Tight Multi-User Security Bound of DbHtS

Nilanjan Datta\textsuperscript{1} and Avijit Dutta\textsuperscript{1} and Mridul Nandi\textsuperscript{1,2} and Suprita Talnikar\textsuperscript{2}

Institute for Advancing Intelligence, TCG CREST, Kolkata, India
Indian Statistical Institute, Kolkata, India.
nilanjan.datta@tcgcrest.org, avirocks.dutta13@gmail.com, mridul.nandi@gmail.com, suprita45@gmail.com

Abstract. In CRYPTO’21, Shen et al. proved that Two-Keyed-DbHtS construction is secure up to $2^{2n/3}$ queries in the multi-user setting independent of the number of users. Here the underlying double-block hash function $H$ of the construction is realized as the concatenation of two independent $n$-bit keyed hash functions $(H_{K_1,h}, H_{K_2,h})$, and the security holds under the assumption that each of the $n$-bit keyed hash function is universal and regular. The authors have also demonstrated the applicability of their result to the key-reduced variants of DbHtS MACs, including 2K-SUM-ECBC, 2K-PMAC\_Plus and 2K-LightMAC\_Plus without requiring domain separation technique and proved $2n/3$-bit multi-user security of these constructions in the ideal cipher model. Recently, Guo and Wang have invalidated the security claim of Shen et al.’s result by exhibiting three constructions, which are instantiations of the Two-Keyed-DbHtS framework, such that each of their $n$-bit keyed hash functions are $O(2^{-n})$ universal and regular, while the constructions themselves are secure only up to the birthday bound. In this work, we show a sufficient condition on the underlying Double-block Hash (DbH) function, under which we prove an improved $3n/4$-bit multi-user security of the Two-Keyed-DbHtS construction in the ideal-cipher model. To be more precise, we show that if each of the $n$-bit keyed hash function is universal, regular, and cross-collision resistant then it achieves the desired security. As an instantiation, we show that two-keyed Polyhash-based DbHtS construction is multi-user secure up to $2^{3n/4}$ queries in the ideal-cipher model. Furthermore, due to the generic attack on DbHtS constructions by Leurent et al. in CRYPTO’18, our derived bound for the construction is tight.

Keywords: DbHtS · PRF · Polyhash · Tight Multi-user Security · H-Coefficient Technique · Mirror Theory.

1 Introduction

Hash-then-PRF [34] (or HtP) is a well-known paradigm for designing variable input-length PRFs, in which an input message of arbitrary length is hashed and the hash value is encrypted through a PRF to obtain a short tag. Most popular MACs including the CBC-MAC [3], PMAC [10], OMAC [20] and LightMAC [24] are designed using the HtP paradigm. Although the method is simple, in particular being deterministic and stateless, the security of MACs following the HtP paradigm is capped at the birthday bound due to the collision probability of the hash function. Birthday bound-secure constructions are acceptable in practice when any of these MACs are instantiated with a block cipher of moderately large block size. For example, instantiating PMAC with AES-128 permits roughly $2^{48}$ queries (using $56q^2/2^n$ [31] bound) when the longest message size is $2^{16}$ blocks, and the success probability of breaking the scheme is restricted to $2^{-10}$. However, the same construction becomes vulnerable if instantiated with some lightweight (smaller block size) block ciphers,
whose number has grown tremendously in recent years, e.g. *PRESENT* [11], *GIFT* [1], *LED* [16], etc. For example, *PMAC*, when instantiated with the *PRESENT* block cipher (a 64-bit block cipher), permits only about $2^{16}$ queries when the longest message size is $2^{16}$ blocks, and the probability of breaking the scheme is $2^{-10}$. Therefore, it becomes risky to use birthday bound-secure constructions instantiated with lightweight block ciphers. In fact, in a large number of financial sectors, web browsers still widely use 64-bit block ciphers *3-DES* instead of *AES* in their legacy applications with backward compatibility feature, as using the latter in corporate mainframe computers is more expensive. However, it does not give adequate security if the mode in which *3-DES* is used provides only birthday bound security, and hence a beyond birthday secure mode solves the issue. Although many secure practical applications use the standard *AES-128*, which provides 64-bit security in a birthday bound-secure mode, which is adequate for the current technology, it may not remain so in the near future. In such a situation, using a mode with beyond the birthday bound security instead of replacing the cipher with a larger block size is a better option.  

**Double-Block Hash-then-Sum.** Many studies tried to tweak the HtP design paradigm to obtain beyond the birthday bound secure MACs; while they possess a similar structural design, the internal state of the hash function is doubled and the two $n$-bit hash values are first encrypted and then xored together to produce the output. In [36], Yasuda proposed a beyond the birthday bound secure deterministic MAC called *SUM-ECBC*, a rate-1/2 sequential mode of construction with four block cipher keys. Followed by this work, Yasuda [37] came up with another deterministic MAC called *PMAC_Plus*, but unlike *SUM-ECBC*, *PMAC_Plus* is a rate-1 parallel mode of construction with three block cipher keys. Zhang et al. [38] proposed another rate-1 beyond the birthday bound secure deterministic MAC called *3kf9* with three block cipher keys. In [30], Naito proposed *LightMAC_Plus*, a rate $(1 - s/n)$ parallel mode of operation, where $s$ is the size of the block counter. The structural design of all these constructions first applies a $2n$-bit hash function on the message, then the two $n$-bit output values are encrypted and xored together to produce the tag, where $n$ is the block size of the block cipher. Moreover, all of them also give $2n/3$-bit security. In FSE 2019, Datta et al. [14] proposed a generic design paradigm dubbed as the *double-block hash-then-sum* or *DbHtS*, defined as follows:

$$\text{DbHtS}(M) \triangleq E_{K_1}(\Sigma) \oplus E_{K_2}(\Theta), \ (\Sigma, \Theta) \leftarrow H_{K_0}(M),$$

where $H_{K_0}$ is a double-block hash function that maps an arbitrary-length string to a $2n$-bit string. Within this unified framework, they revisited the security proof of existing *DbHtS* constructions, including *PolyMAC* [21], *SUM-ECBC* [36], *PMAC_Plus* [37], *3kf9* [38] and *LightMAC_Plus* [30] and also their two-keyed versions [14] and confirmed that all the constructions are secure up to $2^{2n/3}$ queries when they are instantiated with an $n$-bit block cipher.

In CRYPTO 2018, Leurent et al. [22] proposed a generic attack on all these constructions using $2^{3n/4}$ (short message) queries, leaving a gap between the upper and the lower bounds for the provable security of *DbHtS* constructions. Recently, Kim et al. [21] have improved the bound of *DbHtS* constructions from $2^{2n/3}$ to $2^{3n/4}$. They have shown that if the underlying $2n$-bit hash function is the concatenation of two independent $n$-bit-universal hash functions 2, then the resulting *DbHtS* paradigm is secure up to $2^{3n/4}$ queries. They have also improved the security bound of *PMAC_Plus*, *3kf9* and *LightMAC_Plus* from $2^{2n/3}$ to $2^{3n/4}$ and hence closed the gap between the upper and the lower bounds of the provable security of *DbHtS* constructions.

**Multi-user security of *DbHtS***. We have so far discussed the security bounds of *DbHtS* constructions in which adversaries are given access to some keyed oracles for

---

1Note that there are no standard block ciphers of size higher than 128 bits.

2A family of keyed hash functions is said to be universal if for any distinct $x$ and $x'$, the probability of a collision in their hash values for a randomly sampled hash function from the family is negligible.
a single unknown randomly sampled key. Such a model is known as the single-user security model, i.e. when the adversary interacts with one specific machine in which the cryptographic algorithm is deployed and tries to compromise its security. However, in practice, cryptographic algorithms are usually deployed in more than one machine. For example, AES-GCM [25, 26] is now widely used in the TLS protocol to protect web traffic and is currently used by billions of users daily. Thus, the security of DbHtS constructions in the multi-key setting is worth investigating; in other words, we ask to what extent the number of users will affect the security of DbHtS constructions, where adversaries are successful if they compromise the security of one out of many users. That means the adversary’s winning condition is a disjunction of single-key winning conditions.

The notion of multi-user (mu) security was introduced by Biham [8] in symmetric cryptanalysis and by Bellare, Boldyreva, and Micalli [2] in the context of public-key encryption. In the multi-user setting, attackers have access to multiple machines such that a particular cryptographic algorithm F is deployed in each machine with independent secret keys. An attacker can adaptively distribute its queries across multiple machines with independent keys. Multi-user security considers attackers that succeed in compromising the security of at least one machine, among others.

Multi-user security for block ciphers is different from multi-user security for modes. In the single-key setting, the best attacks against block cipher such as AES do not improve with increased data complexity. However, in the multi-key environment, they do, as first observed by Biham [8] and later refined as a time-memory-data trade-off by Biryukov et al. [9]. These results demonstrate how one can take advantage of the fact that recovering a block cipher key out of a large group of keys is much easier than targeting a specific key. The same observation can be applied to any deterministic symmetric-key algorithm, as done for MACs by Chatterjee et al. [13]. A more general result guarantees that the multi-user advantage of an adversary for a cryptographic algorithm is at most $u$ times its single user advantage. Therefore, for any cryptographic algorithm, a multi-user security bound involving a factor $u$ is easily established using a hybrid argument that shows the upper bound of the adversarial success probability to be roughly $u$ times its single-user security advantage. Bellare and Tackmann [5] first formalized a multi-user secure authenticated encryption scheme and also analyzed countermeasures against multi-key attacks in the context of TLS 1.3. However, they derived a security bound that also contained the factor $u$. Such a bound implies a significant security drop of the construction when the number of users is large, and in fact, this is precisely the situation faced in large-scale deployments of AES-GCM such as TLS.

As evident from [4, 5, 12, 18, 19, 23, 29], it is a challenging problem to study the security degradation of cryptographic primitives with the number of users, even when its security is known in the single-user setting. Studies of multi-user security of MACs are somewhat scarce in the literature except for the work of Chatterjee et al. [13], and a very recent work of Morgan et al. [28], and Bellare et al. [6]. The first two consider a generic reduction for MACs, in which the security of the primitive in the multi-user setting is derived by multiplying the number of users $u$ by the single-user security.

In CRYPTO’21, Shen et al. [33] have analyzed the security of DbHtS in the multi-user setting. It is worth noting here that by applying the generic reduction from the single-user to the multi-user setting, the security bound of DbHtS would have capped at worse than the birthday bound, i.e. $uq^{4/3}/2^n$, when each user made a single query and the number of users reached $q$. Thus, a direct analysis was needed for deriving the multi-user bound of the construction. Shen et al. [33] have shown that in the multi-user setting, the two-keyed DbHtS paradigm,

$$\text{Two-Keyed-DbHtS}(M) \triangleq E_K(H_{K_h,1}(M)) \oplus E_K(H_{K_h,2}(M)),$$

where $u$ two-keyed stands for one hash key and one block cipher key.
is secure up to $2^{2n/3}$ queries in the ideal-cipher model when the 2n-bit double-block hash function is the concatenation of two independent n-bit keyed hash functions $H_{K_1}$ and $H_{K_2}$. In particular, they have shown that if both $H_{K_1}$ and $H_{K_2}$ are $O(2^{-n})$-regular and $O(2^{-n})$-universal \(^4\), then the multi-user security bound of the two-keyed DbHtS is of the order of

$$\frac{qp\ell}{2^{k+n}} + \frac{q^3}{2^{2n}} + \frac{q^2 p + q p^2}{2^{2k}},$$

where $q$ is the total number of MAC queries across all $u$ users, $p$ is the total number of ideal-cipher queries, $\ell$ is the maximum number of message blocks among all queries and $n, k$ are the block size and the key size of the block cipher respectively. Note that the above bound is independent of the number of users $u$, which can be adaptively chosen by the adversary and grows as large as $q$. Besides this result, Shen et al. have also shown that 2K-SUM-ECBC [14], 2K-PMAC_Plus [14] and 2K-LightMAC_Plus [14] are all secure roughly up to $2^{2n/3}$ queries (including all MAC and ideal-cipher queries) in the multi-user setting independent of the number of users, where these constructions do not employ domain separation techniques.

Remark 1. In their paper [14], Datta et al. named the two-keyed variants of SUM-ECBC, PMAC_Plus and LightMAC_Plus as 2K-SUM-ECBC, 2K-PMAC_Plus and 2K-LightMAC_Plus respectively, where for each of these constructions, the domain separation technique ensured disjointness of the set of values of $\Sigma$ and $\Theta$. However, in [33], Shen et al. considered the same constructions but without any domain separation, and refer to them using the same names. Henceforth, we shall implicitly mean the non domain-separated variants only (unless otherwise stated) when referring to the two-keyed constructions 2K-SUM-ECBC, 2K-PMAC_Plus and 2K-LightMAC_Plus.

1.1 Issue with the CRYPTO’21 Paper [33]

In this section, we discuss three issues with [33]. The first two issues examine flaws in the security analysis of the construction and the last issue points out a flawed security claim of the construction. We begin by identifying the first issue.

1. The Two-Keyed-DbHtS framework was proven to be multi-user secure up to $2^{2n/3}$ queries in the ideal-cipher model [33] under the assumption that each of the underlying n-bit independent keyed hash functions is $O(2^{-n})$-universal and regular. As an instantiation of the framework, authors have proven 2n/3-bit multi-user security of 2K-SUM-ECBC, 2K-LightMAC_Plus and 2K-PMAC_Plus in the ideal-cipher model, where the underlying DbH function of each of the above three constructions is based on block ciphers. In the security proof of these instantiated constructions, authors have bounded the regular and the universal advantages of their corresponding DbH functions (i.e., the DbH of 2K-SUM-ECBC, 2K-LightMAC_Plus and 2K-PMAC_Plus) up to $O(\ell/2^n)$, where $\ell$ is the maximum number of message blocks among all queries.

Now, one of the natural assumptions in the PRF-security proof of block cipher based DbHtS constructions in the ideal-cipher model is that the adversary should be allowed to query to the underlying block cipher used in the DbH function of the DbHtS construction. However, in [33], authors have proved the security of 2K-SUM-ECBC, 2K-LightMAC_Plus and 2K-PMAC_Plus constructions without considering this assumption. In particular, they derived bounds of the regular and universal advantages of the underlying double block hash functions of 2K-SUM-ECBC, 2K-LightMAC_Plus and 2K-PMAC_Plus in the setting where the adversary did not make any primitive query to the underlying block ciphers of

---

\(^4\) A family of keyed hash function is said to be $\epsilon_1$-regular if for any $x$ and $y$, the probability that a randomly sampled hash function from the family maps $x$ to $y$ is $\epsilon_1$; it is said to be $\epsilon_2$-universal if for any distinct $x, x'$, the probability that a randomly sampled hash function from the family yields a collision on the pair $(x, x')$ is $\epsilon_2$. 
the corresponding hash function. This is different from the fact that one shows a bound on
the regular and universal advantage of a double block hash function with the assumption
that an adversary is allowed to make primitive queries to the underlying block cipher
of the double block hash function. This is because, the definition of the conventional
universal (resp. regular) advantage of a keyed hash function is that no computationally
bounded adversary, without knowing the hash key, can output a pair of messages (resp. a
message $M$ and an arbitrary value $Y$ from the range of the hash function) such that their
hash value collides (resp. such that the hash function maps $M$ to the designated value $Y$)
except with small probability. On the other hand, the definition of the universal (resp.
regular) advantage of a block cipher based keyed hash function in the ideal-cipher model is
the following

**Regular Advantage:** $\Pr[(M, Y) \leftarrow A^{E,E^{-1}}, K_h \leftarrow K_h : H^E_{K_h}(M) = Y] \leq \epsilon$

**Universal Advantage:** $\Pr[(M, M') \leftarrow A^{E,E^{-1}}, K_h \leftarrow K_h : H^E_{K_h}(M) = H^E_{K_h}(M')] \leq \epsilon$,

where $E$ is the underlying block cipher of the block cipher based keyed hash function
$H^E$. The above definition of regular advantage (resp. universal advantage) says that after
the adversary interacts with the block cipher $E$, $E^{-1}$ with some chosen keys, commits
to $(M, Y)$ (resp. commits to a pair of message $(M, M')$) such that the probability that the
hash function maps $M$ to $Y$ is small (resp. probability that the hash value for $M, M'$
collides is small). To illustrate the flaw in the analysis of [33], considering the
example of 2K-LightMAC$_{Plus}$, while bounding the probability of the event $\Sigma_i = \Sigma_j$ (where $\Sigma_i = \Sigma_j \Rightarrow Y_1^i \oplus Y_2^j \oplus \ldots \oplus Y_k^j = Y_1^i \oplus Y_2^i \oplus \ldots \oplus Y_k^j$ and $Y_a^i = E_K(M_a^i || (a)_i)$), the
authors have simply assumed that at least one of variables $Y$ in the above equation will
be fresh, thus providing sufficient entropy for bounding the event. However, the authors
have missed the fact that existence of such a variable $Y$ may not always be guaranteed
in the ideal-cipher model. For example, suppose an adversary makes the following three
forward primitive queries with a chosen ideal-cipher key $J$:

1. forward query with $(x||\langle 1 \rangle)_a$ and obtains $y_1$
2. forward query with $(x'||\langle 1 \rangle)_a$ and obtains $y_2$
3. forward query with $(x''||\langle 2 \rangle)_a$ and obtains $y_3$

Let us assume that the (albeit probabilistic) event $y_1 \oplus y_2 \oplus y_3 = 0$ occurs. Suppose the
adversary makes two more construction queries: the first construction query with $(x)$ and
the second, a construction query with $(x'||x'')$. Then, if the block cipher key $K$ used in the
construction collides with the chosen ideal-cipher key $J$, then one cannot find any fresh
variable $Y$ in the following equations:

$$Y_1^1 = Y_1^2 \oplus Y_2^2.$$

Therefore, to prove the security of such block cipher-based DbHtS constructions in the
ideal-cipher model, one needs to consider the fact that the regular or universal advantage
of the underlying double block hash functions must be bounded under the assumption that
the adversary makes primitive queries to the underlying block cipher. We therefore believe
that to prove the security of the constructions in the ideal-cipher model for the block
cipher-based DbH function, one needs to provide a generalized definition of the universal
and regular advantages in the ideal-cipher model and prove their security under this model,
which was missing in [33].

2. The second issue is regarding the good transcript analysis of the Two-Keyed-DbHtS
construction. In Fig. 4 of [33], the authors have first identified the following set:

$$F(J) := \{(i, a) \in [u] \times [q] \text{ such that } \Sigma^i_a, \Theta^i_a \text{ are fresh}\},$$

where
where $\Sigma_a^i, \Theta_a^i$ fresh means that $\Sigma_a^i \notin \{\Sigma_b^i, \Theta_b^i, b \neq a \in [q_i]\}$ and $\Theta_a^i \notin \{\Sigma_b^i, \Theta_b^i, b \neq a \in [q_i]\}$. Moreover, $\Sigma_a^i, \Theta_a^i$ do not collide with the input of any forward ideal cipher queries such that the chosen ideal cipher key of that forward query collides with the $i$-th user key. They have also defined a set $S(J)$,

$$S(J) := \{(W_a^i, X_a^i) \in (\{0,1\}^n \setminus \text{Ran}(\Phi_j))^{(2^{|F(J)|})} : W_a^i \oplus X_a^i = T_a^i\},$$

where $|\text{Ran}(\Phi_j)| \geq 1$. Then for all $(i, a) \in F(J)$, $(W_a^i, X_a^i)$ is sampled from $S(J)$ and is set as the permutation output of $\Sigma_a^i$ and $\Theta_a^i$, respectively, i.e., $P(\Sigma_a^i) \leftarrow W_a^i, P(\Theta_a^i) \leftarrow X_a^i$. Note that such an assignment is sound and satisfies $P(\Sigma_a^i) \oplus P(\Theta_a^i) = T_a^i$ for all $(i, a) \in F(J)$. Finally, they have provided a lower bound on the cardinality of the set $S(J)$ using Lemma 2, where Lemma 2 provides the following lower bound

$$\Delta := \frac{2^n(2^n-1)\ldots(2^n-2q+1)}{2^{2q}} \cdot \left(1 - \frac{6q^4}{2^{4q}}\right)$$

on the cardinality of the set

$$S := \{(U_i, V_i) \in (\{0,1\}^n)^{2q} : U_i \oplus V_i = T_i\}.$$ 

Finally, authors have used $\Delta$ as a lower bound on $|S(J)|$, reveals a fallacy as the two sets $S$ and $S(J)$ are not of same size.

3. The third issue is regarding the flawed security claim of the Two-Keyed-DbHtS construction in [33]. In Theorem 1 of [33], Shen et al. have shown that when the underlying double block hash function of the Two-Keyed-DbHtS construction is the concatenation of two independent $n$-bit keyed hash functions such that each of them is $O(2^{-n})$-universal and $O(2^{-n})$-regular, Two-Keyed-DbHtS achieves $2^{n/3}$-bit multi-user security in the ideal-cipher model. This claim has been falsified in a recent work by Guo and Wang [17], where the authors came up with three concrete double-block hash functions, each of the which is the concatenation of two independent $n$-bit keyed hash functions and each of the $n$-bit keyed hash functions meets $O(2^{-n})$-universal and $O(2^{-n})$-regular advantages. However, plugging-in these hash functions into the Two-Keyed-DbHtS framework yields a birthday bound distinguishing attack. As a consequence, the security bound of the Two-Keyed-DbHtS construction, as proven in Theorem 1 of [33], stands flawed. We would like to mention here that the attack holds only for those instances of Two-Keyed-DbHtS construction where the underlying DbH function is the concatenation of two independent $n$-bit hash functions and it does not have any domain separation. In fact, authors of [17] were not able to show any birthday bound attack on 2K-PMAC_Plus and 2K-LightMAC_Plus as the underlying DbH function of these two constructions are not merely the concatenation of two independent $n$-bit keyed hash functions. However, it is to be noted that as the double block hash function for 2K-SUM-ECBC is the concatenation of two independent $n$-bit CBC functions, the attack of [17] holds for it.

1.2 Our Contribution

In this paper we prove that the Two-Keyed-DbHtS construction is multi-user secure up to $2^{3n/4}$ queries in the ideal-cipher model. To prove it, we first define the notion of a good double-block hash function, which informally means that the concatenation of two independent $n$-bit keyed hash functions is “good” if each has negligible universal and regular advantages, and there is no cross-collision, i.e., the probability that the outputs of two hash function colliding for any pair of messages $M, M'$ is zero. We prove that if the underlying $2n$-bit DbH function of the Two-Keyed-DbHtS construction is good, such

\textsuperscript{5}This specific security model has been chosen as this paper is the follow-up work of [33], where Shen et al. have analyzed the multi-user security of DbHtS in the ideal-cipher model.
that each of the $n$-bit keyed hash functions is $\epsilon_{\text{reg}}$-regular and $\epsilon_{\text{univ}}$-universal, then the multi-user security of our construction in the ideal-cipher model is of the order, assuming $q^{4/3} \leq 2^n, p \leq 2^{3k/4}, k \geq n$,

\[
\frac{9q^{4/3}}{2^n} + \frac{2u^2}{2^{n+k}} + \frac{2(q + p)}{2^k} + 5q^{4/3}\epsilon_{\text{univ}} + \frac{q^2\epsilon_{\text{reg}}^2}{2} + \frac{2q\epsilon_{\text{reg}}}{2^k} + \frac{pq^{1/3}/2^k + pq^{5/3}}{2^{n+k}},
\]

where $q$ is the total number of MAC queries across all $u$ users, $p$ is the total number of ideal-cipher queries, $n$ is the block size of the block cipher, $k$ is the size of the hash key and $k$ is the key size of the block cipher of the construction. As an instantiation of the Two-Keyed-DbHtS framework, we have proved that $C_2[\text{PH-DbH, E}]$, the Polyhash-based Two-Keyed-DbHtS construction which was proposed in [14] and proven to be secure up to $2^{3n/4}$ queries in the single-user setting, is multi-user secure up to $2^{3n/4}$ queries in the ideal-cipher model. The security proof of the construction crucially depends on a refined result of mirror theory over an abelian group $\{(0,1)^n, \oplus\}$, where one systematically estimates the number of solutions to a system of equations to prove the security of the finalization function of the construction up to $2^{3n/4}$ queries. Due to the attack result of Leurent et al. [22] on the DbHtS paradigm with $2^{3n/4}$ queries, the multi-user security bound of our construction is tight.

How this paper departs from [33]. Our work departs from the result of Shen et al. [33] in two aspects: (i) Unlike the result shown in [17], the birthday bound attack of [17] is not valid for our choice of double block hash function by the virtue of the definition of the good double block hash function. (ii) Unlike [33] where the DbH function of the Two-Keyed-DbHtS was instantiated with block cipher based DbH function, we have instantiated the construction with an algebraic type double block hash function. The merit of our choice of instantiation follows from the fact that the design of algebraic type DbH function does not require any block cipher and hence the adversary can get away with ideal cipher queries to the DbH function while bounding its regular and universal advantage.

Organization. We have developed the required notations and security definitions of cryptographic primitives in Sect. 2. We demonstrate the construction and present its security bound in Sect. 3 and in Sect. 4, we prove the security of the construction. We instantiate the framework along with its security result in Sect. 5.

2 Preliminaries

General Notations. For a positive integer $q$, $[q]$ denotes the set $\{1, \ldots, q\}$, and for two natural numbers $q_1, q_2$ such that $q_2 > q_1$, $[q_1, q_2]$ denotes the set $\{q_1, \ldots, q_2\}$. For a fixed positive integer $n$, we write $\{0,1\}^n$ to denote the set of all binary strings of length $n$ and $\{0,1\}^* = \cup_{n\geq 0}\{0,1\}^n$ to denote the set of all binary strings with arbitrary finite length. We refer to the elements of $\{0,1\}^n$ as blocks. For a pair of blocks $x = (x_l, x_r) \in \{0,1\}^{2n}$, we write $\text{left}(x)$ to denote $x_l$ and $\text{right}(x)$ to denote $x_r$. For any element $x \in \{0,1\}^*$, $|x|$ denotes the number of bits in $x$ and for $x, y \in \{0,1\}^*$, $x \parallel y$ denotes the concatenation of $x$ followed by $y$. We denote the bitwise xor operation of $x, y \in \{0,1\}^n$ by $x \oplus y$. We parse $x \in \{0,1\}^*$ as $x = x_1 || x_2 || \ldots || x_l$, where for each $i = 1, \ldots, l$, $x_i$ is a block and $1 \leq |x_i| \leq n$. For $x \in \{0,1\}^n$, $\text{lsb}(x)$ denotes the least significant bit $x_0$ of $x$. For a given bit $b$, $\text{fix}_b$ is a function from $\{0,1\}^n$ to $\{0,1\}^n$ that takes an $n$-bit binary string $x = x_n || x_{n-1} || \ldots || x_0$ and returns another binary string $x' = (x_n-1 || \ldots || b)$, where $\text{lsb}(x)$ is fixed to bit $b$. Given a tuple $\tilde{x} = (x_1, x_2, \ldots, x_q)$ of $n$-bit binary strings, an element $x_i$ of the tuple $\tilde{x}$ is said to be non-fresh, if there exists at least one $j \neq i$ such that $x_i = x_j$. Otherwise, we call that element $x_i$ is fresh.
Given a finite set $S$ and a random variable $X$, we write $X \leftarrow S$ to denote that $X$ is sampled uniformly at random from $S$. We say that $X_1, X_2, \ldots, X_q$ are sampled with replacement (wr) from $S$, which we denote as $X_1, X_2, \ldots, X_q \leftarrow S$, if for each $i \in [q], X_i \leftarrow S$. We also use this notation to denote that these random variables are sampled uniformly and independently from $S$. For a finite subset $S$ of $\mathbb{N}$, $\max S$ denotes the maximum-valued element of $S$. $\emptyset$ denotes the empty set. We write $S \leftarrow \emptyset$ to denote that $S$ is defined to be an empty set. We also use the same notation $\Phi \leftarrow \emptyset$ to denote that the function $\Phi$ is undefined at every point of its domain. Moreover, the notation $Y \leftarrow X$ is used to denote the assignment of variable $X$ to $Y$.

The set of all functions from $X$ to $Y$ is denoted by $\text{Func}(X, Y)$. Similarly, the set of all permutations over $X$ is represented by $\text{Perm}(X)$. When $X = \{0, 1\}^n$, then we write $\text{Perm}(X)$ as $\text{Perm}$. A function $\Phi$ is said to be a block function if it maps elements from an arbitrary domain to $\{0, 1\}^n$. The set of all block functions with domain $X$ is denoted as $\text{Func}(X)$. We call $\Phi$ to be a double-block function if it maps elements from an arbitrary set $X$ to $\{(0, 1)^n\}^2$. For a given double-block function $\Phi : X \to \{0, 1\}^2$, we write $\Phi(x) = (\Phi_1(x), \Phi_2(x))$. For two block functions $\Phi_1, \Phi_2 : X \to \{0, 1\}^n$ and $\Phi : X \to \{0, 1\}^n$, one can naturally define a double-block function $\Phi : X \to \{0, 1\}^n$ such that $\Phi(x) = (\Phi_1(x), \Phi_2(x))$, which we write as $\Phi = (\Phi_1, \Phi_2)$. For a finite set $X$ and an integer $q$, we write $X^{(q)}$ to denote the set $\{(x_1, x_2, \ldots, x_q) : x_i \in X, x_i \neq x_j\}$. For integers $1 \leq b \leq a$, we write $P(a, b)$ to denote $a(a-1)\ldots(a-b+1)$, where $P(a, 0) = 1$ by convention. Therefore, $|X^{(q)}| = P(|X|, q)$.

2.1 Distinguishing Advantage

An adversary $A$ is modeled as a randomized algorithm with access to an external oracle $O$. Such an adversary is called an oracle adversary. An oracle $O$ is an algorithm that may be a cryptographic scheme being analyzed. The interaction between $A$ and $O$, denoted by $A^O$, generates a transcript $\tau = ((x_1, y_1), (x_2, y_2), \ldots, (x_q, y_q))$, where $x_1, x_2, \ldots, x_q$ are $q$ queries of $A$ to oracle $O$ and $y_1, y_2, \ldots, y_q$ be the corresponding responses, where $y_i = O(x_i)$. We assume that $A$ is adaptive, which means that $x_i$ is dependent on the previous $i - 1$ responses.

Distinguishing Game. Let $F$ and $G$ be two random systems and an adversary $A$ is given oracle access to either of $F$ or $G$. After interaction with an oracle $O \in \{F, G\}$, $A$ outputs 1, which is denoted as $A^O \Rightarrow 1$. Such an adversary is called a distinguisher and the game is called a distinguishing game. The task of the distinguisher in a distinguishing game is to tell with which of the two systems it has interacted. The advantage of the distinguisher $A$ in distinguishing the random system $F$ from $G$ is defined as

$$\text{Adv}^F_G(A) \doteq | \Pr[A^F \Rightarrow 1] - \Pr[A^G \Rightarrow 1] |,$$

where the above probability is defined over the probability spaces of $A$ and $O$. One can easily generalize this setting when the distinguisher interacts with multiple oracles, which are separated by commas. For example, $\text{Adv}^{F_1, \ldots, F_m}_{G_1, \ldots, G_m}(A)$ denotes the advantage of $A$ in distinguishing the oracles $(F_1, \ldots, F_m)$ from the oracles $(G_1, \ldots, G_m)$, i.e.,

$$\text{Adv}^{F_1, \ldots, F_m}_{G_1, \ldots, G_m}(A) \doteq | \Pr[A^{F_1, \ldots, F_m} \Rightarrow 1] - \Pr[A^{G_1, \ldots, G_m} \Rightarrow 1] |,$$

where the above probability is defined over the probability spaces of $A$ and the oracle $O \in \{(F_1, \ldots, F_m), (G_1, \ldots, G_m)\}$.

\(^6\)When $X = \{0, 1\}^n$, we write Func to denote $\text{Func}(\{0, 1\}^n)$. 

2.2 Block Cipher

A block cipher $E : \mathcal{K} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ is a function that takes a key $k \in \mathcal{K}$ and an $n$-bit input data $x \in \{0, 1\}^n$ and produces an $n$-bit output $y$ such that for each key $k \in \mathcal{K}$, $E(k, \cdot)$ is a permutation over $\{0, 1\}^n$. $\mathcal{K}$ is called the key space of the block cipher and $\{0, 1\}^n$ is its input-output space. In short-hand notation, we write $E_k(x)$ to represent $E(k, x)$. Let $BC(\mathcal{K}, \{0, 1\}^n)$ denote the set of all $n$-bit block ciphers with key space $\mathcal{K}$. We say that a block cipher $E$ is an $(q, \epsilon, t)$-secure strong pseudorandom permutation (SPRP), if for all distinguishers $A$ that make a total of $q$ queries to its oracles with run time at most $t$, the following holds:

$$\text{Adv}^{\text{SPRP}}_E(A) \overset{\Delta}{=} \text{Adv}^{E_K, E_K^{-1}}_{\Pi, \Pi^{-1}}(A) \leq \epsilon,$$

where the probability is defined over $K \leftarrow \mathcal{K}$, $\Pi \leftarrow \text{Perm}$, and the randomness of the adversary $A$ (if any).

2.3 PRF Security in the Ideal-Cipher Model

A keyed function family with the key space $\mathcal{K}$, domain $\mathcal{X}$ and range $\mathcal{Y}$ is a function $F : \mathcal{K} \times \mathcal{X} \rightarrow \mathcal{Y}$. We denote $F(k, x)$ by $F_k(x)$. A random function $\text{RF}$ from $\mathcal{X}$ to $\mathcal{Y}$ is a uniform random variable over the set $\text{Func}(\mathcal{X}, \mathcal{Y})$, i.e., $\text{RF} \leftarrow \text{Func}(\mathcal{X}, \mathcal{Y})$. We define the pseudorandomness security of $F$ under the ideal-cipher model. We assume that $F$ makes internal calls to a publicly evaluated block cipher $E$ with more than one key. Typically, $F$ would be keyed with some key $K$ and derive block cipher keys $K_1, K_2, \ldots, K_m$ as a function of $K$ and other inputs ($F$ can make internal calls to multiple block ciphers when all of them are independently and uniformly distributed over the set $\text{BC}(\mathcal{K}, \{0, 1\}^n)$). For simplicity, we write $F^*_K$ to denote $F$ with a uniformly sampled block cipher $E \leftrightarrow \text{BC}((\mathcal{K}, \{0, 1\}^n))$, which is keyed by a randomly sampled key $K \leftarrow \mathcal{K}$. The distinguisher $A$ is given access to either $(F^*_K, E^\pm)$ for $K \leftarrow \mathcal{K}$ or $(\text{RF}, E^\pm)$ for $\text{RF} \leftarrow \text{Func}(\mathcal{X}, \mathcal{Y})$, where $E \leftrightarrow \text{BC}(\mathcal{K}, \{0, 1\}^n)$ is a uniformly sampled $n$-bit block cipher such that $A$ can make forward or inverse queries to $E$, which is denoted as $E^\pm$. We define the prf-advantage of $A$ against the keyed function family $F$ in the ideal cipher model as

$$\text{Adv}^{\text{PRF-ICM}}_F(A) \overset{\Delta}{=} \text{Adv}^{(F^*_K, E^\pm)}_{(\text{RF}, E^\pm)}(A),$$

for $K \leftarrow \mathcal{K}$, $\text{RF} \leftarrow \text{Func}(\mathcal{X}, \mathcal{Y})$, $E \leftrightarrow \text{BC}(\mathcal{K}, \{0, 1\}^n)$ and the randomness of the adversary $A$ (if any). We say that $F$ is a $(q, p, \epsilon, t)$-PRF in the ideal cipher model if $\text{Adv}^{\text{PRF-ICM}}_F(A) \leq \epsilon$ for all adversaries $A$ that make $q$ queries to $F$, $p$ forward and inverse offline queries to $E$ and run for time at most $t$.

2.4 Multi-User PRF Security in the Ideal-Cipher Model

We assume there are $u$ users in the multi-user setting, such that the $i$-th user executes $F^*_K$. Furthermore, the $i$-th user key $K_i$ is independent of the keys of all other users. An adversary $A$ has access to all the $u$ users as oracles. $A$ makes queries to the oracles in the form of $(i, M)$ to the $i$-th user and obtains $T \leftarrow F^*_K(M)$. We call these construction queries. For $i \in [u]$, we assume $A$ makes $q_i$ queries to the $i$-th oracle, where we assume that $A$ is the deterministic adversary that achieves the maximum distinguishing advantage and we bound the distinguishing advantage of our construction with respect to this adversary.\(^7\)

\(^7\)In more details, the number of queries $A$ makes for each user might not be fixed and might vary for each transcript. Hence, all the probability calculations should have been carried out by conditioning on $q_i$ such that $q_1 + \ldots + q_u \leq q$, where $q_i$ is a random variable that depends on the choice of the adaptive adversary. However, the conditional probability yields $q_i$ free term and hence the unconditional probability will also be free from $q_i$. Nevertheless, such a detailed analysis is usually suppressed in almost every other research works in this direction.
We also assume that \( A \) make queries to the underlying block cipher \( E \) and its inverse with some chosen keys \( k^j \). We call these \textit{primitive queries}. Suppose \( A \) chooses \( s \) distinct block cipher keys \((k^1, \ldots, k^s)\) and makes \( p_j \) primitive queries to the block cipher \( E \) with chosen keys \( k^j \) for \( 1 \leq j \leq s \). We call \( A \) to be a \((u, q, p, t)\)-adversary against the multi-user PRF security of \( F \) in the ideal cipher model, where \( q = q_1 + \ldots + q_u \) is the total number of construction queries across all \( u \) users and \( p = p_1 + \ldots + p_s \) is the total number of primitive queries to the block cipher \( E \) with the total running time of \( A \) being at most \( t \). We assume that for any \( i \in [u] \), \( A \) does not repeat any construction query to the \( i \)-th user. Similarly, \( A \) does not repeat any primitive query for any chosen block cipher key \( k^j \) to the block cipher \( E \). The advantage of \( A \) in distinguishing \((F^E_{K_{k^1}}, F^E_{K_{k^2}}, \ldots, F^E_{K_{k^s}}, E^\pm)\) from \((RF_1, RF_2, \ldots, RF_u, E^\pm)\) in the multi-user setting, is defined as

\[
\text{Adv}^\text{muPRF-ICM}_F(A) \triangleq \max_A \text{Adv}^\text{ICM}_{F, (RF_1, \ldots, RF_u, E^\pm)}(A),
\]

for the \( u \) tuple of independently sampled keys \( K_{k^1}, \ldots, K_{k^s} \), \( u \) independently sampled random functions \( RF_1, RF_2, \ldots, RF_u \), a randomly chosen block cipher \( E \leftarrow BC(K, \{0,1\}^n) \) from the set of all block ciphers with \( k \)-bit key and \( n \)-bit input, and the randomness of the adversary (if any). We write

\[
\text{Adv}^\text{muPRF-ICM}_F(u, q, p, t) \triangleq \max_A \text{Adv}^\text{muPRF-ICM}_F(A),
\]

where the maximum is over all \((u, q, p, t)\)-adversaries \( A \). In this paper, we skip the time parameter of the adversary as we shall assume that the adversary is computationally unbounded. This also leads to the assumption that the adversary is deterministic. When \( u = 1 \), \( \text{Adv}^\text{muPRF-ICM}_F(1, q, p, t) \) boils down to the PRF distinguishing advantage of the keyed function family \( F \) in the ideal-cipher model as defined in Eqn. (1).

### 2.5 Security of a Keyed Hash Function

Let \( K_h \) and \( X \) be two non-empty finite sets. A keyed function \( H : K_h \times X \to \{0,1\}^n \) is \( \epsilon_{\text{axu}} \)-almost-xor universal (axu) if for any distinct \( x, x' \in X \) and for any \( \Delta \in \{0,1\}^n \),

\[
\text{Pr}[K_h \leftarrow K_h : H_{K_h}(x) \oplus H_{K_h}(x') = \Delta] \leq \epsilon_{\text{axu}}.
\]

Moreover, \( H \) is an \( \epsilon_{\text{univ}} \)-universal hash function if for any distinct \( x, x' \in X \),

\[
\text{Pr}[K_h \leftarrow K_h : H_{K_h}(x) = H_{K_h}(x')] \leq \epsilon_{\text{univ}}.
\]

A keyed hash function is said to be \( \epsilon_{\text{reg}} \)-regular if for any \( x \in X \) and for any \( \Delta \in \{0,1\}^n \),

\[
\text{Pr}[K_h \leftarrow K_h : H_{K_h}(x) = \Delta] \leq \epsilon_{\text{reg}}.
\]

\textbf{Remark 2.} We would like to note here that the above two definitions of keyed hash functions are defined in the standard model, in which there is no need of any interaction with the adversary. However, the above two definitions would require the involvement of the adversary, if the keyed hash functions are build on the top of block ciphers and the security is analyzed in the ideal-cipher model.

### 2.6 Mirror Theory

Mirror theory is a collection of combinatorial results that give a lower bound on the number of solutions to a system of bivariate affine equations \( E \) over an abelian group \((\{0,1\}^n, \oplus)\). We represent a system of equations by a simple graph \( \mathcal{G} = (V, \mathcal{E}) \) containing no loops or multiple edges, where each vertex denotes an \( n \)-bit unknown (for a fixed \( n \)), and we
connect vertices $P$ and $Q$ with an edge labeled $\lambda \in \{0, 1\}^n$ if $P \oplus Q = \lambda \in \mathcal{E}$. For a path $\mathcal{L} = P_1 \xrightarrow{\lambda_1} P_2 \xrightarrow{\lambda_2} \ldots \xrightarrow{\lambda_{\ell}} P_\ell$ in the graph $\mathcal{G}$, we define the label of the path

$$\lambda(\mathcal{L}) = \lambda_1 \oplus \lambda_2 \oplus \ldots \oplus \lambda_{\ell}.$$ 

In this work, we focus on a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with certain properties as listed below:

1. $\mathcal{G}$ contains no isolated vertex, i.e., every vertex is incident with at least one edge.
2. The vertex set $\mathcal{V}$ is partitioned into two disjoint sets denoted by $\mathcal{P}$ and $\mathcal{Q}$, where there are no edges within the vertex set in partition $\mathcal{P}$ or in partition $\mathcal{Q}$. All edges connect a vertex in $\mathcal{P}$ to a vertex in $\mathcal{Q}$. We call such graphs bipartition graphs.
3. $\mathcal{G}$ contains no cycle.
4. $\lambda(\mathcal{L}) \neq 0^n$ for any path $\mathcal{L}$ in $\mathcal{G}$.

Any bipartition graph $\mathcal{G}$ satisfying the above properties shall be called a good graph. Note that a good bipartition graph $\mathcal{G}$ contains no cycle. Therefore, $\mathcal{G}$ can be decomposed into its connected components, all of which are trees; let

$$\mathcal{G} = C_1 \sqcup C_2 \sqcup \ldots \sqcup C_\alpha \sqcup D_1 \sqcup D_2 \sqcup \ldots \sqcup D_\beta$$

for some $\alpha, \beta \geq 0$, where $C_i$ denotes a component of size greater than 2, and $D_i$ denotes a component size of 2. We write $\mathcal{C} = C_1 \sqcup C_2 \sqcup \ldots \sqcup C_\alpha$ and $\mathcal{D} = D_1 \sqcup D_2 \sqcup \ldots \sqcup D_\beta$. For a given good bipartite graph, we define an associated system of bivariate equations as follows: each vertices of the graph represents a variable in the associated system of equations. If there is an edge $\{P_i, Q_j\} \in \mathcal{E}$ with label being $\lambda_{ij}$, then we include the equation $P_i \oplus Q_j = \lambda_{ij}$ into the associated system of equations.

**Definition 1.** Let $\mathcal{E}_\mathcal{G}$ be a system of equations induced by a good bipartite graph $\mathcal{G}$. An injective function $\Phi : \mathcal{P} \sqcup \mathcal{Q} \rightarrow \{0, 1\}^n$ is said to be an injective solution to $\mathcal{E}_\mathcal{G}$ if $\Phi(P_i) \oplus \Phi(Q_j) = \lambda_{ij}$ for all $\{P_i, Q_j\} \in \mathcal{E}$ with the label of the edge $\{P_i, Q_j\}$ being $\lambda_{ij}$.

We remark that assigning any value to a vertex in $\mathcal{P}$ allows the labeled edges to uniquely determine the values of all the other vertices in the component containing $\mathcal{P}$, since $\mathcal{G}$ contains no cycle. Moreover, the values in the same component are all distinct as $\lambda(\mathcal{L}) \neq 0^n$ for any path $\mathcal{L}$. Let $\mathcal{P}$ be any path of even length $\ell$, defined as follows:

$$\mathcal{P} = P_1 \xrightarrow{\lambda_1} Q_1 \xrightarrow{\lambda_{12}} P_2 \xrightarrow{\lambda_{23}} Q_2 \xrightarrow{\lambda_{34}} \ldots \xrightarrow{\lambda_{\ell-1,\ell}} Q_{\ell-1} \xrightarrow{\lambda_{\ell-1,\ell}} P_\ell.$$ 

Without loss of generality, let us assume that the value assigned to the vertex $P_s$ collides with the value assigned to the vertex $P_t$, where $s < t$. Let $x$ be the value assigned to the vertex $P_s$. Then, the value assigned to the vertex $P_t$ is

$$\Delta = x \oplus \lambda_{s,s} \oplus \lambda_{s,s+1} \oplus \ldots \oplus \lambda_{t-1,t}.$$ 

By our assumption we have $\Delta = x$, which implies

$$\lambda_{s,s} \oplus \lambda_{s,s+1} \oplus \ldots \oplus \lambda_{t-1,t} = 0^n,$$

that contradicts to fact that the label of the path

$$\mathcal{L}(P_s \rightarrow Q_s \rightarrow P_{s+1} \rightarrow Q_{s+1} \rightarrow \ldots \rightarrow Q_{t-1} \rightarrow P_t) \neq 0^n.$$ 

Hence, it ensures that once a value is assigned to a vertex in a component, the values assigned to all the other vertices of the same component are distinct. Therefore, the number
of possible assignments of distinct values to the vertices in $G$ is $\mathbf{P}(2^n, |\mathcal{P}| + |\mathcal{Q}|)$. One may expect that when such an assignment is chosen uniformly at random, it would satisfy all the equations in $G$ with probability $2^{-nq}$, where $q$ denotes the number of edges (i.e., equations) in $G$. Indeed, we can prove that the number of solutions is close to $\mathbf{P}(2^n, |\mathcal{P}| + |\mathcal{Q}|)/2^{nq}$, up to a certain error. Formally, we have the following result:

**Lemma 1.** Let $G$ be a good bipartition graph, and let $q$ and $q^c$ denote the number of edges of $G$ and $C$, respectively. Let $v$ be the number of vertices of $G$. If $q < 2^n/8$, then the number of solutions of $G$, denoted $h(G)$, satisfies

$$h(G)^{2^{nq}} \geq \left(1 - \frac{9(q^c)^2}{8 \cdot 2^n} - \frac{3q^c q^2}{2 \cdot 2^n} - \frac{q^2}{2^{2n}} - \frac{9(q^c)^2 q}{8 \cdot 2^{2n}} - \frac{8q^4}{3 \cdot 2^{3n}}\right).$$

We refer the reader to [21] for a proof of the lemma.

### 3 The Two-Keyed DbHtS Construction

In this section, we describe the Two-Keyed Double-block Hash-then-Sum or in short, Two-Keyed-DbHtS construction to build a beyond birthday bound secure variable input length PRF. Let $H^1: \mathcal{K}_h \times \{0, 1\}^* \rightarrow \{0, 1\}^n$ and $H^2: \mathcal{K}_h \times \{0, 1\}^* \rightarrow \{0, 1\}^n$ be two keyed hash functions. Based on $H^1$ and $H^2$, we define the Double-block Hash or in short DbH function $H: \mathcal{K}_h \times \mathcal{K}_h \times \{0, 1\}^* \rightarrow \{0, 1\}^{2n}$ as follows:

$$H_{(L_1, L_2)}(M) = (H^1_{L_1}(M), H^2_{L_2}(M)). \quad (2)$$

We compose this DbH function with a very simple and efficient single-keyed xor function $\text{XOR}_K(x, y) = E_K(x) \oplus E_K(y)$, where $E_K$ is an $n$-bit block cipher and the block cipher key $K$ is independent from the hash key $(L_1, L_2)$, to realize the two-Keyed-DbHtS construction as follows:

$$C_2[H, E]_{(L_1, L_2, K)}(M) := \text{XOR}_K(H^1_{L_1}(M), H^2_{L_2}(M)).$$

We use the name Two-Keyed-DbHtS construction, as we count the hash key as one key and the xor function requiring one key, which is independent of the hash key. Most of the beyond birthday bound secure variable input length PRFs like 2K-SUM-ECBC, 2K-PMAC_Plus, 2K-LightMAC_Plus are specific instantiations of the Two-Keyed-DbHtS paradigm. These constructions (with domain separation technique) have been proven secured up to $2^{2n/3}$ queries in the standard model [14] for a single-user setting. Later, in [21], Kim et al. have improved their bound up to $2^{2n/4}$ queries. In [33], all these three constructions (without domain separation technique) have been proven secured up to $2^{2n/3}$ queries in the ideal-cipher model for a multi-user setting. We note here that as the xor function is not a PRF over two blocks, we can not apply the traditional Hash-then-PRF composition result directly to analyze the security of the two-keyed DbHtS. Thus, we need a different type of composition result for the security analysis of the Two-Keyed-DbHtS construction that utilizes higher security properties of its underlying DbH function instead of having only the universal or regular property.

**Definition 2.** Let $H^1: \mathcal{K}_h \times \{0, 1\}^* \rightarrow \{0, 1\}^n$ and $H^2: \mathcal{K}_h \times \{0, 1\}^* \rightarrow \{0, 1\}^n$ be two $n$-bit keyed hash functions. We say that the double-block hash function $H: \mathcal{K}_h \times \mathcal{K}_h \times \{0, 1\}^* \rightarrow \{0, 1\}^{2n}$ defined in Eqn. (2) is **good** if it satisfies the following conditions:

- $H^1$ is a family of $\epsilon_{\text{reg}}$-regular and $\epsilon_{\text{univ}}$-universal functions.
- $H^2$ is a family of $\epsilon_{\text{reg}}$-regular and $\epsilon_{\text{univ}}$-universal functions.
- $H$ is cross-collision resistant, i.e., for every $M, M' \in \{0, 1\}^*$, $\Pr[L_1 \leftarrow \mathcal{K}_h, L_2 \leftarrow \mathcal{K}_h: H^1_{L_1}(M) = H^2_{L_2}(M')] = 0$. 


The first two conditions imply that the regular and universal advantages of both the hash functions should be negligible, whereas the last condition indicates that the first hash output for any message cannot collide with the second hash output. Having defined the Two-Keyed-DbHtS construction, we now state and prove its security. For the sake of brevity, we refer to the Two-Keyed-DbHtS construction $C_2 \| H, E_{(L_1, L_2, K)}$ by simply $C_2$ without mentioning the underlying hash function, the block cipher and their associated keys.

**Theorem 1.** Let $k, k_1$ and $n$ be three positive integers and $\mathcal{M}$ be a non-empty finite set. Let $E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ be an $n$-bit block cipher. Let $H : \{0, 1\}^{k_1} \times \{0, 1\}^{k_2} \times \{0, 1\}^* \rightarrow \{0, 1\}^{2n}$ be a good double-block hash function as defined in Eqn. (2). Then any computationally unbounded distinguisher making a total of $q$ construction queries across all $u$ users $^8$ and a total of $p$ primitive queries to the block cipher $E$ can distinguish $C_2$ from an $n$-bit uniform random function with prf advantage

$$\text{Adv}^{\text{unprf-icm}}_{C_2}(u, q, p, \ell) \leq \frac{q + q^2 \epsilon_{\text{univ}} + \frac{9q^{4/3}}{2n} + \frac{3q^{8/3}}{2} + \frac{8q^{4}}{2n} + \frac{9q^{7/3}}{8} + \frac{8q^{4}}{3 \cdot 2^{3n}}}{2} + \frac{2pq}{2^{n+k}} \frac{pq^{1/3}}{2k} + \frac{pq^{5/3}}{2^{n+k}} + \frac{2q \epsilon_{\text{reg}}}{2k} + \frac{2u^2}{2^{k+n}} + \frac{2\epsilon_{\text{reg}}}{2^{k+n}} + \frac{6q^{2/3} \epsilon_{\text{univ}}}{2} + \frac{q^2 \epsilon_{\text{univ}}}{2.}

By assuming $q^{1/3} \leq 2^a$, $p \leq 2^{3k/4}$, and $k \geq n$, we have:

$$\text{Adv}^{\text{unprf-icm}}_{C_2}(u, q, p, \ell) \leq \frac{9q^{4/3}}{2n} + \frac{2u^2}{2^{k+n}} + \frac{2(q + p)}{2k} + \frac{5q^{4/3} \epsilon_{\text{univ}}}{2} + \frac{q^2 \epsilon_{\text{univ}}}{2} + \frac{2q \epsilon_{\text{reg}}}{2k} \frac{(q + p)}{2k} + \frac{2pq}{2^{n+k}} + \frac{pq^{1/3}}{2k} + \frac{pq^{5/3}}{2^{n+k}}.$$

**Remark 3.** We would like to mention that the last condition of the definition of good hash function, i.e., the cross-collision condition of the hash function rules out the possibility of mounting birthday bound attacks on the Two-Keyed-DbHtS construction. As a result of Theorem 1, the attack of Guo and Wang [17] does not apply.

### 4 Proof of Theorem 1

We consider a computationally unbounded non-trivial deterministic distinguisher $A$ that interacts with a pair of oracles in either the real world or the ideal world, described as follows: in the real world, $A$ is given access to $u$ independent instances of the Two-Keyed-DbHtS construction, i.e., to a tuple of $u$ oracles $(C_2[H, E_{(L_1, L_2, K)}])_{i \in \{0, 1\}^*}$, where each $(L_1, L_2)$ is independent of $(L_1', L_2')$, $K^i$ is independent of $K^j$ and $E \leftarrow \mathcal{BC}(\mathcal{X}, \{0, 1\}^n)$ is an ideal block cipher. Additionally, $A$ has access to the oracle $E^* = E$, underneath the construction $C_2$. In the ideal world, $A$ is given access to (i) a tuple of $u$ independent random functions $(RF_1, \ldots, RF_u)$, where each $RF_i$ is the random function over $\{0, 1\}^*$ to $\{0, 1\}^n$ that can be equivalently described as a procedure that returns an $n$-bit uniform string on input of any arbitrary message, and (ii) the oracle $E^\# = E$, where $E \leftarrow \mathcal{BC}(\mathcal{X}, \{0, 1\}^n)$ is an ideal block cipher, sampled independent of the distribution of the sequence of $u$ independent random functions. In both worlds, the first oracle is called the construction oracle and the latter, the ideal cipher oracle. Using the ideal cipher oracle, a distinguisher $A$ can evaluate any

---

$^8$We have assumed that ‘$u$’ to be the total number of “queried” users. The total number of available users may be more than the number of queried users. However, the information of the set of non-queried users are independent over the transcript and hence the presence of the set of non-queried users do not affect the security bound of the construction.
query $x$ under its chosen key $J$. A query to the construction oracle is called a *construction query* and to that of the ideal cipher oracle is called an *ideal cipher query*. Note that $A$ can make either forward (i.e., it evaluates $E$ with a chosen key and input), or inverse ideal cipher queries (i.e., it evaluates $E^{-1}$ with a chosen key and input). The ideal oracle is depicted in Fig. 4.1 and Fig. 4.2.

### 4.1 Description of the Ideal World

The ideal world consists of two phases: (i) the online and (ii) the offline phase. Before the game begins, we sample $u$ independent functions $f_1, f_2, \ldots, f_u$ uniformly at random from the set of all functions $\text{Func}([0,1]^* \rightarrow [0,1]^n)$ that map an arbitrary-length string to an $n$-bit string. We also sample an $n$-bit block cipher $E$ from the set of all block ciphers with a $k$-bit key and an $n$-bit input. In the online phase, when the distinguisher makes the $a$-th construction query for the $i$-th user $M^i_a$ to the construction oracle, it returns $T^i_a \leftarrow f_i(M^i_a)$. Similarly, if the distinguisher makes a forward (resp. inverse) primitive query with a chosen block cipher key $J$ and an input $x$ to the ideal cipher oracle, it returns $E(J, x)$ (resp. $E^{-1}(J, x)$). However, if any response of the construction queries is an all-zero string $0^n$, then the bad flag $\text{Bad-Tag}$ is set to 1 and the game is aborted.

**Figure 4.1:** Online Phase of the Ideal oracle $\mathcal{O}_{\text{ideal}}$; Boxed statements denote bad events. Whenever a bad event is set to 1, the ideal oracle immediately aborts (denoted as $\perp$) and returns the remaining values of the transcript in an arbitrary manner. So, if the game aborts for some bad event, then its previous bad events must not have occurred.

After this interaction is over, the offline phase begins. In this phase, we sample $u$ pairs of dummy hash keys $(L_1, L_2) \in \mathcal{K}_h \times \mathcal{K}_h$ and $u$ dummy block cipher keys $(K^i) \in \mathcal{K}_c \leftrightarrow \mathcal{K}_c$, where $L_1$ (resp. $L_2$) is the left (resp. right) hash key for the $i$-th user and $K^i$ is its block cipher key. If the block cipher key and a left (resp. right) hash key of the $i_2$-th user collides with the block cipher key and left (resp. right) hash key of the $i_2$-th user, then we set the flag $\text{BadK}$ to 1 and abort the game. If the game is not aborted, then we can compute a pair of $2n$-bit hash values $(\Sigma_{a}, \Theta_{a})$ for all queries across $u$ users, where we often refer to $\Sigma_{a} \leftarrow H_{L_1}^{i}(M^i_a)$ as the left hash output and to $\Theta_{a} \leftarrow H_{L_2}^{i}(M^i_a)$ as the right hash output for the $a$-th query of the $i$-th user.

Now, if the block cipher key of the $i$-th user and the left hash or right hash output for its $a$-th query collides with some chosen ideal cipher key and one of the corresponding inputs of the forward ideal cipher query, then we set the bad flag $\text{Bad1}$ to 1 and abort the game.

---

8Defining the ideal world that aborts in H-Coefficient proofs is a standard practice in provable security papers including [14, 33].
For the $i$-th user, if the left or right hash outputs for two of its queries collide and the corresponding responses also collide with each other (i.e., $\Sigma_a^i = \Sigma_b^i, T_a^i = T_b^i$), then we consider it to be a bad event. Similarly, for a pair of users $i_1$ and $i_2$, if their left or right hash outputs collide with each other and the corresponding responses also collide with each other, then we again consider it to be a bad event. If at least one of the above bad events occurs, we set $\text{Bad}_2$ to 1 and abort the game. We also set another flag $\text{Bad}_3$ to 1 and abort the game if for the $i$-th user, the number of the pairs of queries whose either left or right hash outputs collide with each other is at least $q_i^{2/3}$, where $q_i$ is the number of queries made by the $i$-th user.

**Offline Phase of $O_{\text{ideal}}$**

```
1: $\{L_1^i,L_2^i\}_{i\in[u]} \leftarrow \mathcal{K}_h \times \mathcal{K}_h$; \quad \{K^i\}_{i\in[u]} \leftarrow \mathcal{K};
2: \quad$\text{if } \exists b \in \{1,2\}$ and $i_1 \neq i_2 \in [u]$ such that $K^{i_1} = K^{i_2} \land L^{i_1}_b = L^{i_2}_b$;
3: \quad then $\text{Bad}_1 \leftarrow 1$; \quad \perp;
4: \quad $\forall i \in [u], \forall a \in [q_i]$: $(\Sigma_a^i, \Theta_a^i) \leftarrow (H_1^i(M_a^i), H_2^i(M_a^i))$;
5: \quad if one of the following holds:
6: \quad \quad (a) (B.11): $\exists i \in [u], j \in [s], u[0]_{\alpha} \in \text{Dom}(E_{ij})$, such that $K^j = J^j \land \Sigma_a^j = u[0]_{\alpha}$;
7: \quad \quad (b) (B.12): $\exists i \in [u], j \in [s], u[1]_{\alpha} \in \text{Dom}(E_{ij})$, such that $K^j = J^j \land \Theta_a^j = u[1]_{\alpha}$;
8: \quad then $\text{Bad}_2 \leftarrow 1$; \quad \perp;
9: \quad if one of the following holds:
10: \quad \quad (a) (B.31): $\exists i \in [u]$, such that $|\{(a,b) : \Sigma_a^i = \Sigma_b^i\}| \geq q_i^{2/3}$;
11: \quad \quad (b) (B.32): $\exists i \in [u]$, such that $|\{(a,b) : \Theta_a^i = \Theta_b^i\}| \geq q_i^{2/3}$;
12: \quad then $\text{Bad}_3 \leftarrow 1$; \quad \perp;
13: \quad go to Fig. 4.3;
```

**Figure 4.2:** Offline Phase of the Ideal oracle $\mathcal{O}$: Boxed statements denote bad events. Whenever a bad event is set to 1, the ideal oracle immediately aborts (denoted as $\perp$) and returns the remaining values of the transcript in an arbitrary manner. So, if the game aborts for some bad event, then we may assume that the previous bad events have not occurred.

Finally, we set the flag $\text{Bad}_4$ to 1 if at least one of the following events holds: (a) for the $i$-th user, two left hash outputs collide and their corresponding right hash outputs also collide, or (b) for the $i$-th user, there exists a tuple of four query indices $a,b,c,d$ such that
We partition \(16\) Tight Multi-User Security Bound of Offline Phase of the Ideal oracle such that \(Q\) collide with any ideal cipher key. We also define two additional sets:

1. \(\Sigma^i_a\) cannot collide with any ideal cipher query, and
2. \(\Sigma^i_a\) is good, i.e., the hash outputs of two hash functions never collide, it immediately rules out the attack of [17].

If the game is not aborted at this stage, then it follows that none of the bad events have occurred. All the query-response pairs belong to exactly one of the sets \(Q^m\) or \(Q^\#\) as defined in lines 1 and 11 of Fig. 4.3, where \(Q^m\) is the set of all construction queries across all users such that the block cipher key of the \(i\)-th user collides with an ideal cipher key, but none of its hash outputs collide with any ideal cipher query, and \(Q^\#\) is the set of all construction queries across all users such that the block cipher key of the \(i\)-th user does not collide with any ideal cipher key. We also define two additional sets: \(I^m\) and \(I^\#\) for \(Q^m\) and \(Q^\#\), where \(I^m\) (resp. \(I^\#\)) is the set of all \(i\) such that \((i, \ast) \in Q^m\) (resp. \((i, \ast) \in Q^\#\)). We partition \(I^m\) into \(r\) non-empty equivalence classes \(I^m_1, I^m_2, \ldots, I^m_r\) based on the relation that the \(i\)-th user key \(K^i\) collides with \(J^j\) if and only if \(i \in I^m_j\). It is to be noted that we

defined in lines 1 and 11 of Fig. 4.3, where \(Q^m\) is the set of all construction queries across all users such that the block cipher key of the \(i\)-th user collides with an ideal cipher key, but none of its hash outputs collide with any ideal cipher query, and \(Q^\#\) is the set of all construction queries across all users such that the block cipher key of the \(i\)-th user does not collide with any ideal cipher key. We also define two additional sets: \(I^m\) and \(I^\#\) for \(Q^m\) and \(Q^\#\), where \(I^m\) (resp. \(I^\#\)) is the set of all \(i\) such that \((i, \ast) \in Q^m\) (resp. \((i, \ast) \in Q^\#\)). We partition \(I^m\) into \(r\) non-empty equivalence classes \(I^m_1, I^m_2, \ldots, I^m_r\) based on the relation that the \(i\)-th user key \(K^i\) collides with \(J^j\) if and only if \(i \in I^m_j\).

A correct way of writing step 2 of Fig. 4.3 should be \(I^m = I^m_1 \cup I^m_2 \cup \ldots \cup I^m_r\), where \(i_1, i_2, \ldots, i_r \in [s]\) such that \(i \in I^m_i \Rightarrow K^i = J^{i_i}\). However, for the sake of simplicity, we assume \(i = t, t \in [r]\), i.e., each of the first \(r\) many chosen ideal cipher keys have been collided with at least one user key.
assume that the adversary has chosen a total of $s$ distinct ideal cipher keys $J^1, J^2, \ldots, J^s$ during the evaluation of primitive queries and out of these $s$ chosen keys, each of the $r \leq s$ keys are collided with at least one user key. Similarly, we partition $\mathcal{I}^\phi$ into $r'$ equivalence classes based on the equivalence relation $i \sim j$ if and only if $K^i = K^j$. Now, for the $j$-th equivalence class of $\mathcal{I}^\phi$, we consider the tuple

$$
\Sigma_j := \bigcup_{i \in \mathcal{I}^\phi_j} \{ (\Theta_1^i, \Theta_2^i, \ldots, \Theta_q^i) \}, \quad \bar{\Theta}_j := \bigcup_{i \in \mathcal{I}^\phi_j} \{ (\Theta_1^i, \Theta_2^i, \ldots, \Theta_q^i) \}.
$$

Note that due to the event in line number 7.(b) (resp. 7.(d)) of Fig. 4.2, we have $\Sigma^t_i \neq \Sigma^s_i$ (resp. $\Theta^t_i \neq \Theta^s_i$) for $i_1, i_2 \in \mathcal{I}^\phi_j$ and $a \in [q_{i_1}], b \in [q_{i_2}]$. If $\Sigma^s_i$ is not fresh in the tuple $(\Sigma^1_i, \Sigma^2_i, \ldots, \Sigma^q_i)$ for some $(i, a) \in \mathcal{I}^\phi_j \times [q_i]$ and the output of $\Theta^s_i$ has not been sampled yet, then we sample its output $Z^1_{1,a}$ from outside the range of $\mathcal{E}_J$, and set the output of $\Theta^s_i$ as the xor of $Z^1_{1,a}$ and $Z^1_{2,a}$ (see line 6 of Fig. 4.3). Otherwise, we set the output of $\Sigma^s_i$ to the already defined element and adjust the output of the other hash value accordingly (see line 7 of Fig. 4.3). Note that in the latter case, we do not sample the output. In the above adjustment, if the output of $\Theta^s_i$ happens to collide with any previously sampled output, then we set flag $\text{Bad-Samp}$ to 1 and abort the game (see line 8 of Fig. 4.3) and abort the game. Note that this event cannot hold for the real oracle, as $\Theta^s_i$ is fresh in $(\Theta^1_i, \Theta^2_i, \ldots, \Theta^q_i)$ for $i \in \mathcal{I}^\phi_j$ and $a \in [q_i]$. If the above flag is not set to 1, then the sampling for the output of $\Sigma^s_i$, where $(i, a) \in \mathcal{Q}^\phi$ preserves permutation compatibility. Finally, for all other $(i, a) \in \mathcal{Q}^\phi$, we sample $Z^1_{1,a}$ and $Z^2_{2,a}$ such that $Z^1_{1,a} \oplus Z^2_{2,a} = T^a_i$.

### 4.2 Attack Transcript

We summarize here, the interaction between the distinguisher and the challenger in a transcript. The set of all construction queries for $u$ instances are summarized in a transcript $\tau_c = \tau^c_1 \cup \tau^c_2 \cup \ldots \cup \tau^c_u$, where $\tau^c_i = \{ (M^i_1, T^i_1), \ldots, (M^i_s, T^i_s) \}$ denotes the query-response transcript generated from the $i$-th instance of the construction. Moreover, we assume that $A$ has chosen $s$ distinct ideal cipher keys $J^1, \ldots, J^s$ such that it makes $p_j$ ideal cipher queries with the chosen key $J^j$. We summarize the ideal cipher queries in a transcript $\tau_p = \tau^p_1 \cup \tau^p_2 \cup \ldots \cup \tau^p_p$, where $\tau^p_j = \{ (u^j_1, v^j_1), \ldots, (u^j_r, v^j_r), J^j \}$ denotes the transcript of the ideal cipher queries when the chosen ideal cipher key is $J^j$. We assume that $A$ makes $q_i$ construction queries for the $i$-th instance and $p_j$ ideal cipher queries (including forward and inverse queries) with chosen ideal cipher key $J^j$. We also assume that the total number of construction queries across $u$ instances is $q$, i.e., $q = (q_1 + \ldots + q_u)$ and the total number of ideal cipher queries is $p = (p_1 + \ldots + p_u)$. Since $A$ is non-trivial, none of the transcripts contain any duplicate elements.

We modify the experiment by releasing internal information to $A$ after it has finished its interaction but has not yet output the decision bit. In the real world, we reveal all the keys $(L_1, L_2, K')$ for all $u$ instances used in the construction. In the ideal world, we sample them uniformly at random from their respective key spaces and reveal them to the distinguisher. We also reveal the tuple $(\Sigma^0_i, \Psi^0_i, \Psi_j(\Sigma^0_i))$ to the distinguisher $A$, where $i \in \mathcal{I}^\phi_j$ or $i \in \mathcal{I}^\phi_j$. We note that the tuple $(\Sigma^0_i, \Theta^0_i, \Psi_j(\Sigma^0_i), \Psi_j(\Theta^0_i))$ is computed by the challenger of both the world, where the function $\Psi_j$ defined for the ideal world is given in Fig. 4.3, and for the real world, we define $\Psi_j$ as follows:

$$
\Psi_j(\Sigma^0_i) = \mathcal{E}_{K'}(\Sigma^0_i), \quad \Psi_j(\Theta^0_i) = \mathcal{E}_{K'}(\Theta^0_i),
$$

for $i \in \mathcal{I}^\phi_j$ or $i \in \mathcal{I}^\phi_j$. By the virtue of the definition of the function $\Psi_j$ in the real and in the ideal world, for $i \in \mathcal{I}^\phi_j$ and $i' \in \mathcal{I}^\phi_j$ with $j \neq j'$, $\Sigma^0_i = \Sigma^0_i' \Rightarrow \Psi_j(\Sigma^0_i) = \Psi_j(\Sigma^0_i')$. Similarly, for $i \in \mathcal{I}^\phi_j$ and $i' \in \mathcal{I}^\phi_j$ with $j \neq j'$, $\Sigma^0_i = \Sigma^0_i' \Rightarrow \Psi_j(\Sigma^0_i) = \Psi_j(\Sigma^0_i')$. Hence, if
and there exists interpolation probability set of attainable transcripts. Suppose there exists a transcript \( \tau \). We call a transcript \( \tau \) if its realizing a transcript \( \Psi \). Following these notations, we now state the main theorem of the H-coefficient technique \[32\]:

GoodT is the set of all attainable good transcripts. We bound the probability of bad transcripts in the ideal world as follows.

GoodT \( \cup \) BadT be a partition of the set of attainable transcripts. Suppose there exists \( \epsilon_{\text{ratio}} \geq 0 \) such that for any \( \tau = (\tau_c, \tau_p) \in \text{GoodT} \),

\[
\frac{p_{\text{re}}(\tau)}{p_{\text{id}}(\tau)} = \frac{\Pr[\tau_c = \tau]}{\Pr[\tau_p = \tau]} \geq 1 - \epsilon_{\text{ratio}},
\]

and there exists \( \epsilon_{\text{bad}} \geq 0 \) such that \( \Pr[\tau_p = \tau] \leq \epsilon_{\text{bad}} \). Then

\[
\text{Adv}_{\Pi}^{\text{mprf}}(A) \leq \epsilon_{\text{ratio}} + \epsilon_{\text{bad}}.
\]

Therefore, to prove the security of the construction using the H-coefficient technique, we need to identify the set of bad transcripts and compute an upper bound for their probability in the ideal world. Then we find a lower bound for the ratio of the ideal to real interpolation probability for a good transcript. We have already identified the bad transcripts in Fig. 4.1 and Fig. 4.2. Therefore, it only remains to bound the probability of bad transcripts in the ideal world and provide a lower bound for the ratio of the ideal to real interpolation probability for a good transcript. Having explained the H-coefficient technique in the view of our construction, it follows that for each \( i \in [u], \ C_2(H, E_{i(L_1^i, L_2^i, K^i)} \mapsto \tau_i^c) \) denotes the following:

1. \( \Sigma_i^a = (H_{L_1^i}(M_u^i)), \Theta_i^a = (H_{L_2^i}(M_u^i)) \),
2. \( E_k(C_i^a) = \Psi_j(\Sigma_i^a), \text{ and } E_k(C_i^a) = \Psi_j(\Theta_i^a), \) for some \( j \) such that \( i \in I^u_j \) or \( i \in I^f_j \), and
3. \( E_k(\Sigma_i^a) \oplus E_k(\Theta_i^a) = T_i^a \).

### 4.3 Bounding the Probability of Bad Transcripts

We call a transcript \( \tau = (\tau_c, \tau_p) \) bad if at least one of the flags is set to 1 during the generation of the transcript in Fig. 4.1 and Fig. 4.2. Recall that BadT \( \subseteq \mathcal{V} \) is the set of all attainable bad transcripts and GoodT = \( \mathcal{V} \setminus \text{BadT} \) is the set of all attainable good transcripts. We bound the probability of bad transcripts in the ideal world as follows.
Lemma 2. Let $\tau = (\tau_e, \tau_p)$ be any attainable transcript. Let $X_{id}$ and $BadT$ be defined as above. Then

$$\Pr[X_{id} \in BadT] \leq \frac{q}{2^n} + \frac{2u^2}{2k_{h} + k} + \frac{2q\epsilon_{reg}}{2^n} + \frac{2q^2\epsilon_{reg}}{2k} + 4q^{4/3}\epsilon_{univ}$$

$$+ \frac{q^2\epsilon_{univ}}{2} + \frac{2pq}{2^n + k} + \frac{pq^{1/3}}{2k} + \frac{pq^{5/3}}{2n + k}.$$ 

Proof. By abusing the notation, we refer the bad events by their corresponding flag variables as defined in Fig. 4.1, Fig. 4.2 and Fig. 4.3. That is we use $Bad-Tag$ to refer to that event for which $Bad-Tag$ flag has been set to 1. In other words, we say that the event $Bad-Tag$ holds if and only if $Bad-Tag$ flag has been set to 1. Using the union bound, we write

$$\Pr[X_{id} \in BadT] \leq \Pr[Bad-Tag] + \Pr[BadK] + \sum_{i=1}^{4} \Pr[Badi | BadK] + \Pr[Bad-Samp | BadK].$$ (4)

In the following, we individually bound each bad event and then use Eqn. (4) to derive the result. In the subsequent analysis, we assume that $|K_{h}| = k_{h}$ and $|K| = k$.

A. Bounding Event $Bad-Tag$.

For a fixed choice of indices, the probability of the event can be bound by $1/2^n$ as the outputs of the construction queries are sampled uniformly and independently of other random variables. Therefore, by summing over all possible choices of indices, we have

$$\Pr[Bad-Tag] \leq \frac{q}{2^n}.$$ (5)

B. Bounding Event $BadK$.

For a fixed choice of indices, the probability of the event can be bound by $1/2^{k_{h} + k}$ as the event $K_{i}^{1} = K_{i}^{2}$ is independent of $L_{i}^{1} = L_{i}^{2}$ for each $b \in \{1, 2\}$. Therefore, summing over all possible choices of indices, we have

$$\Pr[BadK] \leq \frac{2u^2}{2k_{h} + k}.$$ (6)

C. Bounding Event $Bad1 | BadK$.

We say that the event $Bad1$ holds if either of the events defined in line 5.(a) or in line 5.(b) of Fig. 4.2 holds. We refer to the event defined in line 5.(a) as B.11 and refer to the event defined in line 5.(b) as B.12.

$> Bounding B.11 | BadK$: For a fixed choice of indices, $\Sigma_{a}^{i} = u[0]_{a}$, is bound by the regular advantage of the hash function $H_{L_{i}^{1}}$. As the hash key $L_{i}^{1}$ is independent of the block cipher key $K^{i}$, we have

$$\Pr[B.11 | BadK] \leq \sum_{i \in [s]} \sum_{j \in [s]} \Pr[K^{i} = J^{j}] \cdot \Pr[\Sigma_{a}^{i} = u[0]_{a}]$$

$$= \sum_{i \in [s]} \sum_{a \in [p]} \epsilon_{reg} \cdot \frac{1}{2k} \leq \frac{q\epsilon_{reg}}{2k},$$ (7)
Tight Multi-User Security Bound of DbHtS

where (1) holds due to the fact that \( q_1 + \ldots + q_u = q \) and \( p_1 + \ldots + p_s = p \).

**Bounding B.12 | BADK.** With an identical argument, one can show that the probability of the event B.12 can be bounded by \( \frac{q\epsilon_{\text{reg}}}{2^k} \), i.e.,

\[
\Pr[B.12 | \text{BADK}] \leq \frac{q\epsilon_{\text{reg}}}{2^k}. \tag{8}
\]

Therefore, by combining Eqn. (7) and Eqn. (8), we have

\[
\Pr[\text{Bad1} | \text{BADK}] \leq \Pr[B.11 | \text{BADK}] + \Pr[B.12 | \text{BADK}] \leq \frac{2q\epsilon_{\text{reg}}}{2^k}. \tag{9}
\]

**D. Bounding Event Bad2 | BADK.**

We say that the event BAD2 holds if either of the events defined in line 7.(a) or in line 7.(b) or line 7.(c) or in line 7.(d) of Fig. 4.2 holds. We refer to the event defined in line 7.(a) as B.21, in line 7.(b) as B.22, in line 7.(c) as B.23 and finally in line 7.(d) as B.24.

**Bounding B.21 | BADK.** For a fixed choice of indices, we analyze the probability of the event

\[
\Sigma_a^i = \Sigma_b^i \wedge T_a^i = T_b^i.
\]

Due to independence of the hash key \( L_1^i \) and \( T_a^i \), the probability of this joint event can be bound by the universal property of the \( H^1 \) hash function and the randomness of \( T_a^i \). Therefore,

\[
\Pr[B.21 | \text{BADK}] \leq \sum_{i \in [u], a, b \in [q_i]} \Pr[\Sigma_a^i = \Sigma_b^i \wedge T_a^i = T_b^i] \leq \frac{q^2\epsilon_{\text{univ}}}{2^{n+1}}. \tag{10}
\]

**Bounding B.22 | BADK.** We bound the event given \( \text{BADK} \), i.e. even if the block cipher keys for users \( i_1 \) and \( i_2 \) collide, their corresponding hash keys, i.e., \( L_1^{i_1} \) and \( L_2^{i_2} \) do not collide. Given this event, for a fixed choice of indices, we bound \( \Sigma_a^i = \Sigma_b^i \) using the regular property of the hash function \( H^1 \) with the randomness of the hash key \( L_1^i \). Moreover, the first event is independent of the second event and can thus be bound exactly by \( 2^{-k} \). Therefore,

\[
\Pr[B.22 | \text{BADK}] \leq \sum_{i_1, i_2 \in [u], a \in [q_i], b \in [q_{i_2}]} \epsilon_{\text{reg}} \cdot \frac{1}{2^k} \leq \frac{q^2\epsilon_{\text{reg}}}{2^k}. \tag{11}
\]

**Bounding B.23 | BADK and B.24 | BADK.** Bounding B.23 | BADK and B.24 | BADK is identical to bounding B.21 | BADK and B.22 | BADK respectively. Hence,

\[
\Pr[B.23 | \text{BADK}] \leq \frac{q^2\epsilon_{\text{univ}}}{2^{n+1}}, \quad \Pr[B.24 | \text{BADK}] \leq \frac{q^2\epsilon_{\text{reg}}}{2^k}. \tag{12}
\]

Therefore, by combining Eqn. (10)-Eqn. (12),

\[
\Pr[\text{Bad2} | \text{BADK}] \leq \Pr[B.21 | \text{BADK}] + \Pr[B.22 | \text{BADK}] + \Pr[B.23 | \text{BADK}] + \Pr[B.24 | \text{BADK}] \leq \frac{q^2\epsilon_{\text{univ}}}{2^n} + \frac{2q^2\epsilon_{\text{reg}}}{2^k}. \tag{13}
\]

**E. Bounding Event Bad3 | BADK.**
We say that the event Bad3 holds if either of the events defined in line 9.(a) or in line 9.(b) of Fig. 4.2 holds. We refer to the event defined in line 9.(a) as $B\text{.31}$ and in line 9.(b) as $B\text{.32}$.

**B.31. Bounding $B\text{.31}$ and $B\text{.32}$.** We first bound the event $B\text{.31}$ and $B\text{.32}$. For a fixed choice of indices, we define an indicator random variable $I_{a,b}$ which takes the value 1 if $\Sigma_a = \Sigma_b$, and 0 otherwise. Let $\bar{I} = \sum_{a \neq b} I_{a,b}$. By linearity of expectation,

$$E[\bar{I}] = \sum_{a \neq b} E[I_{a,b}] = \sum_{a \neq b} \Pr[\Sigma_a = \Sigma_b] \leq \frac{q^2 \epsilon_{\text{univ}}}{2}.$$ 

Now,

$$\Pr[B\text{.31} | B\text{.32}] \leq \sum_{i \in [u]} \Pr[(a, b) \in [q_i]^2 : \Sigma_a = \Sigma_b] \geq q^{2/3}$$

$$= \sum_{i = 1}^{u} \Pr[I_i \geq q_i^{2/3}] \leq \sum_{i = 1}^{u} \frac{q_i^2 \epsilon_{\text{univ}}}{2q_i^{2/3}} \leq \frac{q^{4/3} \epsilon_{\text{univ}}}{2},$$

where (1) holds due to the Markov inequality.

Similar to $B\text{.31}$ and $B\text{.32}$, we bound $B\text{.32}$ and $B\text{.33}$ as follows:

$$\Pr[B\text{.32} | B\text{.32}] \leq \frac{q^{4/3} \epsilon_{\text{univ}}}{2}. \quad (15)$$

Therefore, by combining Eqn. (14) and Eqn. (15), we have

$$\Pr[B\text{.33} | B\text{.31} | B\text{.32}] \leq \frac{q^{4/3} \epsilon_{\text{univ}}}{2}. \quad (16)$$

**F. Bounding Event Bad4.**

We say that the event Bad4 holds if either of the events defined in line 11.(a) or in line 11.(b) or in line 11.(c) of Fig. 4.2 holds. We refer to the event defined in line 11.(a) as $B\text{.41}$, line 11.(b) as $B\text{.42}$, and in line 11.(c) as $B\text{.43}$.

**B.41. Bounding $B\text{.41}$ and $B\text{.43}$.** Due to independence of the hash key $L_1$ and $L_2$, for a fixed choice of indices, the probability of this joint event can be bounded by the universal property of the individual hash functions $H_1$ and $H_2$. Therefore, varying over all possible choices of indices, we have

$$\Pr[B\text{.41} | B\text{.43}] \leq \sum_{i \in [u]} \Pr[\Sigma_a = \Sigma_b \land \Theta_a = \Theta_b] = \sum_{i \in [u]} \Pr[\Sigma_a = \Sigma_b] \cdot \Pr[\Theta_a = \Theta_b]$$

$$\leq \frac{q^2 \epsilon_{\text{univ}}}{2}. \quad (17)$$

**B.42. Bounding $B\text{.42}$.** We first bound the event $B\text{.42}$ and $B\text{.43}$. We bound this event given $B\text{.31}$. This results in the fact that for a fixed $i \in [u]$, the number of quadruples $(a, b, c, d)$ such that $\Sigma_a = \Sigma_b$, $\Sigma_c = \Sigma_d$ holds is at most $q_i^{1/3}$. For a fixed choice of such quadruples, the event $\Theta_a = \Theta_b$ holds with probability at most $\epsilon_{\text{univ}}$ due to the universal property of the hash function $H_2$. Therefore,

$$\Pr[B\text{.42} | B\text{.31} | B\text{.43}] \leq \sum_{i \in [u]} q_i^{4/3} \epsilon_{\text{univ}} \leq q^{4/3} \epsilon_{\text{univ}}. \quad (18)$$
Therefore, from Eqn. (14) and Eqn. (18), we have

\[ \Pr[B.42 \mid \text{BadK}] \leq \Pr[B.42 \mid B.31 \land \text{BadK}] + \Pr[B.31 \mid \text{BadK}] \leq \frac{3}{2} q^{4/3} \epsilon_{\text{univ}}. \quad (19) \]

Similar to $B.42$, we bound $B.43$ as follows:

\[ \Pr[B.43 \mid B.32 \land \text{BadK}] \leq q^{4/3} \epsilon_{\text{univ}}. \quad (20) \]

Therefore, from Eqn. (15) and Eqn. (20), we have

\[ \Pr[B.43 \mid \text{BadK}] \leq \Pr[B.43 \mid B.32 \land \text{BadK}] + \Pr[B.32 \mid \text{BadK}] \leq \frac{3}{2} q^{4/3} \epsilon_{\text{univ}}. \quad (21) \]

By combining Eqn. (17), Eqn. (19) and Eqn. (21), we have

\[ \Pr[\text{Bad4} \mid \text{BadK}] \leq q^{4/2} \epsilon_{\text{univ}} + 3q^{4/3} \epsilon_{\text{univ}}. \quad (22) \]

G. Bounding Event Bad-Samp.

We consider bounding this event as a union of several events, namely for a fixed $i \in [u], j \in [s]$ and $a \in [q]$, we define

\[ BS_{i,j,a} \triangleq K^i = J^j \land Z_a^i \oplus T_a^i \in \text{Ran}(E_f). \]

Note that, $Z_a^i \oplus T_a^i$ hits either the output of some primitive query $V_a^j$ with ideal-cipher key being $J^j$ such that $K^i = J^j$, where $a \in [p_j]$. We denote this event as $BS_{i,j,a}$. On the other hand, $Z_a^i \oplus T_a^i$ hits the output of some previously sampled $\hat{Z}_a^i$ or $\hat{J}_a^i$ with the corresponding block cipher key $K^i$ which in turn collides with the ideal-cipher key $J^j$. We denote this event as $BS_{i,j,a}$. Therefore, we have

\[ \Pr[\text{Bad-Samp}] \leq \sum_{(j,\alpha)} \sum_{(i,a)} \Pr[BS_{i,j,a}] + \sum_{(i,a')} \sum_{j} \sum_{(a,a')} \Pr[BS'_{i,j,a,a'}] \quad (23) \]

Now, we bound the probability of the event $BS_{i,j,a}$ and $BS'_{i,j,a,a'}$ as follows:

**CASE 1:**

\[ BS_{i,j,a} \triangleq K^i = J^j \land Z_a^i \oplus T_a^i = V_a^j. \]

By the simple union bound, we have

\[ \sum_{(j,\alpha)} \sum_{(i,a)} \Pr[BS_{i,j,a}] \leq \sum_{(j,\alpha)} \sum_{(i,a)} \frac{1}{2k} \cdot \frac{1}{2n - p_j} \leq \frac{2pq}{2n+k}, \quad (24) \]

where the number of choices for $(i, a)$ and $(j, \alpha)$ are at most $q$ and $p$, respectively and $p_j$ is assumed to be at most $2^{n-1}$. Thus, summing over all possible choices of $(i, j, a, \alpha)$ and by assuming $p_j \leq 2^{n-1}$, we have the result.

**CASE 2:**

\[ BS'_{i,j,a,a'} \triangleq K^{i'} = K^i \land Z_a^i \oplus T_a^i = \hat{J}_a^{i'}/\hat{Z}_a^{i'}. \]

We bound the probability of the event separately depending on the number of queries made by a user. (i) For users with “many” queries, we will argue that the probability of having key-collision for that user is low. (ii) For users with “not many” queries, we will argue that the probability of the above event achieves the desired bound. Detail analysis is as follows:
Let $q_i$ be the number of queries made by user $i$, where $i \in [u]$ and $q_1 + q_2 + \ldots + q_u = q$. W.l.o.g. let us assume that $q_1 \geq q_2 \geq \ldots \geq q_u$. We define the event $\text{Event}$ as follows: if the keys for any of the first $q^{1/3}$ users collide with a primitive query key we call $\text{Event}$ occurs. It is easy to see that
\[
\Pr[\text{Event}] \leq \frac{q^{1/3} \cdot p}{2^k}.
\] (25)
Thus, for the first $q^{1/3}$ users, we bound the probability of the required event by $\frac{2^{1/3} \cdot p}{2^k}$. The case for $u < q^{1/3}$ is trivial. For the remaining users, we bound the probability as follows:
\[
\sum_{i=q^{1/3}}^{u} \frac{p \cdot q_i^2}{2^{n+k}} \leq \frac{pq^{5/3}}{2^{n+k}}.
\] (26)
Here we use the fact that $q_i \leq q^{2/3}$ for all $i = q^{1/3}, \ldots, u$ (as $q_1 + \ldots + q_u = q$). By combining Eqn. (25) and Eqn. (26), we have
\[
\sum_{(i,i')} \sum_{j} \sum_{(a,a')} \Pr[\text{BS2}_{i,i',j,a,a'}] \leq \frac{pq^{1/3}}{2^k} + \frac{pq^{5/3}}{2^{n+k}}.
\] (27)
Finally, by combining Eqn. (23), Eqn. (24) and Eqn. (27), we have
\[
\Pr[\text{Bad-Samp}] \leq \frac{2pq}{2^{n+k}} + \frac{pq^{1/3}}{2^k} + \frac{pq^{5/3}}{2^{n+k}}.
\] (28)
Finally, the result follows by combining Eqn. (5)-Eqn. (28) and by assuming $k \geq n$.

Remark 4. We would like to mention here that during the probability analysis for bounding the event $\text{Bad-Samp}$, we considered only the case that $\text{Bad-Samp}$ happens after line-7 of Fig. 4.3, not after line-8. This is because occurrence of the event after line-8 of Fig. 4.3 implies that $Z_{1,a}$ matches with some previously sampled $Z$ values and that basically falls back to the case in line-7 of Fig. 4.3.

### 4.4 Analysis of Good Transcripts

In this section, we compute a lower bound for the ratio of the real to ideal interpolation probability for a good transcript. We first consider the set of transcripts $Q^w$. For each $j \in [r]$ and for each $i \in I_j^w$, we consider the sequence
\[
\tilde{\Sigma}_i := (\Sigma_1^i, \ldots, \Sigma_{q_u}^i), \tilde{\Theta}_i := (\Theta_1^i, \ldots, \Theta_{q_u}^i).
\]
From this sequence, we construct a bipartite graph $G_i$, where the nodes in one partition represent values $\Sigma_a^i$ and the nodes in other, $\Theta_a^i$. We put an edge between the node corresponding to $\Sigma_a^i$ and $\Theta_a^i$ with the label being $T_a^i$, where $\Sigma_a^i \oplus \Theta_a^i = T_a^i$. If $\Sigma_a^i = \Sigma_{a'}^i$, then we merge the corresponding nodes into a single node, and similarly for $\Theta_a^i = \Theta_{a'}^i$. This leads us to break the graph into $w_i$ components. As the transcript is good, it is easy to see that each component is acyclic (otherwise $\mathbf{B.41}$ would have been satisfied) and contains a path of length at most 3 (otherwise either $\mathbf{B.42}$ or $\mathbf{B.43}$ would have been satisfied). Due to $\mathbf{B.31} \land \mathbf{B.32}$, the component size is restricted up to $q^{2/3}$. Moreover, due to $\mathbf{B.11} \land \mathbf{B.12}$, each vertex of the graph, i.e., $\Sigma_a^i$ or $\Theta_a^i$ does not collide with the input of any ideal-cipher query such that the ideal-cipher key collides with the $i$-th user key. Hence, each vertex of the graph $G_i$ is fresh in the sense that they do not collide with the input of any ideal-cipher query. Note that, $\mathbf{B.21}$ (resp. $\mathbf{B.23}$) ensures the fact that if $\Sigma_a^i$ collides with $\Sigma_{a'}^i$ (resp. $\Theta_a^i$ collides with $\Theta_{a'}^i$), then $T_a^i$ must be distinct from $T_{a'}^i$, otherwise, bad event $\mathbf{B.41}$ would have been satisfied. On the other hand, $\mathbf{B.22}$ (resp. $\mathbf{B.24}$) implies that there should not be any
intersection between the equation variables corresponding to two different users whose keys have been collid
ed.

Let \( v_i \) be the total number of nodes of the graph \( G_i \). Similar to \( Q^\ast \), we consider \( Q^\ast' \). For each \( j \in [r'] \) and for each \( i \in I_j^\ast' \), consider the sequence

\[
\tilde{\Theta}^i := (\Theta^i_1, \Theta^i_2, \ldots, \Theta^i_n).
\]

Similar to \( G_i \), we construct a bipartite graph \( H_i \), one of whose partitions represents the nodes corresponding to \( \Sigma^i_a \) and the other, the nodes corresponding to \( \Theta^i_a \). We put an edge between the node corresponding to \( \Sigma^i_a \) and \( \Theta^i_a \) with the label being \( T_a^i \), where \( \Sigma^i_a \oplus \Theta^i_a = T_a^i \). However, if two nodes represent the same values, then we merge them into a single node.

Let \( w_i' \) be the number of components of \( H_i \) and \( v_i' \) be the total number of vertices. Then for a good transcript \( \tau = (\tau_c, \tau_p) \), realizing \( \tau \) is almost as likely in the real world as in the ideal world:

**Lemma 3 (Good Lemma).** Let \( \tau = (\tau_c, \tau_p) \in \text{GoodT} \) be a good transcript. Let \( X_{re} \) and \( X_{id} \) be defined as above. Then

\[
\frac{\Pr[X_{re} = \tau]}{\Pr[X_{id} = \tau]} \geq 1 - \frac{9q^{4/3}}{8 \cdot 2^n} - \frac{3q^{8/3}}{2 \cdot 2^{2n}} - \frac{q^4}{2^{2n}} - \frac{9q^{4/3}}{3 \cdot 2^{3n}}.
\]

**Proof.** We are now ready to calculate the real interpolation probability. For this, we must bound the total number of input-output pairs on which the block cipher \( E \) with different keys is executed. As the transcript releases the \( 2k_{h_b} \)-bit hash keys and the \( k \)-bit block cipher key for each user, it contributes to a term \( 2^{-(2k_{h_b}+k)} \) in the real interpolation probability calculation. Now, for each \( j \in [r] \), the block cipher \( E \) with key \( J^j \) is evaluated on a total of

\[
p_j + \sum_{i \in I_j^\ast'} v_i
\]

input-output pairs. For the remaining ideal cipher keys, with which none of the users’ block cipher keys have collided, we have \( p_j \) input-output pairs, which are fixed due to the evaluation of the block cipher with those ideal cipher keys. Moreover, for each \( j \in [r] \), the block cipher \( E \) is evaluated on a total of \( \sum_{i \in I_j^\ast'} v_i' \) input-output pairs with key \( K^j \). Summarizing the above,

\[
\Pr[X_{re} = \tau] = \prod_{i=1}^{u} \frac{1}{2^{2k_{h_b}+k}} \left( \prod_{j=1}^{r} \frac{1}{\Pr[2^n, p_j + \sum_{i \in I_j^\ast'} v_i]} \right) \prod_{j \in [r] \setminus [r]} \frac{1}{\Pr[2^n, p_j]} \left( \prod_{j=1}^{r'} \frac{1}{\Pr[2^n, \sum_{i \in I_j^\ast'} v_i']} \right).
\]

**Ideal Interpolation Probability:** The term \( \prod_{i=1}^{u} 2^{-nq_i} \), which is contributed to the ideal interpolation probability due to the sampling of responses of the adversarial query, samples \( 2k_{h_b} \)-bit hash keys and \( k \)-bit block cipher keys for all \( u \) users. For each \( j \in [r] \), and for each \( i \in I_j^\ast \), we construct the graph \( G_i \) as defined above. Now, we have the following claim:

**Claim 1.** For each \( j \in [r] \) and for each \( i \in I_j^\ast \), the graph \( G_i \) good.

**Proof.** First of all, note that graph \( G_i \) does not have any cycle of length 2, otherwise the bad event \( B.41 \) would have been satisfied. Moreover, every component of the graph has a path of length at most three, otherwise the bad event \( B.43 \) would have been satisfied. This excludes the possibility of existence of even length cycle in the bipartite graph \( G_i \). Moreover, due to \( B.21 \wedge B.23 \), the xor of the labels of any path of length two in the graph
$G_i$ is non-zero and due to $\text{B.41} \land \text{B.42} \land \text{B.43}$, it follows that the xor of the labels of any path of length three in the graph is non-zero. Hence, the graph $G_i$ good. \hfill \Box

Now, for each $j \in [r]$ and for each $i \in I_j^a$, we sample the value of a node for each component of the graph $G_i$. Hence, for $j \in [r]$, the total number of sampled points is

$$p_j + \sum_{i \in I_j^a} w_i.$$ 

Moreover, for each $j \in [s \setminus [r]$, the total number of sample points is $p_j$. Subsequently, we consider the set of transcripts $Q^{x}$. For each $j \in [r']$, and for each $i \in I_j^a$, we construct the graph $H_i$ as defined above. As we reasoned about the goodness of the graph $G_i$, the similar reasoning applies for the goodness of the graph $H_i$ too. This allows us to compute the set $S_j$ for each $j \in [r']$ as defined in line 15 of Fig. 4.3 (which is defined as the number of tuples $(Q'_a, R'_a)$ such that $Q'_a \oplus R'_a = T'_a$ for all $i \in I_j^a$ and for all $a \in \{q_i\}$). In summary,

$$\Pr[X_{id} = \tau] = \prod_{i=1}^{u} \frac{1}{2^{2n_i}} \prod_{i=1}^{u} \frac{1}{2^{2\alpha_n \tau k}} \left( \prod_{j=1}^{r} \Pr(2^{n_i}, p_j + \sum_{i \in I_j^a} w_i) \right) \cdot \prod_{j \in [s \setminus [r]} \frac{1}{\Pr(2^{n_i}, p_j)} \left( \prod_{j=1}^{r'} \frac{1}{|S_j|} \right).$$

(30)

**Calculation of the ratio:** By plugging in the value of $|S_j|$ from Lemma 1 into Eqn. (30) and then taking the ratio of Eqn. (29) to Eqn. (30), we have

$$p(\tau) = \prod_{i=1}^{u} 2^{nq_i} \cdot \prod_{j=1}^{r} \frac{\Pr(2^{n_i}, p_j + \sum_{i \in I_j^a} w_i)}{\Pr(2^{n_i}, p_j + \sum_{i \in I_j^a} v_j)} \cdot \prod_{j=1}^{r'} \frac{|S_j|}{\Pr(2^{n_i}, \sum_{i \in I_j^a} v_j)} \cdot \prod_{j=1}^{r'} \frac{\Pr(2^{n_i}, \sum_{i \in I_j^a} v_j')}{\Pr(2^{n_i}, \sum_{i \in I_j^a} v_j')} \cdot \frac{1 - \epsilon_j}{1 - \epsilon_j n \sum_{i \in I_j^a} (v_i - w'_i)},$$

where

$$\epsilon_j \triangleq \sum_{i \in I_j^a} \frac{9(q^2)^2}{8 \cdot 2n} + \frac{3q^2q_i^2}{2 \cdot 2n} + \frac{q_i^2}{22n} + \frac{9(q^2)^2q_i}{8 \cdot 22n} + \frac{8q_i^4}{3 \cdot 23n}.$$ 

(31)
Therefore, we have

\[
p(\tau) = \prod_{i=1}^{n} 2^{nq_i} \cdot \prod_{j=1}^{r'} P(2^n - p_j - \sum_{i \in \mathcal{I}_j} w_i, \sum_{i \in \mathcal{I}_j} (v_i - w_i)) \cdot \prod_{j=1}^{r'} \frac{1}{2} \sum_{i \in \mathcal{I}_j^\#} \prod_{j=1}^{r'} (1 - \epsilon_j)
\]

\[
= \prod_{j=1}^{r} P(2^n - p_j - \sum_{i \in \mathcal{I}_j} w_i, \sum_{i \in \mathcal{I}_j} (v_i - w_i)) \cdot \prod_{j=1}^{r'} \frac{1}{2} \sum_{i \in \mathcal{I}_j^\#} \prod_{j=1}^{r'} (1 - \epsilon_j)
\]

\[
\geq \left(1 - \sum_{j=1}^{r'} \epsilon_j \right) \geq 1 - \sum_{j=1}^{r'} \sum_{i \in \mathcal{I}_j} \left( \frac{9(q_i^2)}{8 \cdot 2^n} + \frac{3q_i^2}{2 \cdot 2^{2n}} + \frac{3q_i}{2 \cdot 2^{2n}} + \frac{9q_i^2}{8 \cdot 2^{2n}} + \frac{8q_i^4}{3 \cdot 2^{3n}} \right)
\]

\[
\geq 1 - \frac{9q_i^{4/3}}{8 \cdot 2^n} + \frac{3q_i^{8/3}}{2 \cdot 2^{2n}} + \frac{9q_i^2}{8 \cdot 2^{2n}} + \frac{8q_i^4}{3 \cdot 2^{3n}}
\]

where (1) holds due to Eqn. (31) and (2) holds due to the fact that \( q_i^c \leq q_i^{2/3} \) for all \( i \in \mathcal{I}_j \) such that \( j \in [r'] \). Note that for each \( j \in [r], \sum_{i \in \mathcal{I}_j} (v_i - w_i) \) denotes the total number of edges in the graph \( \bigcup_{i \in \mathcal{I}_j} G_i \), which is \( \sum_{i \in \mathcal{I}_j} q_i \). Similarly, for each \( j \in [r'], \sum_{i \in \mathcal{I}_j^\#} (v_i' - w_i') \) denotes the total number of edges in the graph \( \bigcup_{i \in \mathcal{I}_j^\#} H_i \), which is \( \sum_{i \in \mathcal{I}_j^\#} q_i \).

5 Tight Security Bound of Two-Keyed Polyhash based Db-HtS Construction

Two-keyed Polyhash-based Db-HtS construction \( C_2[PH-DbH, E] \), as proposed in [14], is the instantiation of the Two-Keyed-Db-HtS framework which is build on the Polyhash based double block hash function \( PH-DbH \). In [14], the PRF security of \( C_2[PH-DbH, E] \) has been proven to be roughly in the order of \( q^{3/2} \) in the single-user setting. In this section we improve its bound up to \( 2^{3n/4} \) queries in the multi-user setting. Moreover, the proof is based on the ideal cipher model. Before going to the security proof of the construction, we first revisit to the two-keyed Polyhash-based Db-HtS construction.

PolyHash [15, 7, 35] is a very efficient algebraic hash function. For a fixed natural number \( n \), it first samples an \( n \)-bit key \( L \) uniformly at random from \( \{0, 1\}^n \). To apply this function on a message \( M \in \{0, 1\}^* \), we first apply an injective padding function \( 10^* \) (i.e. append a bit 1 followed by a minimum number of zeroes to the message \( M \) so that the total number of bits in the padded message becomes a multiple of \( n \)). Let the padded message be \( M^* = M_1 \parallel M_2 \parallel \ldots \parallel M_l \), where \( l \) is the number of \( n \)-bit blocks in it. Then, we define the PolyHash function as follows:

\[
PH_L(M^*) \triangleq M_1 \cdot L^l \oplus M_2 \cdot L^{l-1} \oplus \ldots \oplus M_l \cdot L,
\]

where \( l \) is the number of blocks of \( M \) and the multiplications are defined in the field \( GF(2^n) \). Then PolyHash [27] is \( \ell/2^n \)-regular, \( \ell/2^n \)-axu and \( \ell/2^n \)-universal, as shown in the
following lemma, where \( \ell \) is the maximum number of message blocks (the proof of the lemma is related to a result on the number of distinct roots of a polynomial):

**Lemma 4.** Let \( \text{PH} \) be the PolyHash function as defined above. Then \( \text{PH} \) is \( \ell/2^n \)-regular, \( \ell/2^n \)-almost-xor universal and \( \ell/2^n \)-universal.

**Proof.** We first compute the regular advantage of the hash function. Clearly, for any \( \Delta \in \{0,1\}^n \), and any \( M \in \{0,1\}^* \), \( f(L) := \text{PH}(L,M) \oplus \Delta \) is a polynomial in \( L \) with constant term \( \Delta \) having degree at most \( l^1 \), where \( l \) is the number of message blocks of \( M \) after padding. Therefore, \( \text{PH}(L,M) = \Delta \) if and only if \( f(L) = 0 \). Since, \( f \) is a polynomial of degree at most \( l \), \( f(L) = 0 \) has at most \( l \) solutions. As a result, number of \( L \) such that \( \text{PH}(L,M) = \Delta \) holds is at most \( l \) and hence \( \epsilon_{\text{reg}} = \ell/2^n \), where \( \ell \) is the maximum number of message blocks among all messages. Moreover, for any two distinct messages \( M \) and \( M' \) and for any \( \Delta \in \{0,1\}^n \), \( f(L) := \text{PH}(L,M) \oplus \text{PH}(L,M') \oplus \Delta \) is a polynomial in \( L \) with degree at most \( l \), where \( l \) is the maximum number of message blocks among \( M \) and \( M' \) after padding. Therefore, \( \text{PH}(L,M) \oplus \text{PH}(L,M') = \Delta \) if and only if \( f(L) = 0 \). Since, \( f \) is a polynomial of degree at most \( l \), \( f(L) = 0 \) has at most \( l \) solutions and hence the number of \( L \) such that \( \text{PH}(L,M) \oplus \text{PH}(L,M') = \Delta \) holds is at most \( l \). Therefore, the almost-xor-universal advantage \( \epsilon_{\text{axu}} \) of \( \text{PH} \) is \( \ell/2^n \), where \( \ell \) is the maximum number of message blocks among all queries.

From Lemma 4, a simple corollary immediately follows:

**Corollary 1.** Let \( \text{fix}_b(\text{PH}) \) be the variant of the Polyhash function in which the least significant bit of the \( n \)-bit output of the function is fixed to bit \( b \). Then, \( \text{fix}_b(\text{PH}) \) is a \( 2l/2^n \)-regular, \( 2l/2^n \)-almost-xor universal and \( 2l/2^n \)-universal hash function.

We now define the Polyhash-based double-block hash function, \( \text{PH-DbH} \) function:

\[
\text{PH-DbH}_{(L_1,L_2)}(M) \triangleq \left( \text{fix}_0(\text{PH}_{L_1}(M), \text{fix}_1(\text{PH}_{L_1}(M)) \right).
\]  \hspace{1cm} (32)

Thus, two independent instances of the Polyhash function keyed with two independent keys \( L_1 \) and \( L_2 \) are applied separately to a message \( M \), and the least significant bit of their output is chopped and prepended with bits 0 and 1 respectively. The two-keyed PolyHash-based DbH construction can now be defined directly from the Two-Keyed DbH construction as follows: encrypt \( \text{fix}_0(\text{PH}_{L_1}(M)) \) and \( \text{fix}_1(\text{PH}_{L_1}(M)) \) through a block cipher \( E_K \) and xor the result together to produce the output. An algorithmic description of the construction is shown in Fig. 5.1.

\[
\begin{align*}
C_2[\text{PH-DbH}, E]_{(K_1,K_2,K)}(M) & \quad \text{PH}_L(M) \\
1: & \quad \Sigma = \text{fix}_0(\text{PH}_{K_1}(M)); & 1: & \quad M_1 \ldots M_t \xleftarrow{\$} M || 10^*; \\
2: & \quad \Theta = \text{fix}_1(\text{PH}_{K_2}(M)); & 2: & \quad Y = M_1 \cdot L^t \oplus M_2 \cdot L^{t-1} \oplus \cdots \oplus M_t \cdot L; \\
3: & \quad T = E_K(\Sigma) \oplus E_K(\Theta); & \text{return} \ Y; \\
\text{return} \ T;
\end{align*}
\]

*Figure 5.1:* The two-keyed Polyhash-based DbH construction \( C_2[\text{PH-DbH}, E] \) with \( \text{PH-DbH} \) as the underlying double-block hash function. \( M_1 || M_2 || \ldots || M_t \xleftarrow{\$} M || 10^* \) denotes the parsing of message \( M || 10^* \) into \( n \) bit strings.

\[\text{We write the degree of } f \text{ is at most } l \text{ because the coefficient of the leading term of the polynomial can be zero and in that case the polynomial will be of degree less than } l.\]
Clearly, the PH-DbH function is a good double-block hash function as the individual hash functions $H^1$ and $H^2$ are both $2\ell/2^n$-regular and universal. Furthermore, for a randomly chosen pair of keys $L_1, L_2$, and for any pair of messages $M, M' \in \{0, 1\}^*$,

$$\Pr[\text{fix}_0(PH_{L_1}(M)) = \text{fix}_1(PH_{L_2}(M'))] = 0.$$ 

Therefore, combining the Corollary 1 with Theorem 1, we derive the following security of the two-keyed PolyHash-based DbHtS construction $C_2[PH-DbH, E]$.

**Theorem 3.** Let $K$ be a non-empty finite set. Let $E: K \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ be an $n$-bit block cipher and $PH-DbH : (\{0, 1\}^n \times \{0, 1\}^n) \times \{0, 1\}^* \rightarrow (\{0, 1\}^n)^2$ be the PolyHash-based double-block hash function as defined above. Then any computationally unbounded distinguisher making a total of $q$ construction queries across all $u$ users such that each queried message is at most $\ell$ blocks long with $\ell \leq 2^{n-2}$ and a total of $p$ primitive queries to the block cipher $E$ can distinguish $C_2[PH-DbH, E]$ from an $n$-bit uniform random function with advantage

$$\text{Adv}_{\text{muPRF-ICM}}^{\text{Adv}_{C_2[PH-DbH, E]}}(u, q, p, \ell) \leq \frac{9q^{4/3}}{8 \cdot 2^n} + \frac{3q^{8/3}}{2 \cdot 2^n} + \frac{q^2}{2 \cdot 2^n} + \frac{9q^{7/3}}{8 \cdot 2^{4n}} + \frac{8q^4}{3 \cdot 2^{2n}} + \frac{q}{2^n} + \frac{2n^2}{2^{2n+k}} + \frac{4q\ell}{2^{n+k}} + \frac{4\ell^2}{2^{2n+k}} + \frac{8q^{1/3}}{2^n} + \frac{4\ell^2}{2^{2n+k}} + \frac{2pq}{2^{n+k}} + \frac{pq^{1/3}}{2^k}.$$ 

**Remark 5.** We would like to mention that the definition of the Polyhash function used in this paper is different from that used in [17]. Nevertheless, one can also establish the $3n/4$-bit multi-user security of the two-keyed PolyHash-based DbHtS construction with the Polyhash function used in [17].

### 6 Conclusion and Future Problems

In this paper, we have shown that the Two-Keyed DbHtS construction is multi-user secured up to $2^{3n/4}$ queries in the ideal-cipher model. As an instantiation of the result, we have shown that Polyhash-based DbHtS provides $3n/4$-bit multi-user security in the ideal-cipher model. Combining it with the generic result on the attack complexity of the DbHtS construction makes the bound tight. However, we cannot apply this result to analyze the security of 2K-SUM-ECBC, 2K-PMAC_Plus and 2K-LightMAC_Plus, as their underlying DbH functions are based on block ciphers, and our proof technique does not support their security analysis in the ideal-cipher model. This is because the underlying DbH function of these constructions is built on the top of block ciphers. We believe that proving $3n/4$-bit security of the DbHtS construction based on block cipher-based double-block hash functions needs a careful study.

**Road Block in the Analysis of Block Cipher Based DbH Function.** As mentioned earlier that analysing the security of DbHtS construction which is built on a block cipher based DbH function in the ideal-cipher model, one needs to assume that adversary can make primitive queries to the underlying block cipher used in the DbH function. As a result, one needs to bound the universal and the regular advantage of the underlying DbH function in the ideal-cipher model. The most non-trivial part of bounding the universal and the regular advantage of the underlying DbH function in the ideal-cipher model is when all the inputs to the block cipher are non-fresh, i.e., the adversary chooses message after all the primitive queries are done and the chosen messages are functions of the primitive query responses. We believe that this case is the major road block in establishing the
desired universal and regular bound of the DbH function in the ideal-cipher model. It will be interesting to see $O(\ell/2^n)$ bound of the universal and the regular advantage of a block cipher based DbH function in the ideal-cipher model.

References


