yasuda, K.: Building Quantum-One-Way Functions from Block Ciphers: Davies-Meyer and Merkle-Damgård Constructions. In: Peyrin, T., Galbraith, S. (eds.) Advances in Cryptology – ASIACRYPT 2018, Part I. LNCS, vol. 11272, pp. 275–304. Springer, Heidelberg, Germany, Brisbane, Queensland, Australia (Dec 2–6, 2018)
21. IBM: IBM Quantum Computing, https://www.ibm.com/quantum-computing/
22. Jaques, S., Schrottenloher, A.: Low-gate quantum golden collision finding. Cryptology ePrint Archive, Report 2020/424 (2020), https://eprint.iacr.org/2020/424
23. Jha, A., Nandi, M.: Some Cryptanalytic Results on Zipper Hash and Concatenated Hash. Cryptology ePrint Archive, Report 2015/973 (2015), http://eprint.iacr.org/2015/973
24. Joux, A.: Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions. In: Franklin, M. (ed.) Advances in Cryptology – CRYPTO 2004. LNCS, vol. 3152, pp. 306–316. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 15–19, 2004)
25. Kelsey, J., Kohno, T.: Herding Hash Functions and the Nostradamus Attack. In: Vaudenay, S. (ed.) Advances in Cryptology – EUROCRYPT 2006. LNCS, vol. 4004, pp. 183–200. Springer, Heidelberg, Germany, St. Petersburg, Russia (May 28 – Jun 1, 2006)
26. Kelsey, J., Schneier, B.: Second Preimages on n-Bit Hash Functions for Much Less than 2n Work. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. LNCS, vol. 3494, pp. 474–490. Springer, Heidelberg, Germany, Aarhus, Denmark (May 22–26, 2005)
27. Kortelainen, T., Kortelainen, J.: On Diamond Structures and Trojan Message Attacks. In: Sako, K., Sarkar, P. (eds.) Advances in Cryptology – ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 524–539. Springer, Heidelberg, Germany, Bengalore, India (Dec 1–5, 2013)
28. Leurent, G., Wang, L.: The Sum Can Be Weaker Than Each Part. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 345–367. Springer, Heidelberg, Germany, Sofia, Bulgaria (Apr 26–30, 2015)

29. Merkle, R.C.: One Way Hash Functions and DES. In: Brassard, G. (ed.) Advances in Cryptology – CRYPTO'89. LNCS, vol. 435, pp. 428–446. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 20–24, 1990)
30. National Institute for Standards and Technology, USA: Post-Quantum Cryptography Standardization (2017), https://csrc.nist.gov/projects/post-quantum-cryptography
31. 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
32. Zalka, C.: Grovers quantum searching algorithm is optimal. Physical Review A 60(4), 2746 (1999)
33. Zhandry, M.: A note on the quantum collision and set equality problems. arXiv preprint arXiv:1312.1027 (2013)
34. Zhandry, M.: How to Record Quantum Queries, and Applications to Quantum Indifferentiability. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part II. LNCS, vol. 11693, pp. 239–268. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 18–22, 2019)