Exact Security Analysis of Hash-then-Mask Type Probabilistic MAC Constructions

Avijit Dutta, Ashwin Jha and Mridul Nandi

Applied Statistics Unit, Indian Statistical Institute, Kolkata.
avirocks.dutta13@gmail.com, ashwin.jha1991@gmail.com,
mridul.nandi@gmail.com

Abstract. Probabilistic MAC (message authentication code) is an alternative choice for a stateful MAC where maintaining internal state may be difficult or unsafe. Usually tag of a probabilistic MAC consists of an m-bit random coin (also called *salt*) and an n-bit core-tag depending on the salt. In terms of the security, probabilistic MAC falls under birthday collision of salts which is absent in stateful MAC. XMACR is an example of probabilistic MAC which remains secure up to $o(2^{m/2})$ tag generation queries. To achieve security beyond birthday in n, one can naturally use a large salt. For example, MACRX$_3$ sets $m = 3n$ and provides security up to $o(2^n)$ tag-generation queries. Large salt may restrict its applicability as it increases the cost of random string generation as well as the size of the overall tag. RWMAC (randomized version of WMAC) provides similar security with $m = n$ but it uses a PRF (pseudorandom function) over $2n$-bit inputs which is naturally more costlier than those over n-bit inputs. Achieving beyond birthday security using n-bit PRF and n-bit salt is a practical and challenging problem. Minematsu in FSE 2010 proposed Enhanced Hash-then-Mask (EHtM) using n-bit salt and showed its security up to $o(2^{2n/3})$ tag-generation queries. In this paper we revisit this construction and we provide exact security analysis of EHtM. In particular, we show that it has higher security, namely up to $o(2^{3n/4})$ queries, than what claimed by the designer. Moreover, we demonstrate a single attempt forgery attack which makes about $2^{3n/4}$ tag generation queries. XMACR and EHtM follow the hash-then-mask paradigm due to Carter-Wegman. We revisit six possible constructions following hash-then-mask paradigm and we provide exact security analysis for all of these constructions, some of which however were known before.

Keywords: MAC, EHtM, XOR-MAC, Pseudorandom function.

1 Introduction

Nowadays, it is desirable that every transmitted message or packet will use cryptographic means to ensure authenticity. As a solution of this, a MAC (message authentication code) enables two parties sharing a key to authenticate their transmissions. It is very popular in symmetric key cryptography. PRF [12] (pseudorandom function) is also an essential tool in many cryptographic solutions.

They can also be used to generate a pseudorandom pad for symmetric encryption, and to mask a universal hash function for producing a secure *hash-then-mask* [11] types of MAC. The security of PRF based constructions can be compromised if one applies the PRF twice to the same input. A natural way to avoid repetition is for the sender to use an increasing counter, or other form of varying, non-repeating state (also called *nonce*), which is updated with each application of the function. This method has been adopted to XMACC [4], WMAC [8] etc. However, this can have various drawbacks, especially it involves management of nonce which might in some settings be impractical or unsafe. This can happen, for example, whenever maintaining a synchronized nonce across different applications of the function, which is unsafe or impossible. A possibility of having a stateless scheme is to use random values (also called **salt**) as those on which to evaluate the pseudorandom function. In this paper our main focus is to analyze the security of some of the probabilistic MAC candidates.

A MAC is defined by a pair of algorithms, tag generation and verification algorithm. The verification algorithm must verify any tag generated by the tag generation algorithm. Usually tag of a probabilistic MAC consists of an m-bit random coin (also called *salt*) and an n-bit **core-tag** depending on the salt. A forgery algorithm makes queries to both algorithms. We say that it forges if it can submit a non-trivial message tag pair to the verification algorithm which verifies the queries. By non-trivial we mean that it should not be obtained through a tag generation query. Informally speaking, a MAC is said to have q-security if for any forgery making up to $o(q)$ queries (tag-generation), it can forge with probability at most $q_v/2^n + o(1)$ where q_v is the number of verification queries and n is the size of the core-tag. Note that $o(1)$ is a very small quantity and goes to 0 as n increases. Also note that $q_v/2^n$ forging probability can always be achieved by making random q_v forgery attempts.

A Brief History on Probabilistic Stateless MAC.

XMACR [4], EHtM [18] are some known examples of probabilistic MAC which follow hash-then-mask paradigm due to Carter-Wegman [8]. The core-tag of XMACR [4] is computed by masking hash output by the output of a pseudorandom function applied to the salt. More formally,

$$\mathsf{HtM}_{f,H}(x) = (r, H(x) \oplus f(r))$$

where r is the salt for this construction, H is an AXU hash [24] and f is an n-bit pseudorandom function. The hash function of XOR-MAC is a parallel construction of counter based AXU-hash. This construction has $o(2^{m/2})$ unforgeable security for an m-bit salt due to the birthday phenomenon that the salt can also repeat. Some natural choices are known to obtain beyond birthday security or amplify the security. A trivial solution is to use a larger sized salt, e.g. $m = 2n$. However, this has the following drawbacks.

1. It uses larger salt forcing increased communication cost as well as sender's effort for generating randomness, and
2. it needs $2n$-bit-input PRF instead of n-bit-input PRF.

The second issue has been resolved by the construction MACRX$_3$ [3] which still uses salt of size $m = 3n$ to compute the mask based on n-bit pseudorandom function. Whereas RWMAC [18], a randomized version of nonce based WMAC can provide $o(2^n)$ security using n-bit salt. However, it uses pseudorandom function mapping from $2n$ bits to n bits. As a solution to both of the problems, RMAC [14] and FRMAC [15] are known which provide $o(2^n)$ security. However, their security proofs are based on ideal assumption on the underlying primitive [[14], [15]]. Minematsu [18] proposed a simple variant of hash-then-mask (called EHtM or **Enhanced Hash-then-Mask**) and showed its security for $o(2^{2n/3})$ tag generation queries [18]. It uses only n-bit salt and n-bit PRF. Thus, an appropriate usage of salt can enhance the security to go beyond birthday barrier without requiring larger domain PRF.

Different Types of Hash-then-MAC or HtM. In this paper, we consider six possibilities of hash-then-mask types constructions as shown in Fig. 1.1. We recall that C2 is the one which was introduced by Carter and Wegman and also adopted in many constructions like XOR-MAC [4], poly1305 [7] etc. with a specific choice of hash function H and the pseudorandom function f. Let r be n-bit salt and m denote the message. We note that EHtM is same as C6.

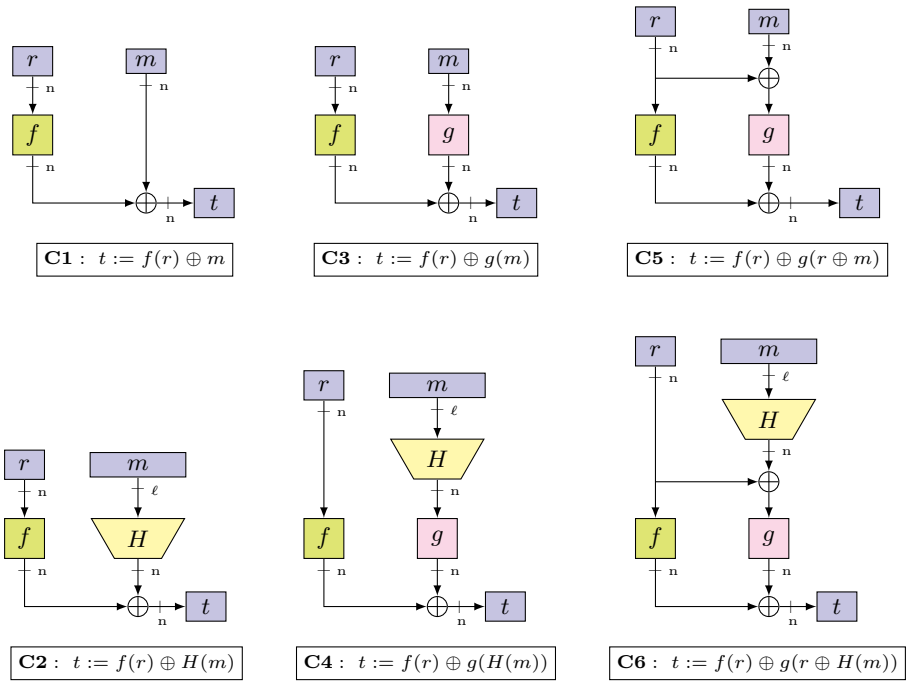

Fig. 1.1: Probabilistic MAC Schemes. $\ell, n \in \mathbb{N}$ and $\ell \geq n$, denote the input and output size, respectively. We simply denote the size by n when $\ell = n$. Blocks with labels f and g represent two independent n-bit to n-bit PRF, and blocks with label H represent an n-bit universal hash. Schemes C1, C3, and C5 take fixed length n bit message m and n bit salt r as input, and return n bit tag t as output. Schemes C2, C4 and C6 take an arbitrary length (depending on H) ℓ bit message m and a fixed length n bit salt as input, and return n bit tag as output.

1.1 Our Contributions

The main contribution of the paper is to show the tight bounds for all of these constructions as shown in Fig. 1.1. Some of the bounds are already known and rest of the bounds are proved in this paper. Most importantly the security of EHtM [18] is shown to have higher security, namely $2^{3n/4}$, compared to what known before. In addition to the MAC security of these constructions, we also study pseudorandom function or PRF property of the core-tag generation algorithm. Like deterministic MAC, bounding PRF advantage of the core-tag generation would essentially bound the forging probability. However, the PRF might be very stronger assumption in probabilistic MAC as the adversary can control the random coin in PRF game. We see that all constructions except C6 are not PRF. Moreover, the PRF bound for C6 is also much less compared to what we have in case of forging. Moreover, we introduce a new security notion, called **pPRF or probabilistic PRF**, which is a weaker version of PRF and may be more appropriate in this probabilistic setting. Definition of pPRF can be found in Sect. 2. Here adversary has no control on the random coin but can observe once he makes a query. This notion is somewhere in-between PRF (in which adversary has control on both message and random coin) and weak-PRF [20] (in which adversary has no control on both). The following table provides the exact security analysis of all six constructions in terms of the PRF, pPRF and forging or MAC security.

	C1	**C2**	**C3**	**C4**	**C5**	**C6**
PRF	$\Theta(1)^*$	$\Theta(1)^*$	$\Theta(1)^*$	$\Theta(1)^*$	$\Theta(1)^*$	$\Theta(2^{n/2})^*$
pPRF	$\Theta(2^{n/2})$	$\Theta(2^{n/2})$	$\Theta(2^{n/2})$	$\Theta(2^{n/2})$	$\Theta(2^{3n/4})$	$\Theta(2^{3n/4})$
MAC	$\Theta(1)^*$	$\Theta(2^{n/2})^*$	$\Theta(2^{n/2})^*$	$\Theta(2^{n/2})^*$	$\Theta(2^{2n/3})$	$\Theta(2^{3n/4})$

Table 1: Summary of security bounds for probabilistic MAC candidates following hash-then-mask paradigm. The columns correspond to the probabilistic MAC candidates illustrated in figure 1.1. All values are in terms of optimal number of adversial tag generation queries. The specified results appear as direct consequence of results discussed in Sect. 6 and Sect. 7. By * we mean the results were known or it is not difficult to observe.

As a side result, we have also shown that pPRF is used to prove an impossibility result that unlike random function in deterministic MAC, there is no idealized

version of unforgeable security of a probabilistic MAC. In fact, pPRF does not imply secure probabilistic MAC in general. The intuitive reason is that, one can have few number of random coins at which the core-tag generation algorithm becomes completely forgeable. In a distinguishing game, observing those weak random coins from tag generation queries will have negligible probability. We discuss this issue in Sect. 4. Moreover, we show that for some constructions (e.g C5) pPRF advantage (i.e $\Theta(2^{3n/4})$) is better than its corresponding MAC advantage (i.e $\Theta(2^{2n/3})$).

2 Basic Definitions

2.1 Notation

Let $\perp$ and $\top$ be two special symbols meaning reject and accept respectively. We define $x \stackrel{?}{=} t$ to be $\top$ if $x = t$, otherwise it is defined to be $\perp$. For a set $\mathcal{X}$, $\mathbf{X} \xleftarrow{\$} \mathcal{X}$ means that $\mathbf{X}$ is chosen uniformly from the set $\mathcal{X}$ and it is independent to all random variables defined so far.

2.2 (Almost-XOR) Universal Hash Functions

An n-bit hash function H is a $(\mathcal{K}, \mathcal{D})$-family of functions $\{H_k := H(k, \cdot) : \mathcal{D} \to \{0,1\}^n\}_{k \in \mathcal{K}}$ defined on its domain or message space $\mathcal{D}$ and indexed by the **key space** $\mathcal{K}$.

Definition 1 (ϵ-AXU hash function [24]). *A $(\mathcal{K}, \mathcal{D})$-family H is called* **ϵ-Almost-XOR Universal** *(or AXU, in short) hash function, if for any two distinct x and x' in $\mathcal{D}$ and a $\delta \in \{0,1\}^n$, the* δ-differential probability

$$\mathit{diff}_{H,\delta}[x, x'] := \mathit{Pr}_K[H_K(x) \oplus H_K(x') = \delta] \leq \epsilon$$

where the random variable K is uniformly distributed over the set $\mathcal{K}$.

Unless mentioned explicitly, we always mean key K to be chosen uniformly from its key space $\mathcal{K}$. The *maximum δ-differential probability* over all possible of two distinct inputs x, x' is denoted by $\Delta_{H,\delta}$. The *maximum differential probability* $\Delta_H := \max_\delta \Delta_{H,\delta}$. If $\Delta_H \leq \epsilon$ then we call the hash function to be a ϵ-AXU Hash function. Multi-linear hash [13, 25], pseudo-dot-product or PDP hash [28, 13, 9, 16], PHASH [19] etc. are some examples of AXU hash functions.

Universal Hash Function. When $\delta = 0$, the 0-differential event is equivalent to collision. So we write $\mathsf{diff}_{H,0}[x, x']$ and $\Delta_{H,0}$ by $\mathsf{coll}_H[x, x']$ and coll_H respectively and we call them collision probabilities.

Definition 2 (ϵ-universal hash function). *A hash family H is called* **ϵ-universal** *(or ϵ-U) if $\mathit{coll}_H := \max_{x \neq x'} \mathit{Pr}_K[H_K(x) = H_K(x')] \leq \epsilon$.*

2.3 Message Authentication Code or MAC

Informally a message authentication code or MAC allows parties to share a secret common key k (chosen uniformly from the key space) to authenticate the data they send to each other. MAC in principle works as follows: The sender applies a tag generation algorithm TG to key k and a message m to generate a tag t, and send (m,t) to the receiver. Bob upon receiving the message-tag pair (m,t) applies verification algorithm VF to the key k and the received (m,t) pair. Verification algorithm returns $\top$ or $\bot$ to indicate whether the received message-tag pair is considered to be authentic or not. Formally, we define MAC as follows.

Definition 3 (MAC or Message Authentication Code). *A MAC scheme Π defined over a message space $\mathcal{M}$, a key space $\mathcal{K}$ and a tag space $\mathcal{T}$ is a pair $(\mathsf{TG},\mathsf{VF})$ of algorithms.*

1. Tag generation algorithm*: TG is a (possibly probabilistic) algorithm from $\mathcal{K}\times\mathcal{M}$ to $\mathcal{T}$. A pair (m,t) is called valid for a key k if $\mathsf{Pr}[\mathsf{TG}(k,m)=t]>0$, otherwise it is said to be invalid.*[1]
2. Verification algorithm*: VF is a deterministic algorithm from $\mathcal{K}\times\mathcal{M}\times\mathcal{T}$ to $\{\top,\bot\}$ such that for all valid pair (m,t) for a key k, $\mathsf{VF}(k,m,t)=\top$.*

We say that a MAC is complete if it satisfies the *completeness condition* - for all invalid pair (m,t) for k, $\mathsf{VF}(k,m,t)=\bot$. A MAC algorithm is called deterministic if its tag generation algorithm is deterministic. In this case, one can always define a complete MAC by defining verification algorithm as

$$\mathsf{TG}(k,m)\stackrel{?}{=}t.$$

A probabilistic MAC, on the other hand, returns a probability distribution on the tag space $\mathcal{T}$ for every pair $(k,m)\in\mathcal{K}\times\mathcal{M}$. To distinguish it from a deterministic MAC, we sometimes denote a probabilistic MAC as a pair $\Pi^{\$}=(\mathsf{TG}^{\$},\mathsf{VF})$. A probabilistic algorithm chooses a random coin R uniformly from a coin space $\mathcal{R}$ and then apply some deterministic algorithm. We also write a probabilistic tag-generation algorithm, abusing notation, as $\mathsf{TG}^{\$}(k,m;R)$ where $\mathsf{TG}^{\$}(\cdot,\cdot;\cdot)$ is the underlying deterministic algorithm of $\mathsf{TG}^{\$}(\cdot,\cdot)$.

We say that a probabilistic MAC is *coin-explicit MAC* if there is a deterministic algorithm, called *core-tag generation*, $\mathsf{cTG}:\mathcal{K}\times\mathcal{M}\times\mathcal{R}\to\mathcal{T}'$ such that

$$\mathsf{TG}^{\$}(k,m;r)=(r,\mathsf{cTG}(k,m,r)).$$

Similar to a deterministic MAC, one can define a verification algorithm appropriately to convert a coin-explicit MAC into a complete MAC. For example, a verification algorithm on input $(k,m,(r,t'))$, returns $(\mathsf{cTG}(k,m,r)\stackrel{?}{=}t')$.

Security Definitions of MAC. We define two types of security notions - (a)

[1] Here the probability is computed under randomness, if any, of the tag-generation algorithm. In case of deterministic algorithm, we can ignore probability and simply write it as $\mathsf{TG}(k,m)=t$.

weak unforgeable or UF and (b) **strong unforgeable** or SUF. To define these security notions, let us first define an experiment $\mathbf{Exp}^{\mathsf{type}}_{\mathcal{A},\Pi}$ where $\mathsf{type} \in \{\mathrm{UF}, \mathrm{SUF}\}$. The experiment basically runs an adversary $\mathcal{A}$ interacting with the tag generation $\mathsf{TG}_k(\cdot) = \mathsf{TG}(k,\cdot)$ and verification oracle $\mathsf{VF}_k(\cdot,\cdot) = \mathsf{VF}(k,\cdot,\cdot)$ where k is sampled uniformly from the key space in the very beginning of the experiment and remains fixed throughout. We say that a verification query (m,t) is trivial if t is obtained in a previous tag generation oracle with m as a query. In this case, clearly verification oracle returns $\top$. In case of a deterministic MAC, a tag generation query m is trivial if it is queried before. Conventionally, we assume that **all adversaries in this paper make no trivial queries**. When $\mathsf{type} = \mathrm{UF}$, the adversary is not allowed to make verification query (m,t) where m has been queried to the tag generation oracle. However, this constraint is not present for SUF experiment. Finally, the experiment returns 1 if $\mathcal{A}$ obtains $\top$ from any one of the verification oracle call, otherwise it returns zero. The advantage of $\mathcal{A}$ in forging the MAC Π is defined as

$$\mathbf{Adv}^{\mathsf{type}}_{\Pi}(\mathcal{A}) := \Pr[\mathbf{Exp}^{\mathsf{type}}_{\Pi,\mathcal{A}} = 1].$$

The maximum advantage of forging Π is defined as

$$\mathbf{Adv}^{\mathsf{type}}_{\Pi}(q_m, q_v, t) := \max_{\mathcal{A}} \mathbf{Adv}^{\mathsf{type}}_{\Pi}(\mathcal{A})$$

where maximum is taken over all $\mathcal{A}$ which makes at most q_m many tag generation oracle and q_v many verification oracle queries and runs in time at most time t. A MAC algorithm Π is called (ϵ, q_m, q_v, t)-type MAC if $\mathbf{Adv}^{\mathsf{type}}_{\Pi}(q_m, q_v, t) \leq \epsilon$. For an unbounded adversary, we may skip the time parameter t.

Experiment $\mathbf{Exp}^{\mathsf{type}}_{\mathcal{A},\Pi}$

1. **initialize** $k \xleftarrow{\$} \mathcal{K}$; $f \leftarrow 0$; list $L = \emptyset$;
2. run $\mathcal{A}$;
   -(responses of queries described below)
3. **return** f;

On a non-trivial verification query (M,T)

1. $b \leftarrow \mathsf{VF}(M,T)$;
2. **if** $b = \top$ **then**
   if $\mathsf{type} = \mathrm{SUF}$ or $M \notin L$
   then $f = 1$;
3. **return** b;

On tag-generation query M

1. $T \xleftarrow{\$} \mathsf{TG}(M)$;
2. $L = L \cup \{M\}$;
3. **return** T;

Fig. 2.1: The experiment describes interaction between $\mathcal{A}$ and a MAC Π.

2.4 Pseudo-Random Function or PRF and its Variants

Let $F := \{f_k : k \in \mathcal{K}\}$ be a finite function family in which each function is indexed by a key $k \in \mathcal{K}$ and $\forall k \in \mathcal{K}$, $f_k : X \to Y$. By $f \xleftarrow{\$} F$, we mean $f = f_k$ where the key $k \xleftarrow{\$} \mathcal{K}$. We write $\mathsf{Func}[X, Y]$ to denote the set of all functions from a finite set X to a finite set Y. A function RF is said to be a *random function* if RF is sampled uniformly from the set $\mathsf{Func}[X, Y]$.

Distinguishing Game. Consider two classes of functions $\mathcal{C}_0$ and $\mathcal{C}_1$. In a **simple distinguishing game**, the oracles $\mathcal{O}_0$ and $\mathcal{O}_1$ behave as follows: for $b \in \{0, 1\}$, $\mathcal{O}_b$ samples $f_b \xleftarrow{\$} \mathcal{C}_b$ and simulates it. The challenger chooses a $b \in \{0, 1\}$, and the adversary $\mathcal{A}$ is allowed to make q queries $x^1, \ldots, x^q$ to $\mathcal{O}_b$, which simply returns $f_b(x^1), \ldots, f_b(x^q)$, respectively. Note that $\mathcal{A}$ is allowed to make these queries adaptively, i.e., for $2 \leq i \leq q$, he can choose x^i after observing $f_b(x^1), \ldots, f_b(x^{i-1})$. $\mathcal{A}$ finally returns a $b' \in \{0, 1\}$, and wins if $b' = b$. In this paper, we'll only be concerned with simple distinguishing games. Conventionally, $\mathcal{O}_0$ imitates a random function RF, and is called the ideal oracle, and $\mathcal{O}_1$ imitates a certain construction, and is called the real oracle.

For an adversary $\mathcal{A}$ interacting with oracles either $\mathcal{O}_0$ or $\mathcal{O}_1$, we define the **distinguishing advantage** of $\mathcal{A}$ as

$$\mathbf{Adv}^{\mathcal{O}_0}_{\mathcal{O}_1}(\mathcal{A}) := |\Pr[\mathcal{A}^{\mathcal{O}_0} \text{ returns } 1] - \Pr[\mathcal{A}^{\mathcal{O}_1} \text{ returns } 1]|.$$

The above definition of advantages can be similarly extended for two or more oracles.

PRF Advantage. Given an oracle adversary $\mathcal{A}$, we define prf-advantage of $\mathcal{A}$ against a keyed function F_K as

$$\mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{A}) = \mathbf{Adv}^{\mathsf{RF}}_{F}(\mathcal{A}) = |\Pr[\mathcal{A}^{F_K} = 1] - \Pr[\mathcal{A}^{\mathsf{RF}} = 1]|.$$

Let $\mathbf{Adv}^{\mathrm{prf}}_{F}(q, t)$ denote $\max_{\mathcal{A}} \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{A})$ where maximum is taken over all adversaries $\mathcal{A}$ running in time t, making at most q queries.

Weak-PRF [20]. Let F be a finite function family and $\mathsf{Func}[X, Y]$ be the set of all functions from X to Y. Given an oracle adversary $\mathcal{A}$ that cannot choose and submit the domain point of the oracle but pings the oracle. Real oracle will choose the domain point x at random and evaluates the function at that point and return it to the adversary. We define weak-prf or wprf advantage of $\mathcal{A}$ against a keyed function F_K as $\mathbf{Adv}^{\mathrm{wprf}}_{F}(\mathcal{A}) = |\Pr[\mathcal{A}^{F_K} = 1] - \Pr[\mathcal{A}^{\mathsf{RF}} = 1]|$. Let $\mathbf{Adv}^{\mathrm{wprf}}_{F}(t, q)$ denote $\max_{\mathcal{A}} \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{A})$ where maximum is taken over all adversaries $\mathcal{A}$ running in time t, making query at most q.

Weak-prf plays important role defining one-time padding based encryption. Given that F_K is a weak-prf, one can define an encryption of a message m as $(r, F_K(r) \oplus m)$. The same idea has been also used for computing hash-then-mask based probabilistic MAC.

Probabilistic PRF (pPRF). In this paper we consider a little bit stronger

version of weak PRF in which adversary has control on a part of the input. Let F_K be a keyed function which has a pair of inputs from $\mathcal{R} \times X$ and returns an element from Y. Informally, the first input represents the random coin whereas the second one is some message or message-dependent part. So we allow adversary to choose only the second input while making queries to this keyed function. More precisely, let \$ denote a random source that on every query, returns a randomly chosen element from $\mathcal{R}$. Let $F_K^{\$}(m) = (R, F_K(R, m))$ where R is chosen randomly from $\mathcal{R}$. The corresponding ideal oracle is $\mathsf{RF}^{\$} := \$\|\mathsf{RF}$ which on query m returns $(R, \mathsf{RF}(R, m))$ where RF is the random function from $\mathcal{R} \times X$ to Y. Then we define the pPRF-advantage of an adversary $\mathcal{A}$ as

$$\mathbf{Adv}^{\mathrm{pPRF}}_{F^{\$}}(\mathcal{A}) := \mathbf{Adv}^{\mathsf{RF}^{\$}}_{F^{\$}}(\mathcal{A}).$$

Let $\mathbf{Adv}^{\mathrm{pPRF}}_{F^{\$}}(q, t)$ denote $\max_{\mathcal{A}} \mathbf{Adv}^{\mathrm{pPRF}}_{F^{\$}}(\mathcal{A})$ where maximum is taken over all adversaries $\mathcal{A}$ running in time t, making at most q queries.

2.5 Coefficients H Technique

In this section we briefly discuss coefficients H technique [23] due to Patarin. It is also known as Decorrelation Theorem due to Vaudenay [27]. Suppose an adversary $\mathcal{A}$ interacts with one of the two oracles - real oracle $\mathcal{O}_{\mathrm{re}}$ and ideal oracle $\mathcal{O}_{\mathrm{id}}$. For notational simplicity we write X_{re} (resp. X_{id}) to denote the random variable representing real transcript $\tau_{\mathcal{A}}^{\mathcal{O}_{\mathrm{re}}}$ and ideal transcript $\tau_{\mathcal{A}}^{\mathcal{O}_{\mathrm{id}}}$ respectively whenever the adversary as well as the real and ideal oracles are well understood. A transcript τ is said to be *attainable w.r.t. ideal* if the probability of realizing τ is positive in the ideal world i.e., $\Pr[X_{\mathrm{id}} = \tau] > 0$. Let Θ be the set of all attainable transcripts w.r.t. ideal. Following these notations we state coefficients H technique.

Theorem 1 (Coefficients H Technique). *Let $\Theta = \Theta_{\mathrm{good}} \sqcup \Theta_{\mathrm{bad}}$ (disjoint union) be some partition of the set of attainable transcripts. Suppose there exists $\epsilon_{\mathrm{ratio}} \geq 0$ such that for any $\tau \in \Theta_{\mathrm{good}}$,*

$$\frac{\Pr[X_{\mathrm{re}} = \tau]}{\Pr[X_{\mathrm{id}} = \tau]} \geq 1 - \epsilon_{\mathrm{ratio}},$$

and there exists $\epsilon_{\mathrm{bad}} \geq 0$ such that $\Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}] \leq \epsilon_{\mathrm{bad}}$. Then,

$$\mathbf{Adv}^{\mathcal{O}_{\mathrm{id}}}_{\mathcal{O}_{\mathrm{re}}}(\mathcal{A}) \leq \epsilon_{\mathrm{ratio}} + \epsilon_{\mathrm{bad}}. \tag{1}$$

When $\mathcal{O}_{\mathrm{id}}$ is the random function RF_X and $\mathcal{O}_{\mathrm{re}}$ is some keyed construction in our interest defined over the domain X then the above Eq. 1 says that $\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{O}_{\mathrm{re}}}(\mathcal{A}) \leq \epsilon_{\mathrm{ratio}} + \epsilon_{\mathrm{bad}}$.

For any fixed transcript τ, we also say the probabilities $\Pr[X_{\mathrm{re}} = \tau]$ and $\Pr[X_{\mathrm{id}} = \tau]$ are real and ideal **interpolation probabilities** respectively. The proof of the

theorem can be found in [23] and so we skip the proof. To apply Coefficients H technique,

(A) we need to identify a set of bad transcripts Θ_{bad} which is realized with negligible probability while interacting with ideal oracle, and

(B) we have to show that for any fixed good (attainable) transcript τ, the ratio of real and ideal interpolation probabilities is at least a number which is very close one.

2.6 Result on the connection between Unforgeability and Pseudorandomness

We first describe some tools which would be used to prove the rest of results. We first show how to bound SUF advantage in terms of distinguishing advantages of two pairs of oracles. By abusing notation, let $\perp$ be an oracle which returns $\perp$ on every verification query. Note that we assume all adversaries make only nontrivial queries. We recall that $\mathsf{RF}^{\$}$ returns $(R, \mathsf{RF}(R, m))$ on a query m where R is chosen randomly on every query (i.e. it is a salt).

Lemma 1. *Let $\Pi^{\$} = (\mathsf{TG}_k^{\$}, \mathsf{VF}_k)$ a probabilistic MAC. Then,*

$$\mathbf{Adv}^{\text{SUF}}_{\Pi^{\$}}(q_m, q_v, t) \leq \mathbf{Adv}^{\mathsf{RF}^{\$},\perp}_{\mathsf{TG}^{\$}_{\mathbf{K}},\mathsf{VF}_{\mathbf{K}}}(q_m, q_v, t).$$

Proof. Let $\mathcal{A}$ be a forgery which makes q_m and q_v queries to left and right oracle respectively. Here the left oracle is the tag generation oracle and the right oracle is the verification oracle. Moreover, it runs in time t. Suppose the SUF-advantage of $\mathcal{A}$ in forging the MAC $\Pi^{\$}$ is at least ϵ. We construct the oracle algorithm $\mathcal{B}$, having access to a pair of oracles. It first runs $\mathcal{A}$. The responses of $\mathcal{A}$ is computed by forwarding its queries to $\mathcal{B}$'s oracles. At the end of the game $\mathcal{B}$ returns a bit b' where b' is 1 if $\exists 1 \leq i \leq q_v$ such that $b_i = \top$, else $b' = 0$ and b_i denotes the response of ith verification query. Having defined the interactive game, we now calculate the distinguishing advantage of $\mathcal{B}$ as follows

$$\begin{aligned}\mathbf{Adv}^{\mathsf{RF}^{\$},\perp}_{\mathsf{TG}^{\$}_{\mathbf{K}},\mathsf{VF}_{\mathbf{K}}}(q_m, q_v, t) &= \mathsf{Pr}[\mathcal{A}^{\mathsf{TG},\mathsf{VF}} = 1] - \mathsf{Pr}[\mathcal{A}^{\mathsf{RF}^{\$},\perp} = 1] \\ &= \mathbf{Adv}^{\mathsf{mac}}(\mathcal{A}) - 0 \\ &\geq \epsilon\end{aligned}$$

Note that, $\mathsf{Pr}[\mathcal{A}^{\mathsf{RF}^{\$},\perp} = 1] = 0$ as the ideal oracle does always return $\perp$. The result follows by taking maximum over all adversaries $\mathcal{A}$. □

We would like to note that this result does not contradict our impossibility result described in Sect. 4. Due to this lemma, it is sufficient to bound the distinguishing advantage of distinguishing two random system i.e $(\mathsf{TG}^{\$}_k, \mathsf{VF}_k)$ and $(\mathsf{RF}^{\$}, \perp)$ to derive the bound on the advantage of the probabilistic MAC $\Pi^{\$}$. To bound the distinguishing advantage we use **Coefficients H technique** [21].

2.7 Results on Alternating Cycle

In this section we discuss some useful lemmas which will be used to bound the PRF/pPRF/MAC advantage of six constructions that we have shown in Sect. 5.

Definition 4 ([1], [22]). *Let us consider a sequence of pair of values τ from the set $\{0,1\}^n$ where*

$$\tau := \{(x_1, y_1), (x_2, y_2), \ldots, (x_q, y_q)\},$$

such that each $x_i, y_i \in \{0,1\}^n$. We will say that we have an alternating-cycle in τ of length k *where $k \geq 2$ and k is an even integer, if we have k pairwise distinct indices $i_1, i_2, \ldots, i_k$ such that $x_{i_1} = x_{i_2}, y_{i_2} = y_{i_3}, x_{i_3} = x_{i_4}, ..., x_{i_{k1}} = x_{i_k}, y_{i_k} = y_{i_1}$.*
We will say that we have an alternating-line in τ of length k *if we have $k+1$ pairwise distinct indices such that $x_{i_1} = x_{i_2}, y_{i_2} = y_{i_3}, x_{i_3} = x_{i_4}, \ldots, y_{i_{k1}} = y_{i_k}, x_{i_k} = x_{i_{k+1}}$, when k is odd. If k is even, then the alternating-line of length k exists if there exists $k+1$ pairwise distinct indices such that $x_{i_1} = x_{i_2}, y_{i_2} = y_{i_3}, x_{i_3} = x_{i_4}, \ldots, x_{i_{k1}} = x_{i_k}, y_{i_k} = y_{i_{k+1}}$*

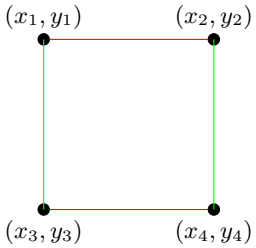

Fig. 2.2: Alternating Cycle of length 4. Red line indicates first coordinate matches. Green line indicates second coordinates matches

Lemma 2 ([22]). *Let f and g be two n-bit independent and uniformly distributed random functions. Let us consider a transcript $\tau = \{(x_i, y_i, t_i)_{1 \leq i \leq q}\}$ which does not contain any alternating cycle. Then*

$$\Pr[f(x_i) \oplus g(y_i) = t_i, 1 \leq i \leq q] = \frac{1}{2^{nq}}.$$

The proof of this can be found in [22]. As a corollary of the above lemma we have the following result.

Lemma 3. *Let $\mathrm{SUM}_{\mathrm{f,g}} := f(x) \oplus g(y)$ then for any adversary $\mathcal{A}$ that queries the oracle $\mathrm{SUM}_{\mathrm{f,g}}$ such that the queries to the oracle do not form an alternating cycle (we call such an adversary NAC-restricted adversary), we have*

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{SUM}}(\mathcal{A}) = 0.$$

The above lemma says that if C is a construction that uses the SUM function, then any adversary $\mathcal{A}$ that wants to break the prf-security of C, then we have

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{C}} \leq \Pr[\mathsf{Alt\text{-}Cycle\ in\ the\ input\ to\ SUM\ function}].$$

We shall use this implicitly later in this section, to bound the pPRF and MAC advantage of constructions C5 and C6.

3 Paradigm of Constructing a Secure MAC

In this section, we briefly recall some of the well known paradigm for constructing a secure MAC. The first basic result concerning to construct a secure MAC is that a *secure PRF is a secure deterministic MAC*. It has been shown [5], if F_k is a secure PRF, then one can use F_k as a secure MAC. Let $F : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ be a ϵ-PRF. Then we define a deterministic MAC $\mathit{\Pi}_F = (\mathsf{TG} = F, \mathsf{VF})$ (where VF is defined naturally to make it complete MAC as described before). It is shown in [5] that $\mathit{\Pi}_F$ is also MAC. More formally, we have the following result.

$$\mathbf{Adv}^{\mathrm{SUF}}_{\mathit{\Pi}_F}(q_m, q_v, t) \leq \mathbf{Adv}^{\mathrm{prf}}_{F}(q_m + q_v, t') + \frac{q_v}{\mathcal{T}}, \quad t' \approx t. \tag{2}$$

From the above lemma, if a construction F_k is proven to be a secure PRF then $\mathit{\Pi}_F$ is also a MAC. This is true only for a deterministic MAC. In this regard, we would like to mention here that for a probabilistic MAC we do not know any such example of ideal object such that the distinguishing advantage can be used to bound the MAC advantage. In the following section, we will show an impossibility result showing that no such ideal system exists so that indistinguishable to this would imply secure probabilistic MAC.

Composition Result. Composition Result is one of the approaches to construct a variable input length PRF from a fixed input length PRF and a universal hash function as shown in [26]. More formally,

Theorem 2 ([26]). *Let $G_{K_1,K_2} := F_{K_2} \circ H_{K_1} : \mathcal{D} \to \{0,1\}^n$ where H is an m-bit ϵ-universal hash defined over $\mathcal{D}$ and F be a keyed function family from $\{0,1\}^m$ to $\{0,1\}^n$. Then,*

$$\mathbf{Adv}^{\mathrm{prf}}_{G}(t, q) \leq \mathbf{Adv}^{\mathrm{prf}}_{F}(t', q) + \binom{q}{2} \times \epsilon,$$

where $t' = t + \mathcal{O}(qT_h)$ and T_h denotes the maximum time for computing H.

Hash-then-Mask Result. A common approach for building an IV based MAC is due to Carter and Wegman [11] construction called **Hash-then-Mask**. It uses an ϵ-AXU hash function $H : \{0,1\}^* \to \{0,1\}^n$ and a n-bit pseudo random function $F : \{0,1\}^n \to \{0,1\}^n$.

Theorem 3 ([18]). *Let $G_{K_1,K_2} := F_{K_2} \oplus H_{K_1} : \mathcal{IV} \times \{0,1\}^* \to \{0,1\}^n$ which is defined as $G_{K_1,K_2}(IV, m) := F_{K_2}(IV) \oplus H_{K_1}(m)$. We denote G_{K_1,K_2} as $G^{\$}_{K_1,K_2}$ when IV is chosen uniformly at random from a random source. Similarly, we denote G_{K_1,K_2} as $G^{\text{ctr}}_{K_1,K_2}$ when IV is a nonce or counter. Then we have the following*

1. $\mathbf{Adv}^{\text{mac}}_{G^{\$}_{K_1,K_2}}(q_m, q_v) \leq \mathbf{Adv}^{\text{prf}}_{F_{K_2}}(q_m + q_v) + \frac{q_m^2}{2^n} + \frac{q_v}{2^n}$
2. $\mathbf{Adv}^{\text{mac}}_{G^{\text{ctr}}_{K_1,K_2}}(q_m, q_v) \leq \mathbf{Adv}^{\text{prf}}_{F_{K_2}}(q_m + q_v) + \frac{q_v}{2^n}$

3.1 Categories of MAC

We have seen in the last section that a MAC could be deterministic or probabilistic depending on whether the tag generation algorithm being deterministic or probabilistic. Apart from these two categories of MAC, we have a *stateful-MAC* in which the signer maintains a state across each consecutive signing requests. The main reason of introducing state is purely for achieving higher security from efficient constructions. The internal state is maintained in such a manner so that the inputs of underlying ideal object are not getting repeated. Probabilistic MAC also tries to do same, however, at the cost of some probability the inputs can remain distinct. Here are some examples of different MAC categorized according to the presence of random salt or counter.

1. Stateless Deterministic MAC (e.g, CBC-MAC [5], PMAC [10], HMAC [2], LightMAC [17], PCS-MAC [6] etc.).
2. Stateless and Probabilistic MAC (e.g, MACRX$_t$[3], XMACR [4], EhtM [18], RWMAC [18], RMAC [14], FRMAC [15] etc.).
3. Stateful and Deterministic MAC (e.g XMACC [4], WMAC [8] etc.).

There can be one more possibility, namely stateful and probabilistic MAC. However, this is not useful in practice as it costs both the randomness and internal state.

3.2 Some Known Constructions of Probabilistic MAC

XOR-MAC. XOR-MAC [4] is a simple, parallelizable and incremental MAC scheme, proposed by Bellare et al, which is proven to be more secure than popularly used CBC-MAC. Two versions of the XOR-MAC have been proposed, namely (a) stateless probabilistic XOR-MAC (or XMACR) and (b) stateful XOR-MAC (or XMACC). As our main focus is to study probabilistic MAC, we only describe XMACR (the details of XMACC can be found in [4]).

XMACR. XMACR is a probabilistic XOR-MAC in which a delta universal hash function H is applied on the message which is xor-ed with the output of $f(r)$ where f is a pseudo-random function and r is a uniformly chosen string. More formally, let f be a keyed function with input length d and output length n. Fix a parameter $b \leq d - 1$ where b is called the block-length. It is assumed

that the maximum length of the message that can be processed is up to $b2^{d-b-1}$. To process a message m, first m is padded to make the length of m multiple of b. Then m is parsed as $m_1||m_2||\ldots||m_L$ such that for each i, $|m_i| = b$. Tag generation algorithm chooses a $d-1$ bit string r uniformly at random from a coin space $\mathcal{R}$. We define $H(m)$ as follows : $H(m) = \oplus_{i=1}^{L} f_k(1||\langle i\rangle_{d-b-1}||m_i)$ where $\langle i\rangle_c$ is the c-bit binary representation of i. By following hash-then-mask paradigm, tag t is computed as $(r, f(0||r) \oplus H(m))$ and send it to the receiver with the corresponding message m. Note that this is a counter based MAC algorithm in which each message block m_i is appended with a fixed length encoding of the counter value $\langle i\rangle_c$. In [4], the following result has been proved.

$$\mathbf{Adv}^{\mathrm{SUF}}_{\mathrm{XMACR}}(q_m, q_v, t) \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{f}}(q_m + q_v, t') + \frac{2q_m^2}{2^d} + \frac{q_v}{2^n}$$

where $t' = t + O(q_m + q_v)$.

It is to be noted that using counter has a positive side in terms of providing better security than probabilistic MAC scheme. But in contrary to this fact, we also have discussed the possible disadvantages and infeasibility of using counter in implementing stateful MAC scheme. Using a random input instead of a nonce makes the degradation of the security to birthday bound. Thus the general question arises to what extent counter is to be used to beat the birthday bound security of the MAC. One possible approach is to use the pseudo random function with larger domain. For example instead of using $f : \{0,1\}^n \to \{0,1\}^m$, we use $g : \{0,1\}^{2n} \to \{0,1\}^m$. Using input length doubling transformation in general is inefficient. Bellare et al. in [3] have suggested a *parity method* in which the pseudo random function f is evaluated at several distinct random points and take the parity of result.

Finally, Minematsu in [18] proposed a probabilistic MAC scheme known as Enhanced-Hash-then-Mask (EHtM) which has been proven secure up to $q \approx 2^{2n/3}$ tag generation queries and still it uses only n-bit salt.

Enhanced Hash then Mask (EHtM). Let f_1 and f_2 be two n-bit pseduorandom function i.e. $f_i : \{0,1\}^n \to \{0,1\}^n$, $i = 1, 2$ and let $H : \{0,1\}^* \to \{0,1\}^n$ be an $\epsilon(\ell)$-almost xor-universal (AXU) hash function. EHtM works as follows : Given a message m, it first samples a n-bit string r uniformly at random from the coin space $\mathcal{R}$, and then pads the message m to make its length multiple of b. To generate the n-bit tag, it applies the hash function H to m and the output is xor-ed with r. f_2 is applied on the result $r \oplus H(m)$ which is again xor-ed with the output of $f_1(r)$. Then it sends the tuple (r, m, t) to the receiver where $t = f_1(r) \oplus f_2(r \oplus H(m))$. Therefore,

$$\mathsf{EHtM}(r, m) := f_1(r) \oplus f_2(r \oplus H(m)),$$

where $r \in \mathcal{R}$ is independent and uniformly sampled. In [18] it has been proved that the construction is secure up to $q \ll 2^{2n/3}$ queries. We note that the security proof is not tight as it has not been supported by any $2^{2n/3}$ query complexity attack.

4 Impossibility Result on Probabilistic MAC

We know that pseudorandom function implies unforgeable MAC for a deterministic tag generation algorithm. In this section we ask the same question for a probabilistic MAC.

Let us first consider a coin-explicit probabilistic MAC defined as $(r, \mathsf{cTG}(r, m))$. As deterministic MAC, one can similarly prove the security of MAC given that cTG is a PRF. More formally, we have the following result, for the sake of completeness proof of which can be found in Sec. A of the supporting material.

Theorem 4. *For any construction cTG_k, let TG denote the coin-explicit tag generation algorithm. Let Π be the probabilistic the complete MAC corresponding to this tag generation algorithm. Then,*

$$\mathbf{Adv}^{\mathrm{SUF}}_{\Pi}(q_m, q_v, t) \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{cTG}}(q_m + q_v, t') + \frac{q_v}{2^n},$$

where $t' \approx t$.

If we use a PRF core tag generation algorithm, then there is no need to use probabilistic MAC. We can simply use it with a fixed random coin as a deterministic MAC and we still achieve same security bound as guaranteed by the above result. Moreover, we can have better SUF-security than what is ensured by PRF security. Consider the following example.

Example 1. It is easy to see that the core-tag generation algorithm of XMACR is not PRF. In fact one can make four queries $(r, m), (r, m'), (r', m)$ and (r', m'). The sum of the output of these queries must be zero for the core-tag generation algorithm of XMACR. However, this can happen with probability 2^{-n} for ideal oracle. Similarly, Enhanced Hash-then-Mask or EHtM is shown to have at least $2^{2n/3}$ security. However, the core-tag generation algorithm can have at most $2^{n/2}$ PRF security (make $2^{n/2}$ queries with same r and so expect collision more than the ideal case). We see all these attacks later in Sect. 6.

Now we come back to our original question of the section :

Do we have any ideal system, indistinguishable to which ensure SUF or UF security?

Let us consider the ideal system $\mathsf{RF}^{\$}(m) = (r, \mathsf{RF}(r, m))$ which has been considered to define probabilistic PRF, where $r \xleftarrow{\$} \mathcal{R}$. It is easy to see that this is indeed a SUF secure MAC. In fact, for any q_m and q_v, we have

$$\mathbf{Adv}^{\mathrm{SUF}}_{\mathsf{RF}^{\$}}(q_m, q_v, t) \leq \frac{q_v}{2^n}.$$

To prove it, one can directly apply the theorem 4. Even though it is a perfectly secure MAC, we can define an UF-insecure construction indistinguishable to this ideal construction. Basically, we can modify the definition of the core-tag generation part for exactly one choice of r and m. For example, let $F(m)$ is same

as before except that when $r = m = 0$, it returns $(0, 0)$. In other words, $(0, 0)$ is a valid tag for the message 0. As it behaves exactly same as the ideal system except at the point $r = 0$, the maximum distinguishing advantage of this system would be at most $q/2^n$ where q is the number of queries. The same argument can be applied for any other ideal system we can imagine. Thus, we have the following result.

Theorem 5. *Let $\mathbf{I}_{mac} = (\mathcal{I}_k, \mathcal{V}_k)$ be a mac construction (a candidate choice for an ideal system) based on a tag generation construction $\mathcal{I}$. Suppose for all k, there exists t_k such that $\mathsf{Pr}[\mathcal{I}_k(m) = t_k] \leq 2^{-s}$ for some $s > 0$. Here the probability is computed under random coin of $\mathcal{I}$. Then there exists $\mathbf{I}'_{mac} = (\mathcal{I}'_k, \mathcal{V}'_k)$ such that $\mathbf{Adv}^{\mathrm{SUF}}_{\mathbf{I}'}(0, 1, t) = 1$ and $\mathbf{Adv}^{\mathcal{I}}_{\mathcal{I}'}(q) \leq 2^{-s}$.*

Proof. We follow exactly the same idea used for the ideal candidate to define probabilistic PRF. Fix any t^*. We modify the probability distribution on the output of the tag generation algorithm as follows. Whenever $t_k \neq t^*$, we define $\mathsf{Pr}[\mathcal{I}'_k(m) = t^*] = \mathsf{Pr}[\mathcal{I}_k(m) = t^*] + \mathsf{Pr}[\mathcal{I}_k(m) = t_k]$. Since the second term is small we are actually changing negligible amount on the probability at t^*. To make $\mathcal{I}'$ probability mass function, we define $\mathsf{Pr}[\mathcal{I}'_k(m) = t_k] = 0$. We can define the verification function $\mathcal{V}'$ accordingly. It can be checked that information theoretically, the distribution of $\mathcal{I}$ and $\mathcal{I}'$ is at most 2^{-s}, i.e. $\mathbf{Adv}^{\mathcal{I}}_{\mathcal{I}'}(q) \leq 2^{-s}$. Now we can construct a forgery which returns (m, t^*). As it is valid for all key k, it forges with probability one. □

Any ideal system capturing a probabilistic MAC with random coin of size d must have $s \approx d$. Thus, the above result says that no such ideal system for tag generation can exist to bound the SUF security.

5 Hash-then-Mask Probabilistic MAC: Candidate Schemes

In this section we explore the general constructions for hash-then mask probabilistic MAC. We start with some basic and trivially insecure schemes and gradually build towards a secure probabilistic MAC. To understand the effect of an AXU-hash, for every choice of hash-then-mask, we also consider a variant without applying hash. Here we need to assume that the message size is n. The simplest approach to get a MAC is: mask the message m (or $H(m)$) with $f(r)$ where f is a random function. These schemes are illustrated as C1 and C2 in Fig. 1.1. Obviously these simple schemes are not secure PRF. But is there any scope of getting adequate pPRF or MAC security? The answer is a partial yes. It is not hard to observe that C1 and C2 have birthday bound pPRF security. This is solely based on the randomness of f. In terms of the MAC security we can have trivial forgery attack on C1 with probability 1. Construction C2, is a generalization of XMACR [4]. Bellare et al. proved that the MAC advantage of XMACR has birthday bound security in terms of number of tag generation queries. We show that the bound is tight by virtue of our results on C2. In

order to improve the security one might think of applying an independent uniform random function on the message input. This is illustrated as C3 and C4 in Fig. 1.1. But as it turns out this will not elevate the security, rather the same attacks will work as in the case of C2 which is discussed in Sect. 6. In order to achieve beyond the birthday bound security, somehow we must have some interdependency between inputs of $f(\cdot)$ and $g(\cdot)$ which will make it harder to detect some pattern in the output. Mixing the two inputs might be a plausible approach to achieve this dependency. This approach is followed in C5 and C6 as illustrated in Fig. 1.1. C6 scheme was originally presented by Minematsu [18] and C5 is the variant of C6 in which hash function H is not applied. Among all these six constructions, C6 has PRF security. Both C5 and C6 have beyond birthday bound pPRF security in terms of the number of queries. We show that C6 carry forwards its pPRF security to MAC security. Quantitatively, we show that C6 is a beyond the birthday secure MAC till $2^{3n/4}$ tag generation queries which is a significant improvement over $2^{2n/3}$ bound shown by Minematsu [18]. Furthermore, all our bounds are tight as we give supporting attack strategies (see Sect. 6) that achieve the claimed advantages.

C1: $f(r) \oplus m$ 1. Input: $m \in \{0,1\}^n$. 2. Output: $t \in \{0,1\}^n$. 3. Choose $r \xleftarrow{\$} \mathcal{R}$. 4. Set $x \Leftarrow f(r)$. 5. Set $t \Leftarrow x \oplus m$. 6. return (r,t).	**C3:** $f(r) \oplus g(m)$ 1. Input: $m \in \{0,1\}^n$. 2. Output: $t \in \{0,1\}^n$. 3. Choose $r \xleftarrow{\$} \mathcal{R}$. 4. Set $x \Leftarrow f(r)$. 5. Set $y \Leftarrow g(m)$. 6. Set $t \Leftarrow x \oplus y$. 7. return (r,t).	**C5:** $f(r) \oplus g(r \oplus m)$ 1. Input: $m \in \{0,1\}^n$. 2. Output: $t \in \{0,1\}^n$. 3. Choose $r \xleftarrow{\$} \mathcal{R}$. 4. Set $x \Leftarrow f(r)$. 5. Set $y \Leftarrow r \oplus m$. 6. Set $z \Leftarrow g(y)$. 7. Set $t \Leftarrow x \oplus z$. 8. return (r,t).
C2: $f(r) \oplus H(m)$ 1. Input: $m \in \{0,1\}^\ell$. 2. Output: $t \in \{0,1\}^n$. 3. Choose $r \xleftarrow{\$} \mathcal{R}$. 4. Set $x \Leftarrow f(r)$. 5. Set $y \Leftarrow H(m)$. 6. Set $t \Leftarrow x \oplus y$. 7. return (r,t).	**C4:** $f(r) \oplus g(H(m))$ 1. Input: $m \in \{0,1\}^\ell$. 2. Output: $t \in \{0,1\}^n$. 3. Choose $r \xleftarrow{\$} \mathcal{R}$. 4. Set $x \Leftarrow f(r)$. 5. Set $y \Leftarrow H(m)$. 6. Set $z \Leftarrow g(y)$. 7. Set $t \Leftarrow x \oplus z$. 8. return (r,t).	**C6:** $f(r) \oplus g(r \oplus H(m))$ 1. Input: $m \in \{0,1\}^\ell$. 2. Output: $t \in \{0,1\}^n$. 3. Choose $r \xleftarrow{\$} \mathcal{R}$. 4. Set $u \Leftarrow f(r)$. 5. Set $v \Leftarrow H(m)$. 6. Set $w \Leftarrow r \oplus v$. 7. Set $x \Leftarrow g(w)$. 8. Set $t \Leftarrow u \oplus x$. 9. return (r,t).

Table 2: Candidate Tag Generation Algorithms Based on Hash-then-Mask Paradigm. The algorithms correspond to the illustrations in Fig. 1.1. f and g are two independent n-bit to n-bit random functions and H is a ϵ-AXU hash function with n bit output. Scheme C1, C3 and C5 take fixed length n bit message as input, where as schemes C2, C4 and C6 take messages of arbitrary length which is at most ℓ as input.

6 Attack Complexity

In earlier sections we defined 6 candidate schemes for hash-then-mask type probabilistic MAC constructions. In this section we provide some attack algorithms on these constructions. This would provide lower bounds of advantage for these schemes with respect to PRF, pPRF and MAC security.

6.1 PRF, MAC and pPRF Attack on C1

Let us first start with the simplest construction C1 whose core-tag is defined as $m \oplus f(r)$. Clearly, it is neither PRF nor MAC. A PRF distinguisher simply queries with (r_1, m_1) and (r_1, m_2) and checks whether $t_1 \oplus t_2 = m_1 \oplus m_2$ where t_1 and t_2 is the response obtained from the first query and second query respectively. Similarly, a forgery attacks works as follows: make a query m and then forge with $(r, m \oplus \Delta, t \oplus \Delta)$ where (r, t) is the response obtained from the first query and Δ is some non zero constant. A pPRF distinguisher makes $q = 2^{n/2}$ distinct queries $m_1, \ldots, m_q$ and observes $(r_1, t_1), \ldots, (r_q, t_q)$. We expect a collision on the observed salts r_i's. Suppose $r_i = r_j$, $i \neq j$. Then the event $t_i \oplus t_j = m_i \oplus m_j$ must hold whenever it interacts with the real construction. However, in case of the ideal oracle, t_i's are uniformly and independently distributed with r_i's and so the above event can happen with probability 2^{-n}.

6.2 PRF Attack on C2-C6

In this section we discuss the general PRF attack idea for construction C2,C3,C4,C5 and C6. The PRF attack idea for C2,C3,C4 and C5 is based on the formation of an alternating cycle of length 4 and requires 4 queries to distinguish it from the ideal oracle with probability 1 whereas for C6, we need at least $2^{n/2}$ many queries to distinguish it from ideal oracle with high probability.
medskip
Formation of Alternating Cycle of Length 4. The core-tag of rest of the constructions (C2-C6) can be viewed as $\mathrm{SUM}_{f,g}(r, y) = f(r) \oplus g(y)$ where f and g are some functions (sometimes g is an identity function) and y can be message m or hash output h of the message or it could be these two values masked by r. (e.g for construction C2 : g is identity function and y is the hash output of message. For consruction C3 : y is the message, for C4 : y is hash output of the message, for C5 : y is the message masked by the random string and for C6 : y is the hash output of the message masked by the random string). When $y = m$ or $y = m \oplus r$, the adversary knows immediately the values of y after it observes the output. However, for other cases, it does not know directly the value of y. The basic idea behind the most of the attacks is the following: Adversary will try to establish an alternating square which is a four tuple of the form $((r_1, y_1), (r_1, y_2), (r_2, y_1), (r_2, y_2))$. This is shown in the following figure:

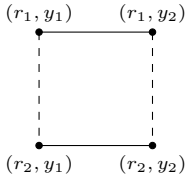

Fig. 6.1: An alternating square or cycle of length 4.

In general we draw a continuous (or dotted) line between two nodes whenever r (or y respectively) values are equal. Whenever an alternating square is formed, we have

$$\sum_{i=1}^{4} \mathrm{SUM}_{f,g}(r_i, y_i) = 0^n.$$

This event can be used as a distinguishing event whenever the values of r_i, y_i's present in the alternating cycle (definition can be found in Def. 4) are independent of the final outputs. In this case, the above event happens for ideal oracle with probability 2^{-n} as the final outputs t_i's are chosen independently.

The idea of forming an alternating cycle of length 4 is directly applied to the construction C2, C3 and C4 to launch PRF attack by just making only 4 distinct queries $(r_1, m_1), (r_1, m_2), (r_2, m_1)$ and (r_2, m_2) to form an alternating square on the values of (r_i, y_i).

However for C4, we do not have to know the value of y_i as $y_i = H(m_i)$. However, we know that the alternating square is formed.

PRF Attack on C6. In C6 the adversary can observe for collisions in the underlying H function. The adversary fixes the r value and uses q random messages. With high probability there will be a collision in the tag values, say t_i and t_j. Therefore either $H(m_i) = H(m_j)$ or there is a collision in g. The adversary can detect the first case by querying (r', m_i) and (r', m_j). If $t'_i = t'_j$, then the adversary is interacting with C6. The adversary can eliminate any false positives arising due to collisions in g, by constant number of repeatitions of the experiment. Clearly the adversary's advantage is lower bounded by the collision probability on t_i values, i.e., $\Omega(q^2/2^n)$.

6.3 pPRF Attack on C2,C3 and C4

We try to ensure alternating square by finding collision on r_i values by querying $2^{n/2}$ pair of messages m and m'. Once we observe two collision pairs, say $r_i = r_{i'}$ and $r_j = r_{j'}$ such that r_i, r_j are random coins for m and $r_{i'}, r_{j'}$ are random coins of m', we know that alternating square is formed for these four queries. Sum of the tag-output for these four queries would be zero with probability one (or 2^{-n}) for real construction (or the ideal respectively).

6.4 Forging Attack on C2, C3 and C4

Forging attacks for C2, C3 and C4 are same as the attack described in [4] for XOR-MAC construction (i.e. C2). For the sake of completeness we briefly describe the attack for C2. We make $q = 2^{n/2}$ distinct queries $m_1, \ldots, m_q$ and observe $r_i = r_j$ with $i \neq j$. This would leak the value of $H(m_i) \oplus H(m_j)$, say δ. Now we make one more query m_i and obtain response (r, t). We can forge $(m_j, (r, t \oplus \delta))$. The exactly same attack can be carried for C3 and C4. In terms of

$\underline{\mathcal{A}}$:

1. for fix m_1 and $i = 1$ to $q/2$ do
   (a) $m_1 \Rightarrow \mathcal{O}$
   $(r_i, t_i) \Leftarrow \mathcal{O}$.
   (b) add (r_i, m_1, t_i) to L_1 indexed on r_i and t_i.
2. for fix m_2 and $i = 1$ to $q/2$ do
   (a) $m_2 \Rightarrow \mathcal{O}$
   $(r_i, t_i) \Leftarrow \mathcal{O}$.
   (b) add (r_i, m_2, t_i) to L_2 indexed on r_i and t_i.
3. for $r_i \in L_1$ do
   (a) if $r_i \in L_2$ then let (r_j, m_2, t_j) be that entry.
   (b) add (r_i, t_i, t_j) to $\mathcal{C}$ as (r, t^α, t^β).
4. if $|\mathcal{C}| > 2$ then
   (a) for each pair of $c_i, c_j \in \mathcal{C}$
   check $t^\alpha_{c_i} \oplus t^\beta_{c_i} = t^\alpha_{c_j} \oplus t^\beta_{c_j}$
   (b) if equal for at least two such pairs return 1.
5. return 0.

Fig. 6.2: pPRF adversary $\mathcal{A}$ for C5.

the alternating square, we make three sides of an alternating square (also called alternating path) by making tag generation queries and then we construct a forging attempt which completes the alternating square.

6.5 pPRF Attack on C5 and C6

Suppose the adversary is interacting with an oracle $\mathcal{O}$. We describe an adversary $\mathcal{A}$ in Fig. 6.2 that distinguishes C5 in q queries with high probability. For $t^\alpha_{c_i} \oplus t^\beta_{c_i} = t^\alpha_{c_j} \oplus t^\beta_{c_j}$ we have either $r_{c_i} \oplus r_{c_j} = m_1 \oplus m_2$ or a collision on g. The false positives due to collisions on g can be eliminated by repeating the experiment a constant number of times. For the other case we need two r collisions between L_1 and L_2 and for those r values r_α and r_β we must have $r_\alpha \oplus r_\beta = m_1 \oplus m_2$. The probability of getting such r values is then bounded by $q^4/2^{3n}$. One can similarly construct an adversary for C6 with similar advantage as in C5.

6.6 Forging Attack on C5 and C6

Forging Attack on C5. Suppose the adversary is interacting with an oracle $\mathcal{O}$. We describe an adversary $\mathcal{A}_1$ in Fig. 6.4 that strongly forges C5 in q_m tag generation queries with high probability. We also show the basic idea of forgery in Fig. 6.5. Forging requires a colliding r pair (r_i, r_j) such that $\exists\ r_d \in L_1$ for which $r_i \oplus r_d = m_1 \oplus m_2$. This can happen with probability to $q_m^3/2^{2n}$.

Forging Attack on C6. Suppose the adversary is interacting with an oracle

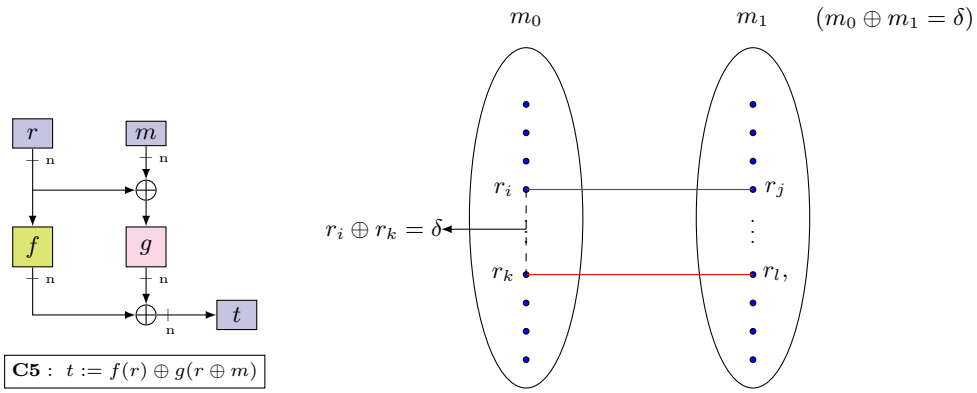

Fig. 6.3: pPRF Attack on C5; Distinguishing Event : If $t_i \oplus t_j \oplus t_k \oplus t_l = 0$, output 1.

$\mathcal{O}$. We describe an adversary $\mathcal{A}_2$ in Fig, 6.4 that strongly forges C6 in q_m tag generation queries with high probability. The SUF adversary is essentially an extension of the pPRF adversary that tries to create a new alternating cycle (and hence a valid forge) using an earlier alternating cycle (from pPRF). The advantage can be bounded similarly.

7 Security Proof

In this section we show upper bounds of constructions C5 and C6 in terms of three security notions (PRF/ pPRF/ MAC) we consider. We have already seen in Sect. 5 that C5 is not a PRF, therefore, we will show the pPRF and MAC bound of C5 and PRF, pPRF and MAC bound of C6. In this regard, we would like to mention that as the security bound of some of the constructions are already known and the security bound of some of the constructions are easy to observe, those results are shown in Sec. C of the supporting material. In specific, we show the pPRF and MAC security proof of constructions C1-C4 in Sec. C of the supporting material. We first observe that except C6, no construction provides PRF security. So we provide brief sketch of the PRF proof of C6 as its main idea is similar to that of hash-then-prf result.

7.1 pPRF and MAC Security Proof of Construction C5

We have seen in Sect. 6 with $q \approx 2^{2n/3}(\ \approx 2^{3n/4})$ queries, any efficient probabilistic adversary can forge the MAC (break the pPPRF security) respectively of the construction $F_{f,g}(m) = f(r) \oplus g(r \oplus m)$ where f and g are two n-bit random functions.

In this section we will show that the MAC and pPRF security bound of the construction is tight by proving the advantage of the MAC is upper bounded by

$\underline{\mathcal{A}_1}$:

1. choose $m_1, m_2 \in \{0,1\}^n$.
2. set $\delta := m_1 \oplus m_2$.
3. for fix m_1 and $i = 1$ to $q_m/2$ do
   (a) $m_1 \Rightarrow \mathcal{O}$
   $(r_i, t_i) \Leftarrow \mathcal{O}$.
   (b) add (r_i, m_1, t_i) to L_1 indexed on r_i and t_i.
4. for fix m_2 and $i = 1$ to $q_m/2$ do
   (a) $m_2 \Rightarrow \mathcal{O}$
   $(r_i, t_i) \Leftarrow \mathcal{O}$.
   (b) add (r_i, m_2, t_i) to L_2 indexed on r_i and t_i.
5. for each $r_i \in L_1$ do
   (a) if $r_i \in L_2$ then let (r_j, m_2, t_j) be that entry.
   (b) store (r_i, t_i, t_j) in $\mathcal{C}$ as (r, t^α, t^β).
6. for each $r_c \in \mathcal{C}$ do
   (a) search for $(r_d, m_1, t_d) \in L_1$ such that $r_d \notin L_2 \wedge r_c \oplus r_d = \delta$.
   (b) if no such r_d is found repeat the experiment.
   (c) return $(m_2, (r_d, t_d \oplus t_c^\alpha \oplus t_c^\beta))$.

$\underline{\mathcal{A}_2}$:

1. for fix m_1 and $i = 1$ to $q/2$ do
   (a) $m_1 \Rightarrow \mathcal{O}$
   $(r_i, t_i) \Leftarrow \mathcal{O}$.
   (b) add (r_i, m_1, t_i) to L_1 indexed on r_i and t_i.
2. for fix m_2 and $i = 1$ to $q/2$ do
   (a) $m_2 \Rightarrow \mathcal{O}$
   $(r_i, t_i) \Leftarrow \mathcal{O}$.
   (b) add (r_i, m_2, t_i) to L_2 indexed on r_i and t_i.
3. for $r_i \in L_1$ do
   (a) if $r_i \in L_2$ then let (r_j, m_2, t_j) be that entry.
   (b) add (r_i, t_i, t_j) to $\mathcal{C}$ as (r, t^α, t^β).
4. if $|\mathcal{C}| > 2$ then
   (a) for each pair of $c_i, c_j \in \mathcal{C}$
   check $t^\alpha_{c_i} \oplus t^\beta_{c_i} = t^\alpha_{c_j} \oplus t^\beta_{c_j}$.
   (b) if equal compute $\delta = r_{c_i} \oplus r_{c_j}$.
   (c) if all δ's are distinct then repeat the experiment.
   (d) if a δ repeats at least twice
      i. for each $r_c \in \mathcal{C}$ do
         A. search for $(r_d, m_1, t_d) \in L_1$ such that $r_d \notin L_2 \wedge r_d \oplus r_c = \delta$.
         B. if no such r_d is found repeat the experiment.
         C. return $(m_2, (r_d, t_d \oplus t_c^\alpha \oplus t_c^\beta))$.

Fig. 6.4: SUF adversary $\mathcal{A}_1$ for C5 and $\mathcal{A}_2$ for C6.

$\frac{q_m^3}{2^{2n}}$ and advantage of pPRF is upper bounded by $\frac{q^4}{2^{3n}}$. The proof for bounding both the advantage is same, only difference will occur in their corresponding bad event.

(A) Bounding MAC Advantage of Construction C5. For bounding MAC advantage we have the following result

Theorem 6. *Let f_{k_1} and f_{k_2} be two independently keyed functions. Then we have*

$$\mathbf{Adv}_{C_5}^{RF^\$,\perp}(q_m, q_v, t) \leq \mathbf{Adv}_{f_{k_1}}^{\mathrm{prf}}(q_m + q_v, t) + \mathbf{Adv}_{f_{k_2}}^{\mathrm{prf}}(q_m + q_v, t) + \frac{q_m^3}{2^{2n}} + \frac{q_v}{2^n}.$$

Proof. We prove the theorem using Coefficients H technique. By using hybrid argument, we can simply assume that C5 is based on two independent random functions f and g instead of keyed PRF at the cost of $\mathbf{Adv}_{f_{k_1}}^{\mathrm{prf}}(q_m + q_v, t) + \mathbf{Adv}_{f_{k_2}}^{\mathrm{prf}}(q_m + q_v, t)$. We apply Coefficients H technique on this hybrid construction.

We first define a transcript which is equivalent to the definition of view mentioned in the discussion related to Coefficient H technique. A transcript is defined as a pair of input and output that the adversary obtains during the interaction with the real oracle.

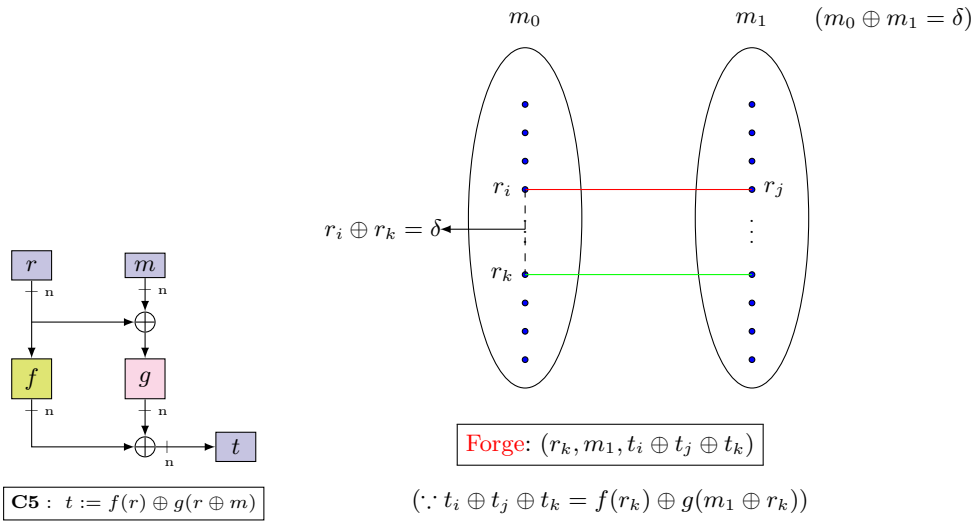

Fig. 6.5: Forging Attack on C5

Let τ be a transcript where $\tau := \{(r_i, m_i, t_i)_{1 \le i \le q_m}, (\widetilde{r}_j, \widetilde{m}_j, \widetilde{t}_j, 0)_{1 \le j \le q_v}\}$. A transcript τ is said to be valid if $\Pr[\tau \text{ is realized }] > 0$ in real world. Regard to our construction, a transcript τ is said to be valid if the following conditions hold

1. $(r_i, m_i) = (r_j, m_j) \Rightarrow t_i = t_j$, $1 \le i \le q_m$
2. $(\widetilde{r}_j, \widetilde{m}_j, \widetilde{t}_j)$ is non-trivial, $1 \le j \le q_v$

Clearly, for an invalid transcript τ, $\Pr[\tau \text{ is realized }] = 0$. Let Θ be the set of all valid transcripts. Having defined the set of valid transcripts, we identify a set of good transcripts.

Step I. Identifying Good Transcripts. We identify a set of good transcripts. A valid transcript τ is said to be bad if the following condition hold:
$\exists i, j, k$ such that $r_i = r_j$ and $r_i \oplus r_k = m_i \oplus m_k$. Let $\Theta_{\text{bad}} \subseteq \Theta$ be the set of all bad transcripts. Therefore, $\Theta_{\text{good}} := \Theta \setminus \Theta_{\text{bad}}$ be the set of all good transcripts. Now we make a following claim.

Claim 1 *Let τ be a good transcript. We define a set τ' corresponding to τ as follows $\tau' := \{(r_i, (r_i \oplus m_i)\}_{1 \le i \le q_m}$. Then τ' does not contain any alternating cycle.*

Proof. For the sake of contradiction, let us assume that there is an alternating cycle in τ', i.e. $r_1 = r_2, r_2 \oplus r_3 = m_2 \oplus m_3, \ldots, r_q \oplus r_1 = m_q \oplus m_1$. But this shows that we have $r_1 = r_2, r_2 \oplus r_3 = m_2 \oplus m_3$ which is not allowed in a good transcript. Therefore, there exists no alternating cycle in τ'. □

Step II. Probability of Bad Transcript in Ideal World. We bound the probability of the identified bad transcript in the ideal world. We calculate the

probability of realizing bad transcript in ideal world as follows.

$$\begin{aligned}\Pr[X_{\text{id}} \in \Theta_{\text{bad}}] &= \sum_{i,j,k} \Pr[r_i = r_j, r_i \oplus r_k = m_i \oplus m_k] \\ &= \sum_{i,j,k} \Pr[r_j \oplus r_k = m_i \oplus m_k] \cdot \Pr[r_i = r_i] \\ &\leq \frac{q_m^3}{2^{2n}}\end{aligned}$$

Therefore, $\epsilon_{\text{bad}} \leq \frac{q_m^3}{2^{2n}}$.

Let us fix a good transcript $\tau \in \Theta_{\text{good}}$ where

$$\tau := \{(r_i, m_i, t_i)_{1 \leq i \leq q_m}, (\widetilde{r}_j, \widetilde{m}_j, \widetilde{t}_j, 0)_{1 \leq j \leq q_v}\}.$$

We show that the real interpolation probability for τ is almost as high as the ideal interpolation probability for the corresponding transcript. To show that, we first derive the ideal interpolation probability of a good transcript. That is we bound the probability that a good transcript is realized in ideal world.

Step III. Probability of Good Transcript in Ideal World. Let $\tau \in \Theta_{\text{good}}$ be a fixed good transcript. Then we have the following

$$\begin{aligned}\Pr[X_{\text{id}} = \tau] &= \Pr[r_i \xleftarrow{\$} \mathcal{R}, \$_{\mathsf{rf}}(r_i, m_i) = t_i, \forall 1 \leq i \leq q_m] \\ &= \Pr[r_i \xleftarrow{\$} \mathcal{R}, \forall `1 \leq i \leq q_m] \cdot \Pr[\$_{\mathsf{rf}}(r_i, m_i) = t_i, \forall 1 \leq i \leq q_m] \\ &= \frac{1}{2^{n(q_m + q_m')}}\end{aligned}$$

where $q_m^{'}$ be the distinct number of t_i in the transcript τ.

Step IV. Probability of Good Transcript in Real World. The last step is to calculate the real interpolation probability of a good transcript $\tau := \{(r_i, m_i, t_i)_{1 \leq i \leq q_m}, (\widetilde{r}_j, \widetilde{m}_j, \widetilde{t}_j, 0)_{1 \leq j \leq q_v}\}$. We bound the probability of τ realized in real world. Let us denote $X_i = f(r_i)$ and $Y_i = g(r_i \oplus m_i)$, $1 \leq i \leq q_m$. We also denote $\widetilde{X}_j = f(\widetilde{r}_j)$ and $\widetilde{Y}_j = g(\widetilde{r}_j \oplus \widetilde{m}_j)$, $1 \leq j \leq q_v$. We also denote the event $X_i \oplus Y_i = t_i, 1 \leq i \leq q_m$ by E and $\widetilde{X}_j \oplus \widetilde{Y}_j \neq t_j$ by F_j

$$\begin{aligned}\Pr[X_{\text{re}} = \tau] &= \Pr[r_i \xleftarrow{\$} \mathcal{R}, X_i \oplus Y_i = t_i, 1 \leq i \leq q_m, \widetilde{X}_j \oplus \widetilde{Y}_j \neq t_j, 1 \leq j \leq q_v] \\ &= \Pr[r_i \xleftarrow{\$} \mathcal{R} 1 \leq i \leq q_m] \cdot \Pr[E \wedge (\wedge_j F_j)] \\ &\geq \frac{1}{2^{nq_m}} \cdot (\Pr[E] - \sum_j Pr[E \wedge \neg F_j]) \\ &\geq \frac{1}{2^{nq_m}} \cdot (\frac{1}{2^{nq_m'}} - \frac{q_v}{2^n} 2^{-nq_m'}) \\ &\geq \frac{1}{2^{n(q_m + q_m')}} \cdot (1 - \frac{q_v}{2^n})\end{aligned}$$

Therefore, $\epsilon_{\text{ratio}} = \frac{q_v}{2^n}$. To finish the proof we just need to argue that $\Pr[E] = \frac{1}{2^{nq'_m}}$ and $\Pr[E \wedge F_j] \leq \frac{1}{2^{(q'_m+1)}}$. We have already seen in Claim 1 that being a good transcript τ, the set $\{r_i, (r_i \oplus m_i)\}_{1 \leq i \leq q'_m}$ does not contain an alternating cycle. Therefore, from Lemma 2 we have, $\Pr[E] = \frac{1}{2^{nq'_m}}$. With the same argument one can show that $\Pr[E \wedge F_j] \leq \frac{1}{2^{(q'_m+1)}}$. Therefore, we have $\epsilon_{\text{bad}} = \frac{q_m^3}{2^{2n}}$ and $\epsilon_{\text{ratio}} = \frac{q_v}{2^n}$. Therefore from Theorem of Coefficients H technique our result follows. □

Corollary 1. *Let f_{k_1} and f_{k_2} be two independently keyed functions. Then we have*

$$\mathbf{Adv}^{\text{SUF}}_{\text{C5}}(q_m, q_v, t) \leq \mathbf{Adv}^{\text{prf}}_{f_{k_1}}(q_m + q_v, t) + \mathbf{Adv}^{\text{prf}}_{f_{k_2}}(q_m + q_v, t) + \frac{q_m^3}{2^{2n}} + \frac{q_v}{2^n}.$$

The proof of the corollary follows from Theorem 6 and Lemma 1.

(B) Bounding pPRF Advantage of Construction C5. The proof for bounding the pPRF advantage for C5 would be exactly the same as that of MAC advantage for C5, the only difference will be in the corresponding bad event. Thus we have the following.

Theorem 7. *Let f_{k_1} and f_{k_2} be two independently keyed functions. Then we have*

$$\mathbf{Adv}^{\text{pprf}}_{\text{C5}}(q, t) \leq \mathbf{Adv}^{\text{prf}}_{f_{k_1}}(q, t) + \mathbf{Adv}^{\text{prf}}_{f_{k_2}}(q, t) + \frac{q^4}{2^{3n}}.$$

Proof. By using hybrid argument, we can simply assume that C5 is based on two independent random functions f and g instead of keyed PRF at the cost of $\mathbf{Adv}^{\text{prf}}_{f_{k_1}}(q_m + q_v, t) + \mathbf{Adv}^{\text{prf}}_{f_{k_2}}(q_m + q_v, t)$. We apply Coefficients H technique on this hybrid construction.

As before, we consider a transcript $\tau := \{(r_i, m_i, t_i)_{1 \leq i \leq q}\}$. Regard to our construction, τ is said to be valid if the following condition holds :

1. $(r_i, m_i) = (r_j, m_j) \Rightarrow t_i = t_j$, $1 \leq i \leq q$.

Θ be the set of all valid transcripts. Having defined the set of valid transcripts, we identify a set of bad transcripts Θ_{bad}. $\tau \in \Theta_{\text{bad}}$ is said to be a bad transcript if $\exists i, j, k, l$ such that $r_i = r_j, r_k = r_l$ and $r_i \oplus r_k = m_i \oplus m_k$. Bounding the probability of realizing bad transcript in ideal world is $\frac{q^4}{2^{3n}}$ as we choose four distinct indices and we have three independent events each of which holds with probability $\frac{1}{2^n}$. Let $\Theta_{\text{good}} := \Theta \setminus \Theta_{\text{bad}}$ be the set of good ranscripts. It is easy to see that for a good transcript $\tau \in \Theta_{\text{good}}$, the set $\{r_i, (r_i \oplus m_i)\}_{1 \leq i \leq q}$ does not contain any alternating cycle proof of which follows exactly in the same way as the proof of Claim 1.

Let $\tau \in \Theta_{\text{good}}$. Then the probability of realizing τ in ideal world is exactly $\frac{1}{2^n(q+q')}$ where q' be the distinct number of t_i in the transcript τ. Now we calculate the probability of realizing a good transcript in real world as follows

Probability of Good Transcript in Real World. We bound the probability that a good transcript τ is realized in real world. Let us denote $X_i = f(r_i)$ and

$Y_i = g(r_i \oplus m_i)$, $1 \le i \le q$. We also denote the event $X_i \oplus Y_i = t_i, 1 \le i \le q$ by E.

$$\begin{aligned}\Pr[X_{\text{re}} = \tau] &= \Pr[r_i \xleftarrow{\$} \mathcal{R}, X_i \oplus Y_i = t_i, 1 \le i \le q] \\ &= \Pr[r_i \xleftarrow{\$} \mathcal{R} 1 \le i \le q] \cdot \Pr[E] \\ &\ge \frac{1}{2^{nq}} \cdot \Pr[E] \\ &\ge \frac{1}{2^{n(q+q')}}\end{aligned}$$

To finish the proof we need to show that $\Pr[E] = \frac{1}{2^{nq'_m}}$. We have argued that for a good transcript τ, the set $\{r_i, (r_i \oplus m_i) : 1 \le i \le q\}$ does not contain any alternating cycle and therefore from Lemma 2 we have $\Pr[E] = \frac{1}{2^{nq'_m}}$. Therefore, according to Theorem of Coefficients H technique, our result follows. □

7.2 PRF Security Bound of Construction C6.

We have shown that none of the constructions from C1 to C5 is secure PRF and we argued a $2^{n/2}$ PRF attack for construction C6. Here we show the PRF security bound of C6 is tight by proving the security bound of C6 is upper bounded by $O(q^2/2^n)$.

Theorem 8. *Let f_{k_1} and f_{k_2} be two n bit pseudo-random functions and H be a ϵ-AXU Universal hash function. Then we have*

$$\mathbf{Adv}^{\text{prf}}_{\text{C6}}(q, \ell, t) \le \mathbf{Adv}^{\text{prf}}_{f_{k_1}}(q, t') + \mathbf{Adv}^{\text{prf}}_{f_{k_2}}(q, t') + \binom{q}{2} \times \epsilon,$$

where $t = t' + O(qT_h)$

Proof. By using hybrid argument, we can simply assume that C6 is based on two independent random functions f and g instead of keyed PRF at the cost of $\mathbf{Adv}^{\text{prf}}_{f_{k_1}}(q, t') + \mathbf{Adv}^{\text{prf}}_{f_{k_2}}(q, t')$. Now this hybrid construction will behave perfectly random as long as there is no collision in the input of g. Note that for any query (r, m), the input of g is $H(m) \oplus r$. As H is ϵ-AXU, the function mapping (r, m) to $r \oplus H(m)$ is ϵ-universal. Hence by using the composition result for hash-then-prf we get our desired bound. Here we note that the effect of $f(r)$ would be completely masked by the independent random function output of g. □

7.3 pPRF and MAC Security Proof of Construction C6 : Enhanced Hash then Mask (EHtM)

We have seen in Sect. 6 that there is a $2^{n/2}$ prf-attack, $2^{3n/4}$ pPRF and MAC attack on the construction $F_{f,g}(m) = f(r) \oplus g(r \oplus H(m))$. We have also shown in Sect. 7, Theorem 8 the PRF security of C6. In this section we will show that the pPRF and MAC security bound of the construction C6 is tight by proving the advantage of the pPRF and MAC is upper bounded by $\frac{q_m^4}{2^{3n}}$.

To bound the pPRF and MAC advantage, we show the distinguishing advantage to distinguish the real oracle (C6) from the ideal one ($\mathsf{RF}^{\$}, \perp$) is upper bounded by $O(q_m^4/2^{3n})$ using Coefficients H technique as a tool.

(A) Bounding MAC Advantage of Construction C6.

Theorem 9. *Let f_{k_1} and f_{k_2} be two independently keyed functions and $H : \{0,1\}^* \to \{0,1\}^n$ be a ϵ-AXU hash function. Then we have*

$$\mathbf{Adv}_{\mathrm{C6}}^{RF^{\$},\perp}(q_m, q_v, \ell, t) \leq \mathbf{Adv}_{f_{k_1}}^{\mathrm{prf}}(q_m + q_v, t') + \mathbf{Adv}_{f_{k_2}}^{\mathrm{prf}}(q_m + q_v, t') + \frac{q_m^4}{2^{3n}} + \frac{10q_v}{2^n},$$

where $t = t' + O(qT_h)$

Proof. By using hybrid argument, we can simply assume that C6 is based on two independent random functions f and g instead of keyed PRF at the cost of $\mathbf{Adv}_{f_{k_1}}^{\mathrm{prf}}(q_m + q_v, t') + \mathbf{Adv}_{f_{k_2}}^{\mathrm{prf}}(q_m + q_v, t')$. We apply Coefficients H technique on this hybrid construction.

To apply the Coefficients H technique, we need to

1. Identify a set of good views Θ_{good} which is realized with high probability while interacting with the ideal oracle.
2. Show that for any fixed good view $\tau \in \Theta_{\mathrm{good}}$, the real interpolation probability is almost as high as the ideal interpolation probability.

As before we define a transcipt τ which is defined as the input-output pair that the adversary obtains during the interaction with the real oracle. Moreover, we also assume that after interaction with the oracle it releases the values of $h_i := H(m_i)$. Thus after interaction, adversary obtains an extended transcript $\tau = \{(r_i, m_i, t_i, h_i)_{1 \leq i \leq q_m}, (\widetilde{r}_j, \widetilde{m}_j, \widetilde{t}_j, \widetilde{h}_j, 0)_{1 \leq j \leq q_v}\}$. For a valid transcript τ the following conditions must hold

1. $m_i = m_j \Rightarrow h_i = h_j$, $\widetilde{m}_i = \widetilde{m}_j \Rightarrow \widetilde{h}_i = \widetilde{h}_j$. Morover, $m_i = \widetilde{m}_j \Rightarrow h_i = \widetilde{h}_j$, $1 \leq i, j \leq q_m$.
2. $(r_i, m_i) = (r_j, m_j) \Rightarrow t_i = t_j$, $1 \leq i, j \leq q_m$.
3. $(\widetilde{r}_j, \widetilde{m}_j, \widetilde{t}_j)$ is fresh, $1 \leq j \leq q_v$.
4. $t_i = \mathsf{EHtM}(r_i, m_i)$, $1 \leq i \leq q_m$.
5. $h_i = H(m_i)$ and $\widetilde{h}_j = H(\widetilde{m}_j)$, $1 \leq i \leq q_m$, $1 \leq j \leq q_v$.

For any valid transcript τ, $\Pr[X_{\mathrm{re}} = \tau] > 0$. Let Θ be the set of all valid transcripts. Having defined the valid transcript, we now identify a set of good transcripts.

Step I. Identifying Good Transcripts. In order to identify the set of good transcripts, we identify the set of bad transcripts Θ_{bad}. A transcript τ is said to be a bad transcript if either of the following conditions hold :

(C.1) $\exists 1 \leq i, j \leq q_m$ such that $(r_i, h_i) = (r_j, h_j)$
(C.2) $\exists 1 \leq i, j, k, l \leq q_m$ such that $r_i = r_j, r_j \oplus r_k = h_j \oplus h_k, r_k = r_l$
(C.3) $\mathsf{mc}(r) \geq 4$

(C.4) $\mathsf{mc}\{t_i + t_j : r_i = r_j\} \geq 4$
(C.5) $\exists 1 \leq i, j, k, l \leq q_m$ such that $t_k \oplus t_l = t_j \oplus \widetilde{t}_i, r_k = r_l, \widetilde{r}_i = r_j$

A transcript τ is said to be *good* if none of the above conditions hold. Let Θ_{good} denotes the set of all good transcripts which is $\Theta_{\text{good}} := \Theta \backslash \Theta_{\text{bad}}$ and $\Theta_{\text{good}} \subseteq \Theta$. Now we make the following claim

Claim 2 *Let τ be a good transcript. We define a set τ' corresponding to τ as follows $\tau' := \{(r_i, (r_i \oplus h_i)\}_{1 \leq i \leq q_m}$. Then τ' does not contain any alternating cycle. Moreover $\tau'_j := \{(r_i, (r_i \oplus h_i)\}_{1 \leq i \leq q_m} \cup \{\widetilde{r}_j, (\widetilde{r}_j \oplus \widetilde{h}_j)\}$ does not contain any alternating cycle.*

Proof. For the sake of contradiction, let us assume that τ' contains an alternating cycle. That implies $r_1 = r_2, r_2 \oplus r_3 = h_2 \oplus h_3, \ldots, r_{q_m} \oplus r_1 = h_{q_m} \oplus h_1$. Therefore, we will get three equations $r_1 = r_2, r_{q_m - 1} = r_{q_m}$ and $r_{q_m} \oplus r_1 = h_{q_m} \oplus h_1$, but it violates the assumption that τ is a good transcript. Thus, τ' does not contain any alternating cycle. Following the same argument one can show that if τ'_j contains an alternating cycle then the assumption of τ being valid would have been violated. Therefore, τ'_j does not contain any alternating cycle. □

Therefore, being a good transcript τ, the set $\{r_i, (r_i \oplus h_i)\}_{1 \leq i \leq q_m}$ and the set $\{r_i, (r_i \oplus h_i)\}_{1 \leq i \leq q_m} \cup \{\widetilde{r}_j, (\widetilde{r}_j \oplus \widetilde{m}_j\}$ does not contain any alternating cycle.

Step II. Probability of Bad Transcript in Ideal World. Let Θ^i_{bad} be the set of all transcripts that satisfies condition (C.i). Therefore, we want to compute $\epsilon_{\text{bad}} = \Pr[X_{\text{id}} \in \Theta_{\text{bad}}] \leq \sum_i \Pr[X_{\text{id}} \in \Theta^i_{\text{bad}}]$. We have bound the probability of realizing bad transcripts in ideal world in the following lemma, proof of which is postponed to the following section.

Lemma 4. *Let Θ_{bad} be the set of all valid and bad transcripts. Let X_{id} be the probability distribution of transcript realized in ideal world. Then,*

$$\Pr[X_{\text{id}} \in \Theta_{\text{bad}}] \leq \frac{q_m^4}{2^{3n}} + \frac{9q_v}{2^n}.$$

Due to Lemma 4 we have, $\epsilon_{\text{bad}} \leq \frac{q_m^4}{2^{3n}} + \frac{9q_v}{2^n}$.

Step III. Probability of Good Transcript in Ideal World. Let us fix any good transcript $\tau \in \Theta_{\text{good}}$. We bound the probability of realizing a good transcript in ideal world as follows.

$$\begin{aligned}
\Pr[X_{\text{id}} = \tau] &= \Pr[r_i \xleftarrow{\$} \mathcal{R}, \$_{\mathsf{rf}}(r_i, m_i) = t_i, H(m_i) = h_i, H(\widetilde{m}_j) = \widetilde{h}_j] \\
&= \Pr[r_i \xleftarrow{\$} \mathcal{R}] \cdot \Pr[\$_{\mathsf{rf}}(r_i, m_i) = t_i] \cdot p_H \\
&\leq \frac{1}{2^{n(q_m + q'_m)}} \cdot p_H
\end{aligned}$$

where $p_H := \Pr[H(m_i) = h_i, H(\widetilde{m}_j) = \widetilde{h}_j, 1 \leq i \leq q_m, 1 \leq j \leq q_v]$ is the joint probability of the hash output and q'_m be the distinct number of t_i in the

transcript τ.

Step IV. Probability of Good Transcript in Real World. The last step is to calculate the real interpolation probability of a good transcript τ. We bound the probability that a good trancript is realized in real world. Let us denote $X_i = f(r_i)$ and $Y_i = g(r_i \oplus h_i)$, $1 \le i \le q_m$. We also denote $\widetilde{X}_j = f(\widetilde{r}_j)$ and $\widetilde{Y}_j = g(\widetilde{r}_j \oplus \widetilde{h}_j)$, $1 \le j \le q_v$. We also denote the event $X_i \oplus Y_i = t_i, 1 \le i \le q_m$ by E and $\widetilde{X}_j \oplus \widetilde{Y}_j \neq t_j$ by F_j

$$
\begin{aligned}
&\Pr[X_{\mathrm{re}} = \tau] \\
&= \Pr[r_i \xleftarrow{\$} \mathcal{R}, X_i \oplus Y_i = t_i, H(m_i) = h_i, \widetilde{X}_j \oplus \widetilde{Y}_j \neq t_j, H(\widetilde{m}_j) = \widetilde{h}_j] \\
&= \Pr[r_i \xleftarrow{\$} \mathcal{R} 1 \le i \le q_m] \cdot \Pr[E \wedge (\wedge_j F_j)] \cdot p_H \\
&\ge \frac{1}{2^{nq_m}} \cdot (\Pr[E] - \sum_j Pr[E \wedge \neg F_j]) \cdot p_H \\
&\ge \frac{1}{2^{nq_m}} \cdot (\frac{1}{2^{nq_m^{'}}} - \frac{q_v}{2^n} 2^{-nq_m^{'}}) \cdot p_H \\
&\ge \frac{1}{2^{n(q_m + q_m^{'})}} \cdot (1 - \frac{q_v}{2^n}) \cdot p_H
\end{aligned}
$$

To finish the proof we just need to argue that $\Pr[E] = \frac{1}{2^{nq_m^{'}}}$ and $\Pr[E \wedge F_j] = \frac{1}{2^{n(q_m^{'}+1)}}$. This is easy to follow due to the good transncript τ, we have already argued that $\{r_i, (r_i \oplus h_i)\}_{1 \le i \le q_m^{'}}$ comtains no alternating cycle and therefore due to Lemma 2 we have $\Pr[E] = \frac{1}{2^{nq_m^{'}}}$. Moreover, using the same argument one can see that $\Pr[E \wedge F_j] = \frac{1}{2^{n(q_m^{'}+1)}}$ follows. This concludes that $\epsilon_{\mathrm{ratio}} \le \frac{q_v}{2^n}$. Therefore, according to Theorem of Coefficient H technniques, $\mathbf{Adv}_{\mathrm{C6}}^{\mathsf{RF}^{\$}, \perp}(q_m, q_v, \ell) \le \frac{q_m^4}{2^{3n}} + \frac{10 q_v}{2^n}$ which proves the result. □

7.4 Proof of Lemma 4

Recall that $\Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}] \le \sum_i \Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}^i]$. Therefore, to bound ϵ_{bad}, we separately bound $\Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}^i]$ for each i.

It is easy to see that $\Pr[X_{\mathrm{bad}} \in \Theta_{\mathrm{bad}}^1] \le \frac{q_m^2}{2^{2n}}$ as the distrbution of r and the hash value h is independent in ideal world and we assume $\epsilon \le \frac{1}{2^n}$.
Moreover, we have $\Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}^3] \le \Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}^3 \wedge X_{\mathrm{id}} \notin \Theta_{\mathrm{bad}}^2 \wedge X_{\mathrm{id}} \notin \Theta_{\mathrm{bad}}^1]$ and it is also easy to observe that

$$
\Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}^3 \wedge X_{\mathrm{id}} \notin \Theta_{\mathrm{bad}}^2 \wedge X_{\mathrm{id}} \notin \Theta_{\mathrm{bad}}^1] \le \sum_{i,j,k,l} \Pr[r_i = r_j = r_k = r_l] \le \frac{q_m^4}{2^{3n}},
$$

as r_i are distributed uniformly and independent.

Proposition 1. $\Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}^2] \le \Pr[X_{\mathrm{id}} \in \Theta_{\mathrm{bad}}^2 \wedge X_{\mathrm{id}} \notin \Theta_{\mathrm{bad}}^1] \le \frac{q_m^4}{2^{3n}}$.

Proof. We recall that Θ^2_{bad} is the set of all transcripts such that $\exists i,j,k,l$ such that $r_i = r_j, r_k = r_l, h_j \oplus h_k = r_j \oplus r_k$. Therefore,

$$\Pr[X_{\text{id}} \in \Theta^2_{\text{bad}} \wedge X_{\text{id}} \notin \Theta^1_{\text{bad}}] \leq \sum_{i,j,k,l} \Pr[r_i = r_j, r_k = r_l, h_j \oplus h_k = r_j \oplus r_k]$$

$$\leq \sum_{i,j,k,l} \Pr[h_j \oplus h_k = r_j \oplus r_k | r_i = r_j, r_k = r_l] \cdot \Pr[r_i = r_j, r_k = r_l]$$

Since r_i's are distributed uniformly and independently, $\Pr[r_i = r_j, r_k = r_l] = \Pr[r_i = r_j] \cdot \Pr[r_k = r_l] \leq \frac{1}{2^{2n}}$. Moreover, we can write $\Pr[h_j \oplus h_k = r_j \oplus r_k | r_i = r_j, r_k = r_l]$ as $\Pr[h_j \oplus h_k = r_i \oplus r_l]$. Note that the distribution of h_j and h_k is not affected by r_i and r_l and thus $\Pr[h_j \oplus h_k = r_i \oplus r_l] \leq \frac{1}{2^n}$ as H is a ϵ-AXU hash function and we assume $\epsilon \leq \frac{1}{2^n}$. Therefore, $\Pr[X_{\text{id}} \in \Theta^2_{\text{bad}} \wedge X_{\text{id}} \notin \Theta^1_{\text{bad}}] \leq \sum_{i,j,k,l} \frac{1}{2^{3n}} \leq \frac{q_m^4}{2^{3n}}$. □

Proposition 2. $\Pr[X_{\text{id}} \in \Theta^4_{\text{bad}}] \leq \Pr[X_{\text{id}} \in \Theta^4_{\text{bad}} \wedge (\wedge_{j<4} X_{\text{id}} \notin \Theta^j_{\text{bad}})] \leq \frac{q_m^4}{2^{3n}}$

Proof. To bound $\Pr[X_{\text{id}} \in \Theta^4_{\text{bad}} \wedge (\wedge_{j<4} X_{\text{id}} \notin \Theta^j_{\text{bad}})]$, is equivalent to bound $\Pr[\mathsf{mc}\{t_i + t_j : r_i = r_j\} \geq 4]$. To compute this probability we need to select four pair of distinct indices: $(i_1, i_2), \ldots, (i_7, i_8)$ such that we have the following linear equations : (1) $T_{i_1} \oplus T_{i_2} = T_{i_3} \oplus T_{i_4}$, (2) $T_{i_3} \oplus T_{i_4} = T_{i_5} \oplus T_{i_6}$ and (3) $T_{i_5} \oplus T_{i_6} = T_{i_7} \oplus T_{i_8}$ and four additional linearly independent restrictions on r, i.e. $r_{i_1} = r_{i_2}, r_{i_3} = r_{i_4}, r_{i_5} = r_{i_6}$ and $r_{i_7} = r_{i_8}$. It is easy to see that the equations involving T_i's can not have rank one. So we have at least 6 independent events each of which holds with probability $\frac{1}{2^n}$. Therefore, $\Pr[X_{\text{id}} \in \Theta^4_{\text{bad}} \wedge (\wedge_{j<4} X_{\text{id}} \notin \Theta^j_{\text{bad}})] \leq \frac{q_m^8}{2^{6n}} \leq \frac{q_m^4}{2^{3n}}$ when $q \leq 2^n$. □

Proposition 3. $\Pr[X_{\text{id}} \in \Theta^5_{\text{bad}}] \leq \Pr[X_{\text{id}} \in \Theta^5_{\text{bad}} \wedge X_{\text{id}} \notin \Theta^4_{\text{bad}} \wedge X_{\text{id}} \notin \Theta^3_{\text{bad}}] \leq \frac{9q_v}{2^n}$

Proof. To compute $\Pr[X_{\text{id}} \in \Theta^5_{\text{bad}} \wedge X_{\text{id}} \notin \Theta^4_{\text{bad}}]$, we need $\sum_{i,j,k,l} \Pr[T'_i \oplus T_j \oplus T_k \oplus T_l = 0, r_k = r_l, r'_i = r_j, h_k \oplus h_j = r_j \oplus r_k]$. Now we fix an index i and define a set $\mathcal{S} = \{(j,k,l) : r'_i = r_j, T'_i \oplus T_j \in \{T_k \oplus T_l : r_k = r_l\}\}$. Now we claim that $|\mathcal{S}| \leq 9$. To argue this, for a fix j, $T'_i \oplus T_j$ becomes fixed. Moreover $T'_i \oplus T_j \in \{T_k \oplus T_l : r_k = r_l\}$ implies at most three elements are there in the set $\{T_k \oplus T_l : r_k = r_l\}$ as we are considering the event $X_{\text{id}} \in \Theta^5_{\text{bad}} \wedge X_{\text{id}} \notin \Theta^4_{\text{bad}}$.

Moreover, r'_i can collide to at most three elements which implies that for a fix i, there exists at most three j such that r'_i collides to all these three r_j's. This justifies the maximum size of set $\mathcal{S}$ to be 9. Moreover, there are q_v many i's such that this happens. Now for each such set $\mathcal{S}$ whose size is at most 9, the event $h_k \oplus h_j = r_j \oplus r_k$ holds with probability $\frac{1}{2^n}$ (as we assume $\epsilon \leq \frac{1}{2^n}$). Hence our result follows. □

Corollary 2. *Let f_{k_1} and f_{k_2} be two independently keyed functions and $H : \{0,1\}^* \to \{0,1\}^n$ be a ϵ-AXU hash function. Then we have*

1. $\mathbf{Adv}^{SUF}_{\mathrm{C6}}(q_m, q_v, \ell, t) \leq \mathbf{Adv}^{\mathrm{prf}}_{f_{k_1}}(q_m+q_v, t')+\mathbf{Adv}^{\mathrm{prf}}_{f_{k_2}}(q_m+q_v, t')+\frac{q_m^4}{2^{3n}}+\frac{10q_v}{2^n}$.
2. $\mathbf{Adv}^{pprf}_{\mathrm{C6}}(q_m, \ell, t) \leq \mathbf{Adv}^{\mathrm{prf}}_{f_{k_1}}(q_m, t') + \mathbf{Adv}^{\mathrm{prf}}_{f_{k_2}}(q_m, t') + \frac{q_m^4}{2^{3n}}$.

where $t = t' + O(qT_h)$

The proof of the first corollary follows from Theorem 9 and Lemma 1 and the proof of the second corollary follows the same argument as that in proof of Theorem 9 except that we skip the bad condition (C.5).

8 Conclusion

In this paper we revisit different hash then mask paradigm probabilistic MAC constructions. We study different security notions (PRF/ pPRF/ MAC) of these constructions and show a tight security analysis for each of them. In particular we study the Enhanced Hash then Mask probabilistic MAC construction and lifts its security to $\Theta(2^{3n/4})$, which is shown to be tight and better than the previous non-tight security bound of $O(2^{2n/3})$. We have also studied that unlike random function in deterministic MAC, there is no idealized version of unforgeable security of a probabilistic MAC. We also have introduced a new security notion pPRF which is used to prove the impossibility result of probabilistic MAC, but it does not imply secure probabilistic MAC in general and we have not studied its practical application in this paper.

Acknowledgement: The authors are thankful to all the anonymous reviewers of ASIACRYPT, 2016 for their useful comments towards this research work.

References

1. William Aiello and Ramarathnam Venkatesan. Foiling birthday attacks in length-doubling transformations - benes: A non-reversible alternative to feistel. In *Advances in Cryptology - EUROCRYPT '96*, pages 307–320, 1996.
2. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In *Advances in Cryptology - CRYPTO '96*, pages 1–15, 1996.
3. Mihir Bellare, Oded Goldreich, and Hugo Krawczyk. Stateless evaluation of pseudorandom functions: Security beyond the birthday barrier. In *Advances in Cryptology - CRYPTO '99*, pages 270–287, 1999.
4. Mihir Bellare, Roch Guérin, and Phillip Rogaway. XOR macs: New methods for message authentication using finite pseudorandom functions. In *Advances in Cryptology - CRYPTO '95*, pages 15–28, 1995.
5. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of cipher block chaining. In *Advances in Cryptology - CRYPTO '94*, pages 341–358, 1994.
6. Daniel J. Bernstein. How to stretch random functions: The security of protected counter sums. *J. Cryptology*, 12(3):185–192, 1999.
7. Daniel J. Bernstein. The poly1305-aes message-authentication code. In *Fast Software Encryption, FSE 2005*, pages 32–49, 2005.
8. John Black and Martin Cochran. MAC reforgeability. In *Fast Software Encryption, FSE 2009*, pages 345–362, 2009.

9. John Black, Shai Halevi, Hugo Krawczyk, Ted Krovetz, and Phillip Rogaway. UMAC: Fast and Secure Message Authentication. In *Advances in Cryptology - CRYPTO '99*, pages 216–233, 1999.
10. John Black and Phillip Rogaway. A block-cipher mode of operation for parallelizable message authentication. In *Advances in Cryptology - EUROCRYPT 2002*, pages 384–397, 2002.
11. Larry Carter and Mark N. Wegman. Universal classes of hash functions. *J. Comput. Syst. Sci.*, 18(2):143–154, 1979.
12. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In *Advances in Cryptology, Proceedings of CRYPTO '84*, pages 276–288, 1984.
13. Shai Halevi and Hugo Krawczyk. MMH: Software Message Authentication in the Gbit/Second Rates. In *Fast Software Encryption, FSE '97*, pages 172–189, 1997.
14. Éliane Jaulmes, Antoine Joux, and Frédéric Valette. On the security of randomized CBC-MAC beyond the birthday paradox limit: A new construction. In *Fast Software Encryption, FSE 2002*, pages 237–251, 2002.
15. Éliane Jaulmes and Reynald Lercier. Frmac, a fast randomized message authentication code. *IACR Cryptology ePrint Archive*, 2004:166, 2004.
16. Ted Krovetz. Message Authentication on 64-bit Architectures. *IACR Cryptology ePrint Archive*, 2006:37, 2006.
17. Atul Luykx, Bart Preneel, Elmar Tischhauser, and Kan Yasuda. A MAC mode for lightweight block ciphers. *IACR Cryptology ePrint Archive*, 2016:190, 2016.
18. Kazuhiko Minematsu. How to thwart birthday attacks against macs via small randomness. In *Fast Software Encryption, FSE 2010*, pages 230–249, 2010.
19. Kazuhiko Minematsu and Toshiyasu Matsushima. New Bounds for PMAC, TMAC, and XCBC. In *Fast Software Encryption, FSE 2007*, pages 434–451, 2007.
20. Kazuhiko Minematsu and Yukiyasu Tsunoo. Expanding weak PRF with small key size. In *Information Security and Cryptology - ICISC 2005*, pages 284–298, 2005.
21. Jacques Patarin. *Etude des Générateurs de Permutations Pseudo-aléatoires Basés sur le Schéma du DES*. PhD thesis, Université de Paris 6, 1991.
22. Jacques Patarin. A proof of security in $o(2^n)$ for the benes scheme. In *Progress in Cryptology - AFRICACRYPT 2008*, pages 209–220, 2008.
23. Jacques Patarin. The "Coefficients H" Technique. In *Selected Areas in Cryptography, SAC*, pages 328–345, 2008.
24. Phillip Rogaway. Bucket Hashing and Its Application to Fast Message Authentication. *J. Cryptology*, 12(2):91–115, 1999.
25. Palash Sarkar. A New Multi-Linear Universal Hash Family. *Des. Codes Cryptography*, 69(3):351–367, 2013.
26. Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004:332, 2004.
27. Serge Vaudenay. Decorrelation: A Theory for Block Cipher Security. *J. Cryptology*, 16(4):249–286, 2003.
28. Shmuel Winograd. A New Algorithm for Inner Product. *IEEE Trans. Computers*, 17(7):693–694, 1968.

APPENDIX

A Reduction from SUF to cTG

Theorem A.1 For any construction cTG_k, let TG denote the coin-explicit tag generation algorithm. Let Π be the probabilistic the complete MAC correspond-

ing to this tag generation algorithm. Then,

$$\mathbf{Adv}^{\mathrm{SUF}}_{\Pi}(q_m, q_v, t) \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{cTG}}(q_m + q_v, t') + \frac{q_v}{2^n},$$

where $t' \approx t$.

Proof. Let $\mathcal{A}$ be an adversary that asks at most q_m many tag generation queries and q_v many verification queries and breaks the SUF security of MAC Π. We construct an adversary $\mathcal{B}$ that asks total $q_m + q_v$ many queries to its oracle such that it breaks the prf security of core-tag generation. We define the game as follows : $\mathcal{B}$ will simulate the challenger for $\mathcal{A}$. When $\mathcal{A}$ asks tag generation query m, $\mathcal{B}$ first samples r uniformly at random from $\mathcal{R}$ and then submits (r, m) to its oracle $\mathcal{O}$. $\mathcal{A}$ will obtain $\mathcal{O}(r, m)$. For a verification query $(\widetilde{r}, \widetilde{m}, \widetilde{t})$ by $\mathcal{A}$, $\mathcal{B}$ will ask the oracle $\mathcal{O}$ with $(\widetilde{r}, \widetilde{m})$ and then checks whether $\widetilde{t} = \mathcal{O}(\widetilde{r}, \widetilde{m})$. If they match then $\mathcal{B}$ stops and returns 1. If $\mathcal{A}$ is executed completely, $\mathcal{B}$ will return 0. Therefore,

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{cTG}}(\mathcal{B}) = \Pr[\mathcal{B}^{\mathsf{cTG}_k(\cdot,\cdot)} = 1] - \Pr[\mathcal{B}^{\mathsf{RF}} = 1].$$

Note that $\Pr[\mathcal{B}^{\mathsf{cTG}_k(\cdot,\cdot)} = 1] \geq \mathbf{Adv}^{\mathrm{SUF}}_{\Pi}(q_m, q_v, t)$ and $\Pr[\mathcal{B}^{\mathsf{cTG}_k(\cdot,\cdot)} = 1] = \frac{q_v}{2^n}$. Hence the result follows. □