The MAGIC Mode for Simultaneously Supporting Encryption, Message Authentication and Error Correction

Michael Kounavis[1], David Durham[1], Sergej Deutsch[1], Krystian Matusiewicz[2] and David Wheeler[3]

[1]Intel Labs, Intel Corporation
[2]Intel Security Architecture and Engineering, Intel Corporation
[3]Amazon Devices, Digital Security Team

{michael.e.kounavis, david.durham, sergej.deutsch, krystian.matusiewicz}@intel.com, davewhee@amazon.com

December 2, 2020

Abstract

We present MAGIC, a mode for authenticated encryption that simultaneously supports encryption, message authentication and error correction, all with the same code. In MAGIC, the same code employed for cryptographic integrity is also the parity used for error correction. To correct errors, MAGIC employs the Galois Hash transformation, which due to its bit linearity can perform corrections in a similar way as other codes do (e.g., Reed Solomon). To provide a cryptographically strong MAC, MAGIC encrypts the output of the Galois Hash using a secret key. To analyze the security of this construction we adapt the definition of the MAC adversary so that it is applicable to systems that combine message authentication with error correction. We demonstrate that MAGIC offers security in the order of $O(2^{N/2})$ with N being the tag size.

1 Introduction

In the paper, we introduce MAGIC, a mode for authenticated encryption that simultaneously supports encryption, message authentication and error correction. In MAGIC, the same message authentication code employed for cryptographic integrity is also the parity value used for error correction. Informally, by "error correction" we mean, as in standard ECC, altering a ciphertext-authentication tag pair, which fails a validation test, in such a way so that the resulting pair passes the same validation test.

To accomplish combined message authentication and error correction, our mode employs the well known Galois Hash transformation [5]. Galois Hash is bit linear, provided that the hash key is known. We demonstrate that due to its bit linearity, Galois Hash can correct errors in a similar manner as other codes do (e.g., Reed Solomon [8]). A novel element we introduce in our error correction process is that of using a Hamming weight test to identify where errors are located. Locating errors via the Hamming weight test makes forming error locator polynomials or finding their roots redundant.

In the paper, we also demonstrate that we can construct a cryptographically strong MAC by encrypting the output of the Galois Hash transformation using a secret key. We call such encryption operation "blinding", and the produced construction "blinded Galois Hash". The concept of encrypting the output of a polynomial transformation is known [25, 26]. In the paper we analyze the security of such construction and demonstrate that our MAC provides security against a MAC adversary [23] which is in the order of $O(2^{N/2})$ with N being the tag size. For our security analysis, we adapt the definition of a MAC adversary in order to take into account the fact that the attacked oracle also performs error correction on its inputs. MAGIC stands for "Message Authentication, Galois Integrity and Correction".

1.1 On combining integrity and error correction

Providing data integrity, while at the same time being able to correct errors is challenging. Much of the challenge is associated with the need to store, access and process security metadata. Supporting message authentication is always needed for security, since it is effective in defending against a range of data corruption attacks. Error correction, on the other hand, is also needed as computing systems need to be able to recover from crashes or other inadvertent changes in their data. Traditionally, error correction and message authentication are two areas that have been researched separately. Indeed the best known methods for supporting those differ substantially [1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 23, 24].

In this paper, we investigate a solution that simultaneously support cryptographically strong integrity and error correction, all with the same code, thus substantially reducing the associated metadata cost. Our work is rooted in the principles behind the design of the GCM/GMAC mode of operation [5], as well as the Reed Solomon codes [8, 9, 10]. We believe its impact is significant, as billions of dollars are required by companies, today, to buy storage devices or memory chips that separately store ECC and MAC codes.

1.2 Related work

One of the earliest proposals for simultaneous support for message authentication and error correction, all with the same code is in [13]. Reference [13] proposes cryptographic CRC codes, which are regular CRCs XOR-ed with a one-time pad. This work suffers from the complexity associated with generating, transmitting and keeping secret a one-time pad. A similar proposal that uses a one-time pad is analyzed in [17].

Next, the NTMAC [14], CRC-NTMAC [15] and BCH-NTMAC [16] proposals support message authentication with some error correcting capability at the expense of the code length. To effectively locate and correct errors, these proposals employ arrays of short MACs, each authenticating a different permutation of some bits of the message. The authors demonstrate that the more the arrays of the employed MACs are, the stronger the resulting error correcting capability is. In contrast, MAGIC employs a single code only, which also serves as error correction parity.

A proposal, which is closer to MAGIC, is in reference [18]. In this proposal, error correction is supported by a class of codes called "additive cellular automata". Additive cellular automata are bit-linear codes like Galois Hash, but are more expensive when used for locating and correcting errors. In the approach of reference [18], multiple blocks of redundancy are required in order to locate and correct errors in a single message block. In contrast, MAGIC adds redundancy of a single block only, which acts as both MAC and ECC parity. To accomplish this, MAGIC employs a Hamming weight test, which is a unique feature of its design. MAGIC also differs from [18] with respect to the way cryptographic integrity is supported. The authors of [18] propose to pass the error correcting code output through a custom non-linear permutation called Nmix(). In this paper, we propose to pass the error correcting code output through a pseudo-random permutation which stands for any well designed block cipher.

Other references [19, 20, 27, 28, 29, 30, 31, 32, 33] address similar problems as this work, but not exactly the same. References [19, 20] address the problem of efficiently co-locating MACs and ECC parity values in the same storage area, without necessarily making them the same code. Reference [19] presents a scheme where ECC bits are efficiently interleaved with MAC bits, whereas reference [20] presents a scheme where MAC and ECC bits are efficiently co-located on the same ECC-DIMMs. Furthermore, a large body of work [27, 28, 29, 30, 31, 32, 33] studies the related problem of designing authentication codes using error correcting codes as mathematical primitives. Typically, these references do not address the simultaneous use of codes for both authentication and error correction. Their goal is to primarily support authentication, but do it in such a way, so that the primitives used for constructing codes are ECC primitives. Many systems have been proposed, the oldest and best known perhaps being McEliece [27].

1.3 Security definitions when introducing error correction

For a classic MAC construction the usual security model is as follows. A MAC scheme Σ is a pair of algorithm families $\Sigma = \{\mathsf{MAC}, \mathsf{Verify}\}$ parameterized by a key $K \in \mathcal{K}$.

The adversary $\mathcal{A}$ is given query access to a randomized MAC signing algorithm $\mathsf{MAC}_K(M)$ that, for a secret fixed key K, accepts a message M and returns an authentication tag $T = \mathsf{MAC}_K(M)$. It also has access to verification oracle queries $\mathsf{Verify}_K(M,T)$ that return either accept or reject.

We refer to the set of queries asked to the signing oracle as $Q_s = \{M_i : i = 1, \ldots, |Q_s|\}$. We allow the adversary to submit $|Q_v|$ verification oracle queries $Q_v = \{(M_i, T_i) : i = 1, \ldots, |Q_v|\}$. We refer to the set of all queries as $Q = Q_s \cup Q_v$ and the total number of queries as $|Q| = |Q_s| + |Q_v|$.

We say that the adversary algorithm $\mathcal{A}$ wins the forgery game if it can come up with a pair M, T such that the pair has not been queried to the signing oracle before, i.e, $M \notin Q_s$, and the verification query returns $\mathsf{Verify}_K(M,T) = \mathsf{accept}$.

We define the advantage of the adversary as the probability that the adversary successfully creates such a message:

$$\begin{gathered}\mathbf{Adv}[\mathcal{A},\Sigma] = \mathrm{Prob}[K \overset{\$}{\leftarrow} \mathcal{K} : (M,T) \leftarrow \mathcal{A}, M \notin Q_s, \\ \mathsf{Verify}_K(M,T) = \mathsf{accept}]\end{gathered} \tag{1}$$

We further say that a MAC scheme Σ is secure if the maximum advantage $\mathbf{Adv}[\mathcal{A},\Sigma]$ taken over efficient adversaries $\mathcal{A}$ is negligible, where the term "negligible" is defined with respect to some specific asymptotic behavior. Now, let us consider what happens when the verification algorithm is preceded by an error correction algorithm $\mathsf{ECC}()$ that corrects up to d single bit errors in the (M,T) pair passed to the verification algorithm.

In the classic model described above, definition (1) immediately yields trivial forgeries. For example, if (M,T) is a pair, where T is obtained from the signing oracle on input M, and δ is a bit pattern of Hamming weight not exceeding d, T will also be a valid authentication tag for $M+\delta$. In this case, an adversary can trivially construct forgeries of the form $(M+\delta, T)$ by simply adding error patterns of bounded Hamming weight to message M.

To avoid such forgeries, we modify the security definition for the MAC adversary. For each message tag pair (M,T), we define the set $\theta^{\mathsf{ECC}()}$ of all message tag pairs which, after error correction performed by $\mathsf{ECC}()$, become (M,T):

$$\theta^{\mathsf{ECC}()}(M,T) = \{(M',T') : \mathsf{ECC}(M',T') = (M,T)\} \tag{2}$$

We further define a set of "excluded values" $\Theta^{\mathsf{ECC}()}(Q_s)$, associated with a set of sign queries Q_s, as the of all message-tag pairs which are "correctable" to one of the signing queries that has been issued:

$$\Theta^{\mathsf{ECC}()}(Q_s) = \bigcup_{M \in Q_s} \theta^{\mathsf{ECC}()}(M, \mathsf{MAC}_K(M)) \tag{3}$$

Now, the advantage of the adversary can be defined as:

$$\begin{aligned} \mathbf{Adv}[\mathcal{A}, \Sigma^{\mathsf{ECC}()}] = \text{Prob}[K \overset{\$}{\leftarrow} \mathcal{K}; (M,T) \leftarrow \mathcal{A}; (M,T) \notin \Theta^{\mathsf{ECC}()}(Q_s), \\ \mathsf{Verify}_K(M,T) = \mathsf{accept}] \end{aligned} \tag{4}$$

Definition (4) requires from the adversary to produce a forgery which is, not only a valid message-tag pair, but also not correctable to any known past query. In this way, definition (4) resolves the trivial forgery issue of definition (1). We note that this is a straightforward extension of the standard definition. If $\mathsf{ECC}()$ is the identity error correction function (i.e., it corrects no errors), we get that $\Theta^{\mathsf{ECC}()}(Q_s) = Q_s$ and the above definition reduces to the standard one.

2 Description of MAGIC

MAGIC is a MAC construction based on universal hashing with additional encryption of the tag, similar to Encrypted Carter-Wegman hash [21, 22]. We denote its block size by N. Both multiplication and addition of bit strings from $\{0,1\}^N$ are defined as finite field operations in $\mathbb{F}_{2^N}$.

The construction operates on an N-bit long hash key value H, which is random uniformly distributed in a set $\mathcal{H}$, $\mathcal{H} \subseteq \{0,1\}^N$. It also employs a plaintext encrypting cipher associated with mode $\mathcal{M}_e$ (e.g., $\mathcal{M}_e$ could be the CBC or XTS mode), key K_e of length l_{K_e}, $K_e \in \{0,1\}^{l_{K_e}}$, initialization vector i_e of length l_{i_e}, $i_e \in \{0,1\}^{l_{i_e}}$ and block size N, which is denoted by $\mathsf{E}_{K_e,\mathcal{M}_e,i_e,N}()$, or, for the sake of simplicity just $\mathsf{E}_{K_e}()$. The purpose of this block cipher is to encrypt the input message blocks, so as to provide confidentiality.

The construction further employs a blinding block cipher associated with mode $\mathcal{M}_B$, key K_B of length l_{K_B}, where $K_B \in \{0,1\}^{l_{K_B}}$, $K_B \neq K_e$, initialization vector i_B of length l_{i_B}, where $i_B \in \{0,1\}^{l_{i_B}}$, $i_B \neq i_e$ and block size N, which is denoted by $\mathsf{E}_{K_B,\mathcal{M}_B,i_B,N}()$, or just $\mathsf{E}_{K_B}()$. The second cipher is used for encrypting a single block only. It encrypts the output of MAGIC's Galois Hash transformation, so as to prevent adversaries from trivially computing its Galois Hash key value H.

Both the encryption and blinding ciphers are considered to be pseudorandom permutations (PRPs). For the blinding cipher, we use the notation $\mathbf{Adv}^{\mathsf{E}_{K_B}()}$ to refer to the advantage of distinguishing $\mathsf{E}_{K_B}()$ from a permutation sampled uniformly from the space $\mathcal{P}$ of all $(2^N)!$ permutations $\pi : \{0,1\}^N \to \{0,1\}^N$.

A concrete instance is defined for $N = 128$, with the finite field defined by the irreducible polynomial $x^{128} + x^7 + x^2 + x + 1$, and the two block ciphers being AES-128 [6] in the XTS mode [7], using different keys and initialization vectors though. We also note that the analysis that follows is applicable to every possible binary representation of the elements of the finite field $\mathbb{F}_{2^N}$, with respect to the order of bits and bytes.

2.1 Tag generation

MAGIC operates on a fixed number n of full message blocks. For the generation of the tag, the message M to be authenticated is first split into blocks $M_1, \ldots, M_n$. Let D denote a single block of additional authenticated data, which are not encrypted. The tag generation proceeds by computing:

$$(C_1, C_2, \ldots, C_n) \leftarrow \mathsf{E}_{K_e}(M_1, M_2, \ldots, M_n) \tag{5}$$

and

$$T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()} \leftarrow \mathsf{E}_{K_B}(D + C_1 \cdot H + \ldots + C_n \cdot H^n) \tag{6}$$

Finally, the output of the authenticated encryption of message $(M_1, M_2, \ldots,$ $M_n)$, when combined with an authenticated data block D is defined by:

$$\mathrm{MAGIC}_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()}(D, M_1, M_2, \ldots, M_n) \leftarrow (D, C_1, C_2, \ldots, C_n, T) \tag{7}$$

The additional authenticated data D carry information specific to particular instantiations of MAGIC. For example, in one instantiation, D may represent a memory address where ciphertext blocks $C_1, \ldots, C_n$ are stored. In another instantiation, D may represent an anti-replay invocation counter protected by a Merkle hash tree. In the analysis that follows it is assumed that D can take any value in the set $\{0, 1\}^N$, and the analysis does not depend on the way D is defined by a MAGIC instantiation.

2.2 Error correction and tag verification

The error correction and tag verification functionality of MAGIC is given by procedures $\mathsf{LocateError}()$, $\mathsf{CanCorrectParity}()$ and $\mathsf{Verify}()$ below. In the code below, the binary representation of a set of concatenated elements is denoted by the $\mathsf{Str}()$ operator. Procedure $\mathsf{Verify}()$ is the verification oracle of MAGIC. MAGIC can correct errors which are present in a single block. Such block can either be a ciphertext block from among $C_1, \ldots, C_n$ or the tag block T. In order for the correction to be successful, the error vector should have Hamming weight less than or equal to a threshold value T_{th}. In the pseudocode below, $\mathsf{HW}()$ is the Hamming weight function.

Procedure Verify() returns not only an accept/reject boolean but also two strings. When no error correction takes place, these strings are empty. When error correction takes place, the returned strings are equal to the corrected values of the ciphertext and tag respectively.

These are the steps followed by Verify(). Initially, procedure Verify() checks whether the supplied tag value T is the correct MAC computed from the supplied ciphtertext blocks $C_1, \ldots, C_n$ and authenticated data D. If this is the case it returns (accept, $\perp$, $\perp$).

If this is not the case, procedure Verify() attempts to correct errors in one of the supplied ciphertext blocks. This is done in lines 4-10 of the pseudocode. It first computes a syndrome value S. The syndrome is equal to the Galois Hash output computed from the ciphertext blocks and authenticated data, plus the result of the decryption of the supplied tag T. The decryption is done using the blinding cipher.

```
LocateError($S_1, S_2, \ldots, S_n, T_{th}$)
1. if $\exists\ i_{err} \in [1, n] : \mathsf{HW}(S_{i_{err}}) \leq T_{th}$
2.     if $\forall i \neq i_{err}, i \in [1, n] : \mathsf{HW}(S_i) > T_{th}$
3.         return $i_{err}$
4. else
5.     return $\perp$

CanCorrectParity($T, T', T_{th}$)
1. if $\mathsf{HW}(T + T') \leq T_{th}$
2.     return true
3. else
4.     return false

Verify$_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()}$($D, C_1, \ldots, C_n, T, T_{th}$)
1. if $T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n)$
2.     return (accept, $\perp$, $\perp$)
3. else
4.     $S \leftarrow D + C_1 \cdot H + \ldots + C_n \cdot H^n + \mathsf{E}_{K_B}^{-1}(T)$
5.     for $i \leftarrow 1$ to $n$
6.         do $S_i \leftarrow S \cdot H^{-i}$
7.     $i_{err} \leftarrow$ LocateError($S_1, S_2, \ldots, S_n, T_{th}$)
8.     if $i_{err} \neq \perp$
9.         return (accept, Str($C_1, \ldots, C_{i_{err}} + S_{i_{err}}, \ldots, C_n$), $T$)
10.    else
11.        $T' \leftarrow \mathsf{E}_{K_B}(D + C_1 \cdot H + \ldots + C_n \cdot H^n)$
12.        if CanCorrectParity($T, T', T_{th}$) = true
13.            return (accept, Str($C_1, \ldots, C_n$), $T'$)
14.        else
15.            return (reject, $\perp$, $\perp$)
```

Procedure Verify() also computes error location indicator values S_i, one for each input block C_i, $i \in [1, n]$. The computation, done in lines 5-6 of the pseudocode, involves multiplying the syndrome S with the inverse of power H^i of the hash key value H. The error location indicators are computed in such a way, so that, if there is a single error vector e of length N corrupting block $C_{i_{err}}$, $i_{err} \in [1, n]$, then for this location the indicator $S_{i_{err}}$ should be equal to e. For all other locations, the indicators $S_i, i \neq i_{err}$ should be equal to $e \cdot H^{i_{err}-i}$. This property of the indicator values S_i helps with establishing the error correcting capability of MAGIC, as discussed in Section 2.4.

To determine the index of a block which may be corrupted, procedure Verify() invokes procedure LocateError(). Procedure LocateError() employs a Hamming weight test. LocateError() operates on the assumption that the Hamming weight of indicator S_i is lower than a given threshold T_{th} only if index i is the location where the error occurs, i.e., $i = i_{err}$. The location of a corrupted block is identified by computing the Hamming weight of all error location indicators, and by checking whether bounded Hamming weight appears at a single location only. Once an error is located, the error is corrected by setting $C_{i_{err}} \leftarrow C_{i_{err}} + S_{i_{err}}$.

If the location of a single corrupted block cannot be identified, procedure Verify() checks whether correctable errors are present in the supplied tag T. This is done by invoking procedure CanCorrectParity(). If errors can be corrected in the supplied tag, then Verify() returns accept and a pair of strings consisting of the supplied ciphertext blocks and the corrected tag.

If the location of a single corrupted block cannot be identified neither in the ciphertext blocks nor in the tag, Verify() returns reject and the empty strings. This may happen, for instance, if multiple blocks are corrupted.

2.3 The sets of hash keys $\mathcal{H}$ and excluded hash keys $\mathcal{H}_{\mathcal{E}}$

The intuition behind the Hamming weight test of LocateError() is that all error vectors, which can be corrected, have by definition Hamming weight bounded by T_{th}. If only a single block is corrupted, then $S_{i_{err}} = e$, as we establish in Section (2.4).

To locate errors without ambiguity, we specify the set of Galois Hash key values $\mathcal{H}$ to include only those hash keys, the powers of which if multiplied with bounded Hamming weight values return products with Hamming weight greater than T_{th}.

Definition 1: The set $\mathcal{H} \subseteq \{0,1\}^N$ from which the hash keys values H of MAGIC are drawn is defined by:

$$\mathcal{H} = \{H \in \{0,1\}^N, \; \big(\forall e : e \neq 0, \; \mathsf{HW}(e) \leq T_{th}\big) \wedge \big(\forall i \in [1, n-1]\big) : \\ \mathsf{HW}(e \cdot H^{-i}) > T_{th}, \; \mathsf{HW}(e \cdot H^{i}) > T_{th} \; \} \tag{8}$$

where $T_{th} \in [1, N]$ is the Hamming weight threshold value passed as input to procedure Verify().

Definition 2: The set $\mathcal{H}_{\mathcal{E}} \subseteq \{0,1\}^N$ of excluded hash key values is defined as the set of all values in $\{0,1\}^N$ which are not in $\mathcal{H}$. These are the values which should never be assigned to hash keys of the MAGIC mode:

$$\mathcal{H}_{\mathcal{E}} = \{0,1\}^N - \mathcal{H} \tag{9}$$

Lemma 1: The cardinality of the set $\mathcal{H}_{\mathcal{E}}$ is bounded by:

$$|\mathcal{H}_{\mathcal{E}}| \leq \frac{n(n-1)}{2} \cdot \left(\sum_{t=1}^{T_{th}} \binom{N}{t} \right)^2 \tag{10}$$

Proof of Lemma 1: Let's first consider the set $\mathcal{E}_{\mathcal{T}}$ of all non-zero values $e \in \{0,1\}^N$ with Hamming weight bounded by T_{th}:

$$\mathcal{E}_{\mathcal{T}} = \{e \in \{0,1\}^N,\ e \neq 0,\ \mathsf{HW}(e) \leq T_{th}\} \tag{11}$$

By definition (11) $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$. One subset of hash key values, which set $\mathcal{H}_{\mathcal{E}}$ contains, is the set of all solutions to equation (12) below:

$$e_1 = H \cdot e_2 \tag{12}$$

Equation (12) is defined for all pairs $(e_1, e_2) \in \mathcal{E}_{\mathcal{T}}^2$. Since there can be at most $\sum_{t=1}^{T_{th}} \binom{N}{t}$ values for both e_1 and e_2, the number of hash keys values which are solutions to equation (12) is bounded by $(\sum_{t=1}^{T_{th}} \binom{N}{t})^2$.

Set $\mathcal{H}_{\mathcal{E}}$ also contains those hash key values, which are solutions to equation (13) below. This is similar to (12) but H appears in its inverse form:

$$e_1 = H^{-1} \cdot e_2,\ (e_1, e_2) \in \mathcal{E}_{\mathcal{T}}^2 \tag{13}$$

Due to the symmetry between (12) and (13), the solutions to (13) are also solutions to (12) for $e_1 \leftarrow e_1^{-1}$ and $e_2 \leftarrow e_2^{-1}$. The remaining elements of set $\mathcal{H}_{\mathcal{E}}$ are solutions to equation (14) in which an exponent value u appears, where $u \in [-n+1, -2] \cup [2, n-1]$:

$$e_1 = H^u \cdot e_2,\ (e_1, e_2) \in \mathcal{E}_{\mathcal{T}}^2 \tag{14}$$

With e_1, e_2 and u considered constant, equation (14) is a polynomial equation satisfied by at most $|u|$ roots. Therefore, the number of hash key values satisfying (14) is bounded by $|u| \cdot (\sum_{t=1}^{T_{th}} \binom{N}{t})^2$, for fixed u. These are also the roots of an equation of the form of (14), defined for $e_1 \leftarrow e_1^{-1}$, $e_2 \leftarrow e_2^{-1}$ and $u \leftarrow -u$. We conclude the proof by summing up the bounds for the cardinalities of the sets of all values satisfying equations (12)-(14):

$$|\mathcal{H}_{\mathcal{E}}| \leq \sum_{|u|=1}^{n-1} |u| \cdot \left(\sum_{t=1}^{T_{th}} \binom{N}{t} \right)^2 = \frac{n(n-1)}{2} \cdot \left(\sum_{t=1}^{T_{th}} \binom{N}{t} \right)^2 \tag{15}$$

□

Corollary 1: The cardinality of the set of hash keys $\mathcal{H}$ satisfies:

$$|\mathcal{H}| \geq 2^N - \frac{n(n-1)}{2} \cdot \left(\sum_{t=1}^{T_{th}} \binom{N}{t} \right)^2 \tag{16}$$

The number of hash keys which are excluded typically corresponds to a small fraction of the total number of N-bit values. For example, for $N = 128$, $n = 4$ and $T_{th} = 10$, $|\mathcal{H}_{\mathcal{E}}| \leq 2^{98.213}$. This is out of 2^{128} hash keys. On the other hand, for the same parameters, $|\mathcal{H}| > 2^{128-\delta}$, where $\delta = 0.00000000156$.

2.4 Establishing the error correcting capability of MAGIC

We are now in a position where we can establish the error correcting capability of MAGIC. First, we specify which sets of input ciphertext blocks and tag values are considered "correctable" by the MAGIC mode.

Definition 2: Let $C_1, \ldots, C_n$ be a set of n N-bit values, $C_i \in \{0,1\}^N$, $i \in [1, n]$, which we refer to as "ciphertext blocks". We define the set of "correctable ciphertext blocks" $\theta_{cipher}(C_1, \ldots, C_n, T_{th})$ associated with $C_1, \ldots, C_n$ and threshold value $T_{th} \in [1, N]$ as the set:

$$\begin{aligned} \theta_{cipher}(C_1, \ldots C_n, T_{th}) = \{ & (C'_1, \ldots, C'_n) \in \{0,1\}^{n \cdot N}, \\ & \big(\exists\, e,\ \mathsf{HW}(e) \leq T_{th} \ \wedge\ \exists\, i_{err} \in [1, n] \big) : \\ & \big(C'_{i_{err}} = C_{i_{err}} + e \big) \ \wedge\ \big(\forall\, i\ \in [1, n],\ i \neq i_{err} :\ C'_i = C_i \big) \} \end{aligned} \tag{17}$$

The set of correctable ciphertext blocks $\theta_{cipher}(C_1, \ldots C_n, T_{th})$ is formed from all sets of ciphertext blocks that differ from $(C_1, \ldots, C_n)$ by at most one block value, at location i_{err}. Furthermore, that block value is formed by adding an error vector e to C_{ierr}, where the Hamming weight of e does not exceed T_{th}.

Corollary 2: For every set of ciphertext blocks $(C_1, \ldots, C_n) \in \{0,1\}^{n \cdot N}$ and $T_{th} \in [1, n]$ it holds that:

$$(C_1, \ldots, C_n) \in \theta_{cipher}(C_1, \ldots C_n, T_{th}) \tag{18}$$

Next, we specify which input tags are considered "correctable" by MAGIC:

Definition 3: Let T be an N-bit value, $T \in \{0,1\}^N$, which we refer to as "tag". We define the set of "correctable tags" $\theta_{tag}(T, T_{th})$ associated with T and threshold value $T_{th} \in [1, N]$ as the set:

$$\theta_{tag}(T, T_{th}) = \{T' \in \{0,1\}^N,\ T' = T + e,\ \mathsf{HW}(e) \leq T_{th}\} \tag{19}$$

The following two Lemmas establish the fact that it is not possible to corrupt ciphertext blocks or tags within the sets $\theta_{cipher}()$ and $\theta_{tag}()$, defined above, and still be able to produce ciphertext-tag pairs that pass the verification test of MAGIC without error correction. In other words, it is not possible for non-zero corruptions from $\theta_{cipher}()$ and $\theta_{tag}()$ to return (accept, $\perp$, $\perp$) when passed into procedure $\mathsf{Verify}()$ above.

Lemma 2: For every set of N-bit ciphertext blocks $(C_1, \ldots, C_n) \in \{0,1\}^{n \cdot N}$, every threshold value $T_{th} \in [1, N]$, every set of correctable ciphertext blocks $(C'_1, \ldots, C'_n) \in \theta_{cipher}(C_1, \ldots C_n, T_{th})$ such that $(C'_1, \ldots, C'_n) \neq (C_1, \ldots C_n)$, and every block of authenticated data $D \in \{0,1\}^N$, it holds that:

$$\mathsf{Verify}_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()}(D, C'_1, \ldots, C'_n,\ T,\ T_{th}) \neq (\mathsf{accept}, \perp, \perp) \tag{20}$$

where $T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n)$.

Proof of Lemma 2: In order for procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()}(D, C'_1, \ldots, C'_n, T, T_{th})$ to return (accept, $\perp$, $\perp$), the condition $T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(), \mathsf{E}_{K_B}()}(D, C'_1, \ldots, C'_n)$, which is checked in line 1 of the pseudocode, must be found to be true. Such condition can be written as:

$$\mathsf{E}_{K_B}(D + C_1 \cdot H + \ldots + C_n \cdot H^n) = \mathsf{E}_{K_B}(D + C'_1 \cdot H + \ldots + C'_n \cdot H^n) \tag{21}$$

By applying the decryption operator $\mathsf{E}_{K_B}^{-1}()$ to both sides of the equation and eliminating the term D, we rewrite (21) as:

$$(C_1 + C'_1) \cdot H + \ldots + (C_n + C'_n) \cdot H^n = 0 \tag{22}$$

Since $(C'_1, \ldots, C'_n)$ belongs to the set $\theta_{cipher}(C_1, \ldots C_n, T_{th})$, all ciphertext blocks from $(C'_1, \ldots, C'_n)$ but one are equal to their corresponding blocks from $(C_1, \ldots, C_n)$. Therefore equation (22) can be written as:

$$(C_{i_{err}} + C'_{i_{err}}) \cdot H^{i_{err}} = 0 \tag{23}$$

for some $i_{err} \in [1, n]$.

Since, according to Definition 2, $C_{i_{err}} + C'_{i_{err}} = e$, equation (23) can be further simplified as $e \cdot H^{i_{err}} = 0$, where $e \in \{0,1\}^N$ is the error vector with

Hamming weight $\mathsf{HW}(e) \leq T_{th}$. The latter is only satisfied for $e = 0$. However, this is not possible, as it is assumed that $(C'_1, \ldots, C'_n) \neq (C_1, \ldots, C_n)$. Therefore, the condition in line 1 of procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$ is never true and Lemma 2 holds.

$\square$

Lemma 3: For every block of authenticated data $D \in \{0,1\}^N$, every tag value $T \in \{0,1\}^N$ for which there exists a set of ciphertext blocks $(C_1, \ldots, C_n) \in \{0,1\}^{n \cdot N}$ such that $T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n)$, every threshold value $T_{th} \in [1, N]$ and every correctable tag value $T' \in \theta_{tag}(T, T_{th})$ such that $T' \neq T$, it holds that:

$$\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n,\ T',\ T_{th}) \neq (\mathsf{accept}, \perp, \perp) \tag{24}$$

Proof of Lemma 3: To return $(\mathsf{accept}, \perp, \perp)$, procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$ must determine that the truth value of condition $T' = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n)$ checked in line 1 of the pseudocode is true. This condition is written as:

$$\mathsf{E}_{K_B}(D + C_1 \cdot H + \ldots + C_n \cdot H^n) + e = \mathsf{E}_{K_B}(D + C_1 \cdot H + \ldots + C_n \cdot H^n) \tag{25}$$

where $e \in \{0,1\}^N$ is an error vector with Hamming weight $\mathsf{HW}(e) \leq T_{th}$. Equation (25) is only satisfied for $e = 0$, which is not possible, as $T' \neq T$. Hence Lemma 3 holds.

$\square$

Now that we established that corruptions within the sets $\theta_{cipher}()$ and $\theta_{tag}()$ never pass the verification test of MAGIC without any error correction, we show that, for such corruptions, the verification oracle of MAGIC returns the error-free versions of the corresponding inputs. This is done by applying the error correction steps of lines 4-15 in the pseudocode. For such corruptions the verification oracle returns accept and some non-empty strings indicating that error correction took place.

The next Lemma is about a property of the error location indicators S_i of the pseudocode, which is used for establishing the error correcting capability of MAGIC.

Lemma 4: For every set of N-bit ciphertext blocks $(C_1, \ldots, C_n) \in \{0,1\}^{n \cdot N}$, every threshold value $T_{th} \in [1, N]$, every set of correctable ciphertext blocks $(C'_1, \ldots, C'_n) \in \theta_{cipher}(C_1, \ldots C_n, T_{th})$ such that $(C'_1, \ldots, C'_n) \neq (C_1, \ldots C_n)$, and every block of authenticated data $D \in \{0,1\}^N$, there exists an error vector $e \in \{0,1\}^N$ with Hamming weight $\mathsf{HW}(e) \leq T_{th}$, and an index value $i_{err} \in [1, n]$ such that:

$$\begin{aligned} S_i(D, C'_1, \ldots, C'_n, T, T_{th}) &= e && \text{if } \ i = i_{err} \\ S_i(D, C'_1, \ldots, C'_n, T, T_{th}) &= e \cdot H^{i_{err}-i} && \text{if } \ i \neq i_{err} \end{aligned} \tag{26}$$

where $T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n)$ and $S_i(D, C'_1, \ldots, C'_n, T, T_{th})$, $i \in [1, n]$, are the values of the error location indicators computed in line 6 of procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$, when the procedure accepts as input authenticated data D, ciphertext blocks $C'_1, \ldots, C'_n$, tag T, and threshold value T_{th}. Moreover, i_{err} is the index that identifies the block by which $(C'_1, \ldots, C'_n)$ differ from $(C_1, \ldots, C_n)$ and e is the error vector which is added to $C_{i_{err}}$ to produce $C'_{i_{err}}$.

Proof of Lemma 4: The syndrome value S computed in line 4 of the pseudocode of procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C'_1, \ldots, C'_n, T, T_{th})$ is given by:

$$\begin{aligned} S &= D + C'_1 \cdot H + \ldots + C'_n \cdot H^n + \mathsf{E}_{K_B}^{-1}(T) \\ &= (C_1 + C'_1) \cdot H + \ldots + (C_n + C'_n) \cdot H^n \\ &= (C_{i_{err}} + C'_{i_{err}}) \cdot H^{i_{err}} = e \cdot H^{i_{err}} \end{aligned} \tag{27}$$

where i_{err} is the index that identifies the location where $(C'_1, \ldots, C'_n)$ differ from $(C_1, \ldots, C_n)$ and e is the error vector added to $C_{i_{err}}$. Lemma 4 follows from equation (27) and the fact that each error location indicator S_i results by multiplying syndrome S with the power H^{-i}.

□

We conclude the section with the following two theorems, which establish the error correcting capability of MAGIC:

Theorem 1: *On the ability of* MAGIC *to deterministically correct errors of bounded Hamming weight on a single corrupted ciphertext block.* For every set of N-bit ciphertext blocks $(C_1, \ldots, C_n) \in \{0, 1\}^{n \cdot N}$, every threshold value $T_{th} \in [1, N]$, every set of correctable ciphertext blocks $(C'_1, \ldots, C'_n) \in \theta_{cipher}(C_1, \ldots C_n, T_{th})$ such that $(C'_1, \ldots, C'_n) \neq (C_1, \ldots C_n)$, and every block of authenticated data $D \in \{0, 1\}^N$, it holds that:

$$\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C'_1, \ldots, C'_n,\ T,\ T_{th}) = (\mathsf{accept}, \mathsf{Str}(C_1, \ldots, C_n), T) \tag{28}$$

where $T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n)$ and the hash key value H is random uniformly distributed from the set $\mathcal{H}$, defined by (8).

Proof of Theorem 1: Let $i_{err} \in [1, n]$ be the index of the location where $(C'_1, \ldots, C'_n)$ differ from $(C_1, \ldots, C_n)$ and $e \in \{0, 1\}^N$, with $\mathsf{HW}(e) \leq T_{th}$, the error vector corrupting $C_{i_{err}}$. Lemma 2 establishes that the flow of execution of procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C'_1, \ldots, C'_n, T, T_{th})$ does not exit in line 2 but moves onto line 4. This is done, in order for the procedure to check whether errors on the supplied ciphertext blocks can be corrected.

From line 4, the execution flow moves onto repeating the loop of lines 5-6 n times. These lines compute error location indicator values $S_i, i \in [1, n]$, which satisfy the relation (26). Next, procedure $\mathsf{LocateError}(S_1, \ldots, S_n, T_{th})$ is invoked. Since the Hamming weight of $e = S_{i_{err}}$ is bounded by T_{th}, the condition in line 1 of $\mathsf{LocateError}()$ is found to be true and the flow moves onto line 2. Since the hash key H is drawn from the set $\mathcal{H}$, any product between any non-zero power of H and any non-zero value with Hamming weight bounded by T_{th}, has Hamming weight that exceeds T_{th}. Because of this reason and (26), the condition in line 2 of $\mathsf{LocateError}()$ is found to be true as well. Hence, the flow of $\mathsf{LocateError}()$ proceeds to line 3, where the procedure returns the correct location of the block in error i_{err}.

We conclude the proof by observing that once $\mathsf{LocateError}()$ returns, the flow of execution of procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$ moves onto line 9, where it corrects the existing error $e = S_{i_{err}}$, by adding this error value to block $C'_{i_{err}}$.

$\square$

Theorem 2: *On the ability of* MAGIC *to probabilistically correct errors of bounded Hamming weight on the tag value.* For every block of authenticated data $D \in \{0,1\}^N$, every tag value $T \in \{0,1\}^N$ for which there exists a set of ciphertext blocks $(C_1, \ldots, C_n) \in \{0,1\}^{n \cdot N}$ such that $T = \mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n)$, and every threshold value $T_{th} \in [1, N]$, there exists a subset $\vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \subseteq \theta_{tag}(T, T_{th})$, which depends on the choice of the blinding cipher $\mathsf{E}_{K_B}()$ and hash key value $H \in \mathcal{H}$, such that for every $T' \in \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th})$, $T' \neq T$:

$$\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n, T', T_{th}) = (\mathsf{accept}, \mathsf{Str}(C_1, \ldots, C_n), T) \tag{29}$$

Moreover for fixed T, H drawn uniformly from the set $\mathcal{H}$, key K_B drawn uniformly from $\{0,1\}^{l_{K_B}}$ and initialization vector i_B drawn uniformly from $\{0,1\}^{l_{i_B}}$, the probability that some tag value $T' \in \theta_{tag}(T, T_{th})$ also belongs to the set $\vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th})$ satisfies:

$$\text{Prob}[T' \in \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \mid \mathsf{Cond}] \geq 1 - \frac{(n^2 + n) \cdot |\mathcal{E}_{\mathcal{T}}|}{2^{N+1} - n^2 \cdot |\mathcal{E}_{\mathcal{T}}|^2} - \mathbf{Adv}^{\mathsf{E}_{K_B}()} \tag{30}$$

where Cond is the set of conditions $\mathsf{Cond} = (T \in \{0,1\}^N,\ T_{th} \in \{0,1\}^N,\ T' \in \theta_{tag}(T, T_{th}),\ H \xleftarrow{\$} \mathcal{H},\ K_B \xleftarrow{\$} \{0,1\}^{l_{K_B}},\ i_B \xleftarrow{\$} \{0,1\}^{l_{i_B}})$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$.

Proof of Theorem 2: Lemma 2 establishes that the flow of execution of procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n, T', T_{th})$ does not exit in line 2 but moves onto line 4, where error correction is done. We define $\vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th})$ to be the set of all tag values $T' \in \theta_{tag}(T, T_{th})$, which, if passed as input to $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$ together with authenticated data D, ciphertext blocks $C_1, \ldots, C_n$ and threshold T_{th}, cause procedure $\mathsf{LocateError}()$ invoked by $\mathsf{Verify}()$ to return $\perp$. In other words, set $\vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th})$ contains all those tags from $\theta_{tag}(T, T_{th})$, for which procedure $\mathsf{Verify}()$ correctly determines that a correctable error, if present, is not in one of the ciphertext blocks $C_1, \ldots, C_n$.

With inputs T' from the set $\vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th})$, the flow of execution of procedure $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}(D, C_1, \ldots, C_n,\ T',\ T_{th})$ moves onto line 11 of the pseudocode. In that line, a new tag value is computed, directly from the authenticated data D and ciphertext blocks $C_1, \ldots, C_n$. Since, these are without error, the tag value T computed in line 11 is the correct tag associated input blocks $C_1, \ldots, C_n$ and authenticated data D. Tag T' is the corrupted version of T. Furthermore, since the error vector added to T has Hamming weight bounded by T_{th}, procedure $\mathsf{CanCorrectParity}()$, invoked in line 12 of the pseudocode, returns true. In the next line, procedure $\mathsf{Verify}_H()$ exits returning accept, as well as non-empty strings containing the correct tag value T.

To prove the second part of the theorem, we first write the syndrome value S computed in line 4 of the pseudocode as:

$$S = \mathsf{E}_{K_B}^{-1}(T) + \mathsf{E}_{K_B}^{-1}(T') \tag{31}$$

For a given syndrome value $S \leftarrow S'$, threshold $T_{th} \in \{0,1\}^N$ and hash key H drawn uniformly from the set $\mathcal{H}$, we compute a bound for probability $P_{S'}$ that procedure $\mathsf{LocateError}()$ returns a non-empty response, when accepting as input indicators $S'H^{-1}, \ldots, S'H^{-n}$, and threshold T_{th}. This is the probability that for some given syndrome value S', the flow of execution of procedure $\mathsf{Verify}()$, which calls $\mathsf{LocateError}()$, moves onto executing line 9 of the pseudocode correcting a single block error on its input ciphertext blocks:

$$P_{S'} \leftarrow \text{Prob}[\,\mathsf{LocateError}(S'H^{-1}, \ldots, S'H^{-n}, T_{th}) \neq \perp \mid \mathsf{Cond}_2\,] \tag{32}$$

where $\mathsf{Cond}_2 = (S' \in \{0,1\}^N,\ T_{th} \in \{0,1\}^N,\ H \xleftarrow{\$} \mathcal{H})$.

For a given S' and H drawn uniformly from $\mathcal{H}$, procedure $\mathsf{LocateError}()$ returns some response other than $\perp$, if and only if there exists exponent value $u \in [1, n]$ such that $\mathsf{HW}(S' \cdot H^{-u}) \leq T_{th}$. Indeed, if such exponent value exists, and since $H \in \mathcal{H}$, then for every exponent $v \in [1, n]$, $v \neq u$, it must hold that $\mathsf{HW}(S' \cdot H^{-v}) > T_{th}$. In this case, the flow of $\mathsf{LocateError}()$ executes lines 1-3 of the pseudocode, returning some value $i_{err} \neq \perp$.

$$\begin{aligned} P_{S'} &= \text{Prob}[\sum_{u=1}^{n} \mathsf{HW}(S' \cdot H^{-u}) \leq T_{th} \mid \mathsf{Cond}_2\,] \\ &\leq \sum_{u=1}^{n} u \cdot \frac{\sum_{t=1}^{T_{th}} \binom{N}{t}}{|\mathcal{H}|} = \frac{n(n+1)}{2} \cdot \frac{|\mathcal{E}_{\mathcal{T}}|}{|\mathcal{H}|} \end{aligned} \tag{33}$$

We note that the converse is also true. If $\mathsf{LocateError}()$ returns $i_{err} \neq \perp$, then its flow must be executing lines 1-3.

We proceed with the proof observing that, if in equation (31) the inverse blinding cipher $\mathsf{E}^{-1}_{K_B}()$ is replaced by a randomly chosen permutation π^* from the set $\mathcal{P}$ of all permutations $\pi : \{0,1\}^N \rightarrow \{0,1\}^N$, the returned syndrome value would be non-zero with uniformly bounded probability. Specifically:

$$\text{Prob}[\, S = \pi^*(T) + \pi^*(T') \mid \mathsf{Cond}_3\,] \leq \frac{1}{2^N - 1} \tag{34}$$

where $\mathsf{Cond}_3 = (T \in \{0,1\}^N,\ T' \in \theta_{tag}(T, T_{th}),\ T_{th} \in \{0,1\}^N,\ \ T \neq T',\ \pi^* \overset{\$}{\leftarrow} \mathcal{P})$.

Indeed, for fixed discrete mappings $T \rightarrow v_0$ and $T' \rightarrow v_1$, defined between tag values T, T' and values $v_0, v_1 \in \{0,1\}^N$, there are $(2^N - 2)!$ permutations supporting these mappings. On the other hand, there can be no more than 2^N different pairs of values $(v_0, v_1) \in \{0,1\}^{2N}$ such that $v_0 + v_1 = S$, for some syndrome value $S \in \{0,1\}^N$. Hence, if the inverse blinding cipher $\mathsf{E}^{-1}_{K_B}()$ were replaced by a randomly chosen permutation $\pi^* : \{0,1\}^N \rightarrow \{0,1\}^N$ and T, T' satisfy Cond_3, the probability that $S = \pi^*(T) + \pi^*(T')$ would be bounded by $\frac{2^N \cdot (2^N - 2)!}{(2^N)!} = \frac{1}{2^N - 1}$.

Next, we express the probability that the error in tag T' is not corrected by MAGIC, i.e., $T' \notin \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th})$ given Cond as the probability of procedure $\mathsf{LocateError}()$ returning a non-empty response, when syndrome S takes all possible values S':

$$
\begin{aligned}
&\text{Prob}[T' \notin \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \mid \mathsf{Cond}] \\
&\quad = \sum_{\substack{S' \in \{0,1\}^N, \\ S' \neq 0}} P_{S'} \text{Prob}[\, S' = \mathsf{E}_{K_B}^{-1}(T) + \mathsf{E}_{K_B}^{-1}(T') \mid \mathsf{Cond}_4 \,]
\end{aligned} \tag{35}
$$

where $\mathsf{Cond}_4 = (T \in \{0,1\}^N,\ T' \in \theta_{tag}(T, T_{th}),\ T_{th} \in \{0,1\}^N,\ \ T \neq T',\ K_B \xleftarrow{\$} \{0,1\}^{l_{K_B}},\ i_B \xleftarrow{\$} \{0,1\}^{l_{i_B}})$.

To rewrite equation (35), we revisit the definition of the distinguishing advantage $\mathbf{Adv}^{\mathsf{E}_{K_B}}()$. Let's consider as distinguisher a binary output algorithm $\mathsf{A}^{\mathsf{LocateError}()}$, which invokes procedure $\mathsf{LocateError}()$ and returns 1 if the response is non-empty. This distinguisher passes into procedure $\mathsf{LocateError}()$ indicator values computed from a syndrome value S that satisfies (31) which, in turn, is computed by invoking either $\mathsf{E}_{K_B}^{-1}()$ or $\pi^*()$, on some given T, T'. The distinguisher attempts to identify whether the oracle invoked is $\mathsf{E}_{K_B}^{-1}()$ or $\pi^*()$. We assume that T, T' and oracle $\mathsf{E}_{K_B}^{-1}()$ satisfy Cond_4. By the definition of this distinguisher:

$$
\begin{aligned}
&\text{Prob}[\mathsf{A}^{\mathsf{LocateError}()^{\mathsf{E}_{K_B}^{-1}()}} \Rightarrow 1 \mid \mathsf{Cond}_4,\ H \xleftarrow{\$} \mathcal{H}] \\
&\quad = \sum_{\substack{S' \in \{0,1\}^N, \\ S' \neq 0}} P_{S'} \text{Prob}[\, S' = \mathsf{E}_{K_B}^{-1}(T) + \mathsf{E}_{K_B}^{-1}(T') \mid \mathsf{Cond}_4 \,] \\
&\quad = \text{Prob}[T' \notin \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \mid \mathsf{Cond}]
\end{aligned} \tag{36}
$$

Moreover, by the definition of $\mathbf{Adv}^{\mathsf{E}_{K_B}}()$, and assuming that $\mathbf{Adv}^{\mathsf{E}_{K_B}^{-1}}() = \mathbf{Adv}^{\mathsf{E}_{K_B}}()$, the probability that the algorithm $\mathsf{A}^{\mathsf{LocateError}()^{\mathsf{E}_{K_B}^{-1}()}}$ outputs 1 is bounded by the probability of the same event, if $\mathsf{E}_{K_B}^{-1}$ is replaced by random permutation $\pi^* : \{0,1\}^N \to \{0,1\}^N$, and term $\mathbf{Adv}^{\mathsf{E}_{K_B}}()$ is added to this bound:

$$
\begin{aligned}
&\text{Prob}[\mathsf{A}^{\mathsf{LocateError}()^{\mathsf{E}_{K_B}^{-1}()}} \Rightarrow 1 | \mathsf{Cond}_4,\ H \xleftarrow{\$} \mathcal{H}] \\
&\quad \leq\ \mathsf{A}^{\mathsf{LocateError}()^{\pi^*()}} \Rightarrow 1 | \mathsf{Cond}_3,\ H \xleftarrow{\$} \mathcal{H}] + \mathbf{Adv}^{\mathsf{E}_{K_B}}() \\
&\quad = \sum_{\substack{S' \in \{0,1\}^N, \\ S' \neq 0}} P_{S'} \text{Prob}[\, S' = \pi^*(T) + \pi^*(T') \mid \mathsf{Cond}_3 \,] + \mathbf{Adv}^{\mathsf{E}_{K_B}}()
\end{aligned} \tag{37}
$$

Combining (33), (34), (35), (36) and (37) we get:

$$\begin{aligned}
&\text{Prob}[T' \notin \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \mid \mathsf{Cond}] \\
&\quad \leq \sum_{\substack{S' \in \{0,1\}^N, \\ S' \neq 0}} P_{S'} \text{Prob}[\, S' = \pi^*(T) + \pi^*(T') \mid \mathsf{Cond}_3 \,] + \mathbf{Adv}^{\mathsf{E}_{K_B}()} \\
&\quad \leq \frac{n(n+1) \cdot |\mathcal{E}_{\mathcal{T}}|}{2|\mathcal{H}|} + \mathbf{Adv}^{\mathsf{E}_{K_B}()} \leq \frac{(n^2+n) \cdot |\mathcal{E}_{\mathcal{T}}|}{2^{N+1} - n^2 \cdot |\mathcal{E}_{\mathcal{T}}|^2} + \mathbf{Adv}^{\mathsf{E}_{K_B}()}
\end{aligned} \tag{38}$$

We conclude with computing a lower bound for the complement of the event $\{T' \notin \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \mid \mathsf{Cond}\}$, using relation (38).

$$\begin{aligned}
&\text{Prob}[T' \in \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \mid \mathsf{Cond}] \geq 1 - \frac{(n^2+n) \cdot |\mathcal{E}_{\mathcal{T}}|}{2^{N+1} - n^2 \cdot |\mathcal{E}_{\mathcal{T}}|^2} - \mathbf{Adv}^{\mathsf{E}_{K_B}()} \\
&\quad = 1 - \frac{(n^2+n) \cdot \sum_{t=1}^{T_{th}} \binom{N}{t}}{2^{N+1} - n^2 \cdot \left(\sum_{t=1}^{T_{th}} \binom{N}{t} \right)^2} - \mathbf{Adv}^{\mathsf{E}_{K_B}()}
\end{aligned} \tag{39}$$

□

From inequality (39), it follows that the fraction of corrupted tag values $T' \in \theta_{tag}(T, T', T_{th})$ which are not correctable is small for typical values of N, n and T_{th}. For example, for $N = 128$, $n = 4$ and $T_{th} = 10$, the probability $\text{Prob}[\, T' \in \vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th}) \mid \mathsf{Cond}\,]$ is greater than $1 - 2^{-76.864}$.

3 Security proofs

3.1 MAC forgery game

We consider a MAC adversary as defined in Section 1.3. The adversary is an algorithm $\mathcal{A}$ who is given query access to MAGIC's signing algorithm $\mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$ and to MAGIC's verification oracle $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$. We assume that $\mathcal{A}$ can perform both chosen plaintext as well as chosen ciphertext attacks. Furthermore, the adversary can observe the ciphertext blocks $C_1, C_2, \ldots, C_n$ that are produced from a sequence of submitted input plaintext blocks $P_1, P_2, \ldots, P_n$. These two assumptions are equivalent to considering that $\mathcal{A}$ can always submit a sequence of ciphertext blocks $C_1, C_2, \ldots, C_n$ of his choice either to $\mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$ or $\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}()$.

In such game, the encryption of the input plaintext is bypassed and the adversary directly submits ciphertext blocks. For this reason, we will be denoting the signing and verification oracles of MAGIC as just $\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}()$ and $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}()$ respectively, omitting the term $\mathsf{E}_{K_e}()$ from the notation.

We recall from Section 2 that the encryption function of MAGIC is performed independently from the message authentication and error correction functions and does not share secrets with them. Moreover, MAGIC does not introduce any new block cipher or mode. Instead, it uses some unspecified cipher denoted by $\mathsf{E}_{K_e}()$ and mode $\mathcal{M}_e$ for data confidentiality, both of which are assumed to be well designed. Therefore, for the purpose of this analysis it is sufficient to focus only on MAGIC's message authentication and error correction functions.

For this analysis, we consider that the adversary $\mathcal{A}$ submits $|Q_s|$ non-repeating queries to the signing oracle $\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}()$ and $|Q_v|$ non-repeating queries to the verification oracle $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}()$. The signing queries are from a set Q_s, where each element of Q_s comprises a single block of authentication data and a sequence of ciphertext blocks:

$$Q_s = \{(D^{(i)}, C_1^{(i)}, \ldots, C_n^{(i)}) : i = 1, \ldots, |Q_s|\} \tag{40}$$

On accepting signing query $q_s^{(i)} = (D^{(i)}, C_1^{(i)}, \ldots, C_n^{(i)})$, the signing oracle returns a tag value $T^{(i)}$ which is computed using relation (6) on inputs $D^{(i)}$ and $C_1^{(i)}, \ldots, C_n^{(i)}$.

The verification queries are from a set Q_v, where each element of Q_v comprises a single block of authentication data, a sequence of ciphertext blocks and a tag value:

$$Q_v = \{(D^{(i)}, C_1^{(i)}, \ldots, C_n^{(i)}, T^{(i)}) : i = 1, \ldots, |Q_v|\} \tag{41}$$

For all verification operations performed by $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}()$, the threshold value $T_{th} \in \{0,1\}^N$ used is assumed to be a constant. On accepting verification query $q_v^{(i)} = (D^{(i)}, C_1^{(i)}, \ldots, C_n^{(i)}, T^{(i)})$, the verification oracle executes lines 1-15 of the pseudocode presented in Section 2.2, returning a response $(b^{(i)}, \mathfrak{s}_1^{(i)}, \mathfrak{s}_2^{(i)})$, where $b^{(i)} \in \{\mathsf{accept}, \mathsf{reject}\}$ is a boolean and the two strings $\mathfrak{s}_1^{(i)}, \mathfrak{s}_2^{(i)}$ indicate whether error correction took place, and what the corrected inputs are. We also assume that $\mathcal{A}$ does not submit straightforward replays of sign queries to the verification oracle.

The adversary succeeds if, after submitting $|Q| = |Q_s| + |Q_v|$ queries, it successfully computes an input-tag set $(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)})$ which should not be "correctable" to any of the queries that have been issued. Considering the definition of Section 1.3 in the context of the MAGIC mode, it should hold that either:

$$(C_1^{(r)}, \ldots, C_n^{(r)}) \notin \bigcup_{i=1}^{|Q_s|} \theta_{cipher}(C_1^{(i)}, \ldots,\ C_n^{(i)}, T_{th}) \tag{42}$$

or:

$$\begin{gathered}\exists\, j \in [1, |Q_s|],\ (D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}) \in Q_s : \\ (D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}) = (\, D^{(j)}, C_1^{(j)}, \ldots,\ C_n^{(j)})\,, \\ (\, T^{(r)} \notin \vartheta_H^{\mathsf{E}_{K_B}()}(\, \mathsf{MAC}_H^{\mathsf{E}_{K_B}()}(D^{(j)},\, C_1^{(j)}, \ldots, C_n^{(j)}),\, T_{th})\,)\end{gathered} \tag{43}$$

Condition (43) is required in the definition of success, as an alternative to (42), because there exist corruptions in the tag within Hamming weight distance of T_{th}, which are not correctable, as discussed in Section 2.4. These corruptions cause the pseudocode of procedure $\mathsf{Verify}()$ to incorrectly execute lines 8-9, further corrupting the ciphertext blocks instead of correcting the error in the tag. This is one of the situations where the error correction functionality of MAGIC is exploited for creating forgeries. These situations are captured in the analysis that follows.

Definition 4: Let $\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}()$ and $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}()$ be sign and verification oracles of a MAGIC mode that uses a uniformly drawn hash key $H \xleftarrow{\$} \mathcal{H}$ and blinding cipher $\mathsf{E}_{K_B}()$ associated with mode $\mathcal{M}_B$, encryption key K_B of length l_{K_B}, $K_B \xleftarrow{\$} \{0,1\}^{l_{K_B}}$, and initialization vector i_B of length l_{i_B}, $i_B \xleftarrow{\$} \{0,1\}^{l_{i_B}}$. Let $\mathcal{A}$ be a polynomial time algorithm that submits $|Q_s|$ queries to the signing oracle and $|Q_v|$ queries to the verification oracle, also referred to as adversary. The event $\mathcal{W}^{\mathcal{A}}$ that adversary $\mathcal{A}$ wins a MAC forgery game played on the sign and verification oracles is defined as the event:

$$\begin{aligned}\mathcal{W}^{\mathcal{A}} = H \xleftarrow{\$} \mathcal{H};\ & K_B \xleftarrow{\$} \{0,1\}^{l_{K_B}};\ i_B \xleftarrow{\$} \{0,1\}^{l_{i_B}}; \\ & (D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) \leftarrow \mathcal{A};\ \mathsf{Cond}_5 \vee \mathsf{Cond}_6; \\ & \mathsf{Verify}_H^{\mathsf{E}_{K_B}()}(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) = (\mathsf{accept},\, \mathfrak{s}_1, \mathfrak{s}_2)\end{aligned} \tag{44}$$

for some strings $\mathfrak{s}_1, \mathfrak{s}_2$. Cond_5 refers to the condition of (42) and Cond_6 refers to the condition of (43).

Corollary 3: The advantage of adversary $\mathcal{A}$ is given by:

$$\mathbf{Adv}^{\mathcal{A}} = \mathrm{Prob}[\mathcal{W}^{\mathcal{A}}] \tag{45}$$

The sign and verification queries are not issued sequentially, one set after the other, but are mixed. This happens because adversary $\mathcal{A}$ issues queries in an arbitrary order specific to the way $\mathcal{A}$ plays the MAC forgery game. In the analysis that follows, we refer to queries with respect to both their order of issue inside the signing and verification query sets Q_s, Q_v, and inside the union set $Q = Q_s \cup Q_v$.

We adopt the following notation: A sign query q_s of index i_s indicating order of issue inside set Q_s is denoted by $q_s^{(i_s)}$. A verification query q_v of index i_v indicating order of issue inside set Q_v is denoted by $q_v^{(i_v)}$. This notation for q_s, q_v has been used in relations (40) and (41). A query q that is either a sign query or a verification query of index i indicating order of issue inside the union set $Q = Q_s \cup Q_v$ is denoted by $q^{[i]}$. For example, for a sign query $q_s = q$, there exist two indexes $i_s \in [1, |Q - s|]$ and $i \in [1, |Q_s| + |Q_v|]$ such that $q_s^{(i_s)} = q^{[i]}$.

Responses coming from oracles $\mathsf{MAC}()$ and $\mathsf{Verify}()$ are referred to using the symbol $\mathcal{O}()$. Specifically, a response coming from $\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}()$ or $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}()$ on accepting input $q^{[i]} \in Q$, $i \in [1, |Q_s| + |Q_v|]$, is denoted by $\mathcal{O}_H^{\mathsf{E}_{K_B}()}(q^{[i]})$ or just $\mathcal{O}(q^{[i]})$. It holds that either $\mathcal{O}(q^{[i]}) = \mathsf{MAC}_H^{\mathsf{E}_{K_B}()}(q^{[i]})$, if $q^{[i]} \in Q_s$, or $\mathcal{O}(q^{[i]}) = \mathsf{Verify}_H^{\mathsf{E}_{K_B}()}(q^{[i]}, T_{th})$, if $q^{[i]} \in Q_v$.

Finally, in the analysis that follows we consider adversaries that are identical to $\mathcal{A}$, but act based on information coming only from queries of the $\{q^{[1]}, q^{[2]}, \ldots, q^{[i]}\} \in Q$ for some index $i \in [1, |Q_s| + |Q_v|]$. We refer to such adversaries as $\mathcal{A}^{[i]}$.

3.2 Query budget constraint for adversary $\mathcal{A}$

Our analysis introduces a query budget constraint for adversary $\mathcal{A}$. The constraint is used in the proof of Proposition 2 below. The constraint is that the number of queries $|Q|$ issued by adversary $\mathcal{A}$ needs to satisfy the condition:

$$|Q| < \frac{2^{N+1}}{n(n+1)|\mathcal{E}_{\mathcal{T}}| + 4} \tag{46}$$

where $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$.

For typical values of n, N and T_{th}, the query budget constraint does not impact the security of MAGIC. As we establish in this section, MAGIC offers security in the order of $O(2^{N/2})$. On the other hand, for typical values of n, N and T_{th}, the bound of condition (46) is significantly larger than $2^{N/2}$. Hence, for typical values of n, N and T_{th}, adversary $\mathcal{A}$ succeeds before the query budget of (46) is exhausted. For example, for $N = 128$, $n = 4$ and $T_{th} = 10$, the bound of condition (46) is equal to $2^{76.864}$, which is greater than 2^{64}.

3.3 Attacking MAGIC when blinding is performed by a random permutation

We proceed with considering the MAC forgery game played on sign and verification oracles $\mathsf{MAC}()$ and $\mathsf{Verify}()$, for which the blinding cipher $\mathsf{E}_{K_B}()$ is replaced by a randomly chosen permutation $\pi^* \xleftarrow{\$} \mathcal{P}$ from the set $\mathcal{P}$ of all permutations $\pi : \{0,1\}^N \to \{0,1\}^N$. We refer to these attacked oracles as $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}$ respectively, and the adversary playing such game as $\mathcal{A}^*$.

Adversary $\mathcal{A}^*$ is identical to $\mathcal{A}$ in terms of behavior, but attacks a version of the MAGIC mode for which blinding is not performed by cipher $\mathsf{E}_{K_B}()$ but by the random permutation $\pi^*()$. For reasons that will be made clear in the paper, it is easier to analyze the security of oracles $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}$ when attacked by $\mathcal{A}^*$, than the security of oracles $\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}()$ and $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}$ attacked by $\mathcal{A}$.

We can associate the advantage of $\mathcal{A}^*$ with the advantage of $\mathcal{A}$. An adversary who plays a MAC forgery game, the success of which is defined by (44) is also a distinguisher. Such algorithm attempts to distinguish between interacting with the pair of oracles ($\mathsf{MAC}_H^{\pi^*()}()$, $\mathsf{Verify}_H^{\pi^*()}$) and the pair of oracles ($\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}()$, $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}$). The distinguisher plays the MAC forgery game of relations (44) and (45) in two different ways. In a first way, the blinding cipher queried by $\mathsf{MAC}()$ and $\mathsf{Verify}()$ is $\mathsf{E}_{K_B}()$. In a second way, the blinding cipher queried is $\pi^*()$. In the end of each of both games the distinguisher presents output equal to "1", if the MAC forgery game is won, and "0" otherwise. By the definition of this distinguisher and the definition of $\mathbf{Adv}^{\mathsf{E}_{K_B}()}$, it holds that:

$$\mathbf{Adv}^{\mathcal{A}} \leq \mathbf{Adv}^{\mathcal{A}^*} + \mathbf{Adv}^{\mathsf{E}_{K_B}()} \tag{47}$$

Let $\mathcal{W}^{\mathcal{A}^*}$ be the event defined by (44) when blinding is performed by $\pi^*()$. Then equation (47) can be written as:

$$\text{Prob}[\mathcal{W}^{\mathcal{A}^*}] \leq \text{Prob}[\mathcal{W}^{\mathcal{A}}] + \mathbf{Adv}^{\mathsf{E}_{K_B}()} \tag{48}$$

The analysis that follows focuses on adversary $\mathcal{A}^*$. Once we compute a bound for the advantage of $\mathcal{A}^*$, we add the term $\mathbf{Adv}^{\mathsf{E}_{K_B}()}$ to this bound to compute a bound for $\mathcal{A}$. We also note that the query index notation, the definition of the symbol $\mathcal{O}()$, and query budget constraint introduced above, all apply to $\mathcal{A}^*$ as well.

3.4 Security critical events during the MAC forgery game

We begin our security analysis by observing that there is a number of security critical events associated with the specification of the MAGIC mode which, if occurring, make adversary $\mathcal{A}^*$ win the game. One such event is a "MAC collision event".

Definition 5: *The MAC collision event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode for which blinding is performed by a random permutation π^*. Let $\mathcal{A}^*$ be a polynomial time algorithm playing a game the success of which is defined by relation (44), and where $\mathsf{E}_{K_B}() \leftarrow \pi^*()$. A MAC collision event $\mathcal{E}_{\mathcal{C}}^{H,\pi^*}(i_{\mathcal{C}})$, or just $\mathcal{E}_{\mathcal{C}}(i_{\mathcal{C}})$, associated with query of index $i_{\mathcal{C}}$, is defined as the event that the $\mathsf{MAC}()$ signing algorithm output for some query $q^{[i_{\mathcal{C}}]} \in Q_s$, $i_{\mathcal{C}} \in [1, |Q_s| + |Q_v|]$ is identical to the $\mathsf{MAC}()$ signing algorithm output of a different query $q^{[j_{\mathcal{C}}]} \in Q_S$, $j_{\mathcal{C}} \in [1, |Q_s| + |Q_v|]$, $j_{\mathcal{C}} < i_{\mathcal{C}}$.

$$\begin{gathered}\mathcal{E}_{\mathcal{C}}(i_{\mathcal{C}}) = q^{[i_{\mathcal{C}}]} \in Q_s;\;\; \exists\, j_{\mathcal{C}} : q^{[j_{\mathcal{C}}]} \in Q_s,\ j_{\mathcal{C}} < i_{\mathcal{C}} : \\ \mathsf{MAC}_H^{\pi^*()}(D^{[i_{\mathcal{C}}]}, C_1^{[i_{\mathcal{C}}]}, \ldots, C_n^{[i_{\mathcal{C}}]}) = \mathsf{MAC}_H^{\pi^*()}(D^{[j_{\mathcal{C}}]}, C_1^{[j_{\mathcal{C}}]}, \ldots, C_n^{[j_{\mathcal{C}}]})\end{gathered} \tag{49}$$

If such event occurs, adversary $\mathcal{A}^*$ deterministically wins the game. Since the blinding cipher $\pi^*()$ is a random permutation, equality of the outputs of the $\mathsf{MAC}()$ signing algorithm for queries $q^{[i_{\mathcal{C}}]}$ and $q^{[j_{\mathcal{C}}]}$ means equality of the outputs of the Galois Hash transformation for the same queries. By equating the Galois hash transformation outputs for these queries, adversary $\mathcal{A}^*$ forms a polynomial equation of degree n, where the unknown is the hash key value H. As the roots of this polynomial equation can be trivially found, adversary $\mathcal{A}^*$ can create forgeries at will. We note that a similar event can be defined for the adversary $\mathcal{A}$.

The next definition covers another type of event which is security critical, the "Galois Hash computation event":

Definition 6: *The Galois Hash computation event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Definition 5. A Galois Hash computation event $\mathcal{E}_{\mathcal{G}}^{H,\pi^*}(i_{\mathcal{G}})$, or just $\mathcal{E}_{\mathcal{G}}(i_{\mathcal{G}})$, associated with query of index $i_{\mathcal{G}}$, is defined as the event that for some query $q^{[j_{\mathcal{G}}]} \in Q_s$, $j_{\mathcal{G}} \in [1, |Q_s|+|Q_v|]$, $j_{\mathcal{G}} \leq i_{\mathcal{G}}$, for which $(C_1^{[j_{\mathcal{G}}]}, \ldots, C_n^{[j_{\mathcal{G}}]}) \neq (0, \ldots, 0)$, the adversary $\mathcal{A}^{*[i_{\mathcal{G}}]}$ can successfully compute the Galois Hash transformation output for this query's input.

$$\mathcal{E}_{\mathcal{G}}(i_{\mathcal{G}}) = q^{[j_{\mathcal{G}}]} \in Q_s;\ q^{[i_{\mathcal{G}}]} \in Q;\ j_{\mathcal{G}} \leq i_{\mathcal{G}};\ (C_1^{[j_{\mathcal{G}}]}, \ldots, C_n^{[j_{\mathcal{G}}]}) \neq (0, \ldots, 0)$$
$$G_{j_{\mathcal{G}}} \leftarrow \mathcal{A}^{*[i_{\mathcal{G}}]};\ D^{[j_{\mathcal{G}}]} + C_1^{[j_{\mathcal{G}}]} \cdot H + \ldots + C_n^{[j_{\mathcal{G}}]} \cdot H^n = G_{j_{\mathcal{G}}} \tag{50}$$

If $\mathcal{E}_{\mathcal{G}}(i_{\mathcal{G}})$ is true, then for query $q^{[j_{\mathcal{G}}]}$, the adversary can successfully decrypt the output of the $\mathsf{MAC}()$ signing algorithm. As in the case of the MAC collision event, the polynomial equation formed from equating the Galois Hash output to $G_{i_{\mathcal{G}}}$ can be trivially solved. One of the n roots of the equation must be equal to the hash key value H. Knowledge of hash key value H enables adversary $\mathcal{A}^{*[i_{\mathcal{G}}]}$, and thus $\mathcal{A}^*$ also, to create forgeries.

A third security critical event is an "ECC function exploitation event". We use the term "ECC function exploitation event" to refer to a group of situations where the adversary successfully exploits the ECC functionality of the MAGIC mode in order to create forgeries. In these situations, it is not algorithm $\mathcal{A}^*$ which crafts forgeries, but instead, procedure $\mathsf{Verify}()$ invoked by the verification oracle of MAGIC.

Specifically, there are situations where procedure $\mathsf{Verify}()$ may incorrectly execute lines 8-9 of its pseudocode attempting to correct a single block in error, whereas in reality there may be more than one ciphertext blocks corrupted. There are also situations where a single block may be in error, but the error demonstrates Hamming weight larger than T_{th}. All these corruptions may cause procedure $\mathsf{LocateError}()$, invoked in line 7 of the $\mathsf{Verify}()$ pseudocode, to incorrectly return a non-empty response. Such non-empty response makes procedure $\mathsf{Verify}()$ execute lines 8-9. In this way, procedure $\mathsf{Verify}()$, instead of correcting any errors present, creates a forgery.

There is a third group of situations where procedure $\mathsf{Verify}()$ may create a forgery. This is when it incorrectly execute lines 12-13 of its pseudocode attempting to correct an error in the tag value. In reality there may be errors in one or more ciphertext blocks.

Definition 7: *The ECC function exploitation event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Definition 5. An ECC function exploitation event $\mathcal{E}_{\mathcal{X}}^{H,\pi^*}(i_{\mathcal{X}})$, or just $\mathcal{E}_{\mathcal{X}}(i_{\mathcal{X}})$, associated with query of index $i_{\mathcal{X}}$, is defined as the event that some query $q^{[i_{\mathcal{X}}]} = (D^{[i_{\mathcal{X}}]}, C_1^{[i_{\mathcal{X}}]}, \ldots, C_n^{[i_{\mathcal{X}}]}, T^{[i_{\mathcal{X}}]})$ $\in Q_v$ passed to the verification oracle of MAGIC, which satisfies either (42) or (43) for $(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) \leftarrow q^{[i_{\mathcal{X}}]}$ and $\mathsf{E}_{K_B}() \leftarrow \pi^*()$, causes the verification oracle to return response $(b^{[i_{\mathcal{X}}]}, \mathfrak{s}_1^{[i_{\mathcal{X}}]}, \mathfrak{s}_2^{[i_{\mathcal{X}}]})$, where $b^{[i_{\mathcal{X}}]} = \mathsf{accept}$, $\mathfrak{s}_1^{[i_{\mathcal{X}}]} \neq \perp$ and $\mathfrak{s}_2^{[i_{\mathcal{X}}]} \neq \perp$.

$$\mathcal{E}_{\mathcal{X}}(i_{\mathcal{X}}) = q^{[i_{\mathcal{X}}]} \leftarrow \mathcal{A}^{*[i_{\mathcal{X}}-1]};\ \ q^{[i_{\mathcal{X}}]} \in Q_v;\ \ \mathsf{Cond}_5 \vee \mathsf{Cond}_6; \\ \mathsf{Verify}_H^{\pi^*()}(q^{[i_{\mathcal{X}}]}) = (\mathsf{accept},\ \mathfrak{s}_1^{[i_{\mathcal{X}}]},\ \mathfrak{s}_2^{[i_{\mathcal{X}}]});\ \ \mathfrak{s}_1^{[i_{\mathcal{X}}]} \neq \bot,\ \mathfrak{s}_2^{[i_{\mathcal{X}}]} \neq \bot \tag{51}$$

where Cond_5 and Cond_6 refer to the conditions of (42) and (43) evaluated on $(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) \leftarrow q^{[i_{\mathcal{X}}]}$, and for which $\mathsf{E}_{K_B}() \leftarrow \pi^*()$.

Again, we can see why the ECC function exploitation event as defined above is security critical. Query $q^{[i_{\mathcal{X}}]}$ is not correctable to any of the other queries issued by adversary $\mathcal{A}^*$, according to one of conditions Cond_5, Cond_6. On other hand the verification oracle $\mathsf{Verify}_H^{\pi^*()}()$, on accepting input $q^{[i_{\mathcal{X}}]}$, returns non-empty strings $\mathfrak{s}_1^{[i_{\mathcal{X}}]}$ and $\mathfrak{s}_2^{[i_{\mathcal{X}}]}$. This is an indication that the procedure either executes lines 8-9 of its pseudocode or lines 12-13. In both cases it attempts to perform error correction.

Proposition 1: Let's consider an ECC function expolitation event as in Definition 7. Let query $q^{[i_{\mathcal{X}}]}$ be equal to $(D^{[i_{\mathcal{X}}]}, C_1^{[i_{\mathcal{X}}]}, \ldots, C_n^{[i_{\mathcal{X}}]}, T^{[i_{\mathcal{X}}]})$. Let $C_1^+, \ldots, C_n^+$ be the ciphertext blocks, the binary representation of which is the string $\mathfrak{s}_1^{[i_{\mathcal{X}}]}$ returned from query $q^{[i_{\mathcal{X}}]}$. Let T^+ be the tag value, the binary representation of which is the string $\mathfrak{s}_2^{[i_{\mathcal{X}}]}$, also returned from query $q^{[i_{\mathcal{X}}]}$. Then, one of the following three statements must be true:

i. *The ECC function is not exploited, but the ECC output is a forgery.* In this case $(C_1^{[i_{\mathcal{X}}]}, \ldots, C_n^{[i_{\mathcal{X}}]}) \in \theta_{cipher}(C_1^+, \ldots, C_n^+, T_{th})$, but $(D^{[i_{\mathcal{X}}]}, C_1^+, \ldots, C_n^+) \notin Q_S$.

ii. *The ECC function is exploited, and the ECC output is a forgery.* In this case $(C_1^{[i_{\mathcal{X}}]}, \ldots, C_n^{[i_{\mathcal{X}}]}) \notin \theta_{cipher}(C_1^+, \ldots, C_n^+, T_{th})$, and also $(D^{[i_{\mathcal{X}}]}, C_1^+, \ldots, C_n^+) \notin Q_S$.

iii. *The ECC function is exploited, and the ECC output is a replay.* In this case $(C_1^{[i_{\mathcal{X}}]}, \ldots, C_n^{[i_{\mathcal{X}}]}) \notin \theta_{cipher}(C_1^+, \ldots, C_n^+, T_{th})$, but $(D^{[i_{\mathcal{X}}]}, C_1^+, \ldots, C_n^+) \in Q_S$.

Proof of Proposition 1: From lines 8-9 and 12-13 of the pseudocode of $\mathsf{Verify}()$, it is deduced that the returned tag value T^+ is the valid tag computed from authenticated data $D^{[i_{\mathcal{X}}]}$ and the returned ciphertext blocks $C_1^+, \ldots, C_n^+$. Specifically, $T^+ = \mathsf{MAC}_H^{\pi^*()}(D^{[i_{\mathcal{X}}]},\ C_1^+, \ldots, C_n^+)$. Because of this fact, and the fact that query $q^{[i_{\mathcal{X}}]}$ satisfies conditions Cond_5 and Cond_6 as defined in Definition 7, it is not possible for the following two relations to be simultaneously true: (a) $(C_1^{[i_{\mathcal{X}}]}, \ldots, C_n^{[i_{\mathcal{X}}]}) \in \theta_{cipher}(C_1^+, \ldots, C_n^+, T_{th})$ and (b) $(D^{[i_{\mathcal{X}}]}, C_1^+, \ldots, C_n^+) \in Q_S$. If this was the case, then $q^{[i_{\mathcal{X}}]}$ would have been correctable to one of the sign queries. Given this fact, the correctness of Proposition 1 follows from the observation that statements i-iii cover the space of all remaining possible events and are mutually exclusive.

□

In the case of statement i, the ciphertext-tag pair contained in $q^{[i_{\mathcal{X}}]}$ is corrected, and a single block error is removed. The corrected ciphertext, however, has never been passed as a query to the signing oracle. In this case the adversary wins the game. A forgery has been produced.

In the case of statement ii, the ciphertext-tag pair contained in $q^{[i_{\mathcal{X}}]}$ becomes further corrupted by procedure $\mathsf{Verify}()$. The ECC functionality of the MAGIC mode is abused. Procedure $\mathsf{Verify}()$ produces some ciphertext-tag pair which, one the one hand is valid, and one the other hand has never been passed as a query to the signing oracle. This is another forgery case.

In the case of statement iii, as in the previous one, the ciphertext-tag pair contained in $q^{[i_{\mathcal{X}}]}$ becomes further corrupted by procedure $\mathsf{Verify}()$. However, the ciphtertext produced by $\mathsf{Verify}()$ has been passed as query to the signing oracle $\mathsf{MAC}()$ before. According to our definition of success for the MAC adversary, this is a forgery too. In fact, it is also a replay. The adversary starts with a query which is not correctable to any other known query, but successfully abuses the ECC functionality to craft a known ciphertext-tag pair.

A last type of security critical event studied is a "blind forgery event". This is the simplest case of forgery. We use the term to refer to the situation where adversary $\mathcal{A}^*$ presents to the verification oracle $\mathsf{Verify}()$ of the MAGIC mode a valid ciphertext-tag pair never queried before:

Definition 8: *The blind forgery event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Definition 5. A blind forgery event $\mathcal{E}_{\mathcal{B}}^{H,\pi^*}(i_{\mathcal{B}})$, or just $\mathcal{E}_{\mathcal{B}}(i_{\mathcal{B}})$, associated with query of index $i_{\mathcal{B}}$, is defined as the event that some query $q^{[i_{\mathcal{B}}]} \in Q_v$ passed to the verification oracle of MAGIC, which satisfies either (42) or (43) for $(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) \leftarrow q^{[i_{\mathcal{B}}]}$ and $\mathsf{E}_{K_B}() \leftarrow \pi^*()$, causes the verification oracle to return response $(\mathsf{accept}, \perp, \perp)$.

$$\begin{gathered}\mathcal{E}_{\mathcal{B}}(i_{\mathcal{B}}) = q^{[i_{\mathcal{B}}]} \leftarrow \mathcal{A}^{*[i_{\mathcal{B}}-1]};\;\; q^{[i_{\mathcal{B}}]} \in Q_v;\;\; \mathsf{Cond}_5 \,\vee\, \mathsf{Cond}_6; \\ \mathsf{Verify}_H^{\pi^*()}(q^{[i_{\mathcal{B}}]}) = (\mathsf{accept}, \perp, \perp)\end{gathered} \tag{52}$$

where Cond_5 and Cond_6 refer to the conditions of (42) and (43) evaluated on $(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) \leftarrow q^{[i_{\mathcal{X}}]}$, and for which $\mathsf{E}_{K_B}() \leftarrow \pi^*()$.

The fact that the strings returned by the verification oracle are empty is an indication that no error correction took place. By the definition of this event, the input ciphertext-tag pair is valid, it has never been queried before, and is thus a forgery. Adverary $\mathcal{A}^*$ wins the game in this case as well.

3.5 Complementary events and probabilities of first occurrence

For the MAC collision, Galois Hash computation, ECC function exploitation and blind forgery events, we refer to their complements as:

$$\begin{aligned} \mathcal{N}_{\mathcal{C}}(i) = \neg\, \mathcal{E}_{\mathcal{C}}(i), \quad \mathcal{N}_{\mathcal{G}}(i) = \neg\, \mathcal{E}_{\mathcal{G}}(i), \\ \mathcal{N}_{\mathcal{X}}(i) = \neg\, \mathcal{E}_{\mathcal{X}}(i), \quad \mathcal{N}_{\mathcal{B}}(i) = \neg\, \mathcal{E}_{\mathcal{B}}(i) \end{aligned} \tag{53}$$

where (53) applies to every query index $i \in [1, |Q_s| + |Q_v|]$, $q^{[i]} \in Q$.

We also refer to the event of no security critical events occurring at query of index i, $\mathcal{N}(i)$, as:

$$\mathcal{N}(i) = \mathcal{N}_{\mathcal{C}}(i) \;\wedge\; \mathcal{N}_{\mathcal{G}}(i) \;\wedge\; \mathcal{N}_{\mathcal{X}}(i) \;\wedge\; \mathcal{N}_{\mathcal{B}}(i) \tag{54}$$

where, again, (54) applies to every query index $i \in [1, |Q_s| + |Q_v|]$, $q^{[i]} \in Q$.

Next, we refer to the event of no security critical events occurring during the entire MAC forgery game played by $\mathcal{A}^*$, $\mathcal{N}^{\mathcal{A}^*}$, as:

$$\mathcal{N}^{\mathcal{A}^*} = \bigwedge\nolimits_{i=1}^{|Q_s|+|Q_v|} \mathcal{N}(i) \tag{55}$$

It's complement $\mathcal{S}^{\mathcal{A}^*}$ is the event that at least one security critical event occurs during the MAC forgery game, at the completion of at least one query:

$$\mathcal{S}^{\mathcal{A}^*} = \neg\, \mathcal{N}^{\mathcal{A}^*} \tag{56}$$

The next definition is about the probability that a security critical event occurs, if it is known that no other security critical events occur at previous queries. We refer to this probability as a "probability of first occurrence" of a security critical event.

Definition 9: *Probability of first occurrence of a security critical event.* Let $\mathcal{E}(i)$, $\mathcal{E}(i) \in \{\mathcal{E}_{\mathcal{C}}(i), \mathcal{E}_{\mathcal{G}}(i), \mathcal{E}_{\mathcal{X}}(i), \mathcal{E}_{\mathcal{B}}(i)\}$ be a security critical event associated with query $q^{[i]}$ of index $i \in [1, |Q_s| + |Q_v|]$, and defined by one of Definitions 5, 6, 7 or 8. A probability of first occurrence $F(i) \in \{F_{\mathcal{C}}(i),\ F_{\mathcal{G}}(i),\ F_{\mathcal{X}}(i),\ F_{\mathcal{B}}(i)\}$ of $\mathcal{E}(i)$ is defined as the probability of event $\mathcal{E}(i)$ occurring at query $q^{[i]}$, given the events $\mathcal{N}(1), \mathcal{N}(2), \ldots, \mathcal{N}(i-1)$:

$$F(i) \;=\; \text{Prob}\,[\,\mathcal{E}(i)\,|\,\mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \tag{57}$$

In what follows we use the notation $F(i)$ to refer to one of $F_{\mathcal{C}}(i)$, $F_{\mathcal{G}}(i)$, $F_{\mathcal{X}}(i)$ or $F_{\mathcal{B}}(i)$, depending on whether $\mathcal{E}(i)$ is a MAC collision, Galois Hash computation, ECC function exploitation, or blind forgery event respectively.

3.6 Structure of the proof

The first of our auxiliary results is about the structure of our proof. Our strategy to bound $\mathbf{Adv}^{\mathcal{A}^*}$ is to consider the event $\mathcal{W}^{\mathcal{A}^*}$ of adversary $\mathcal{A}^*$ succeeding in conjunction with the events $\mathcal{N}^{\mathcal{A}^*}$ and $\mathcal{S}^{\mathcal{A}^*}$, which are complementary. Computing a bound for $\mathcal{W}^{\mathcal{A}^*}$ is much easier, if it known that no security critical events occur during adversary $\mathcal{A}^*$'s game. On the other hand, the probability $\text{Prob}[\mathcal{W}^{\mathcal{A}^*} \wedge \mathcal{S}^{\mathcal{A}^*}]$ can be bounded by a sum of probabilities of first occurrence $F(i) \in \{F_{\mathcal{C}}(i),\ F_{\mathcal{G}}(i),\ F_{\mathcal{X}}(i),\ F_{\mathcal{B}}(i)\}$, $i \in [1, |Q_s| + |Q_v|]$. Bounds for such probabilities are also computed.

Lemma 5: Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode, for which blinding is performed by a random permutation π^*. Let $\mathcal{A}^*$ be a polynomial time algorithm playing a game, the success $\mathcal{W}^{\mathcal{A}^*}$ of which is defined by (44) for $\mathsf{E}_{K_B}() \leftarrow \pi^*()$ and $\mathcal{A} \leftarrow \mathcal{A}^*$. Let $\mathcal{E}_{\mathcal{C}}(i)$, $\mathcal{E}_{\mathcal{G}}(i)$, $\mathcal{E}_{\mathcal{X}}(i)$ and $\mathcal{E}_{\mathcal{B}}(i)$ be security critical events defined by Definitions 5, 6, 7 and 8, associated with the attacked oracles $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$, and the game played by adversary $\mathcal{A}^*$. Finally, let $F_{\mathcal{C}}(i)$, $F_{\mathcal{G}}(i)$, $F_{\mathcal{X}}(i)$ and $F_{\mathcal{B}}(i)$ be probabilities of first time occurrence associated with each of $\mathcal{E}_{\mathcal{C}}(i)$, $\mathcal{E}_{\mathcal{G}}(i)$, $\mathcal{E}_{\mathcal{X}}(i)$ and $\mathcal{E}_{\mathcal{B}}(i)$ respectively, and which are defined by Definition 9. Then, the probability of adversary $\mathcal{A}^*$ succeeding $\text{Prob}[\mathcal{W}^{\mathcal{A}^*}]$ is bounded by:

$$\begin{aligned}\text{Prob}[\mathcal{W}^{\mathcal{A}^*}] \ \leq \ & \text{Prob}[\mathcal{W}^{\mathcal{A}^*} | \mathcal{N}^{\mathcal{A}^*}] + \\ & \sum_{i=1}^{|Q|} (F_{\mathcal{C}}(i) + F_{\mathcal{G}}(i) + F_{\mathcal{X}}(i) + F_{\mathcal{B}}(i))\end{aligned} \tag{58}$$

where $\mathcal{N}^{\mathcal{A}^*}$ is the event of no security critical events occurring during adversary $\mathcal{A}^*$'s game and $|Q|$ is the total number of queries issued by the adversary.

Proof of Lemma 5: The probability of the event $\mathcal{W}^{\mathcal{A}^*}$ can be computed when the event is considered in conjunction with the complementary events $\mathcal{N}^{\mathcal{A}^*}$ and $\mathcal{S}^{\mathcal{A}^*}$.

$$\begin{aligned}\text{Prob}[\mathcal{W}^{\mathcal{A}^*}] &= \text{Prob}[\mathcal{W}^{\mathcal{A}^*} \wedge \mathcal{N}^{\mathcal{A}^*}] \ + \ \text{Prob}[\mathcal{W}^{\mathcal{A}^*} \wedge \mathcal{S}^{\mathcal{A}^*}] \\ &\leq \ \text{Prob}[\mathcal{W}^{\mathcal{A}^*} \,|\, \mathcal{N}^{\mathcal{A}^*}] \ + \ \text{Prob}[\,\mathcal{S}^{\mathcal{A}^*}]\end{aligned} \tag{59}$$

We recall from the previous section that $\mathcal{S}^{\mathcal{A}^*}$ is the event that at least one security critical event occurs during the MAC forgery game of $\mathcal{A}^*$, at the completion of at least one query. By the definition of $\mathcal{S}^{\mathcal{A}^*}$:

$$\text{Prob}[\mathcal{S}^{\mathcal{A}^*}] = \sum_{i=1}^{|Q|} \text{Prob}[\,\big(\,\mathcal{E}_{\mathcal{C}}(i) \vee \mathcal{E}_{\mathcal{G}}(i) \vee \mathcal{E}_{\mathcal{X}}(i) \vee \mathcal{E}_{\mathcal{B}}(i)\,\big) \wedge \big(\,\mathcal{N}(1) \wedge \ldots \wedge \mathcal{N}(i-1)\,\big)\,] \tag{60}$$

From equality (60), we bound the probability $\text{Prob}[\mathcal{S}^{\mathcal{A}^*}]$ as shown below:

$$\begin{aligned}&\text{Prob}[\mathcal{S}^{\mathcal{A}^*}] \leq \\ \sum_{i=1}^{|Q|} &\big(\,\text{Prob}[\,\mathcal{E}_{\mathcal{C}}(i)\,|\,\mathcal{N}(1),\ldots,\mathcal{N}(i-1)\,] + \text{Prob}[\,\mathcal{E}_{\mathcal{G}}(i)\,|\,\mathcal{N}(1),\ldots,\mathcal{N}(i-1)\,] + \\ &\text{Prob}[\,\mathcal{E}_{\mathcal{X}}(i)\,|\,\mathcal{N}(1),\ldots,\mathcal{N}(i-1)\,] + \text{Prob}[\,\mathcal{E}_{\mathcal{B}}(i)\,|\,\mathcal{N}(1),\ldots,\mathcal{N}(i-1)\,]\,\big)\end{aligned} \tag{61}$$

By definition (57), the right side of inequality (61) is equal to $\sum_{i=1}^{|Q|}(F_{\mathcal{C}}(i) + F_{\mathcal{G}}(i) + F_{\mathcal{X}}(i) + F_{\mathcal{B}}(i))$. The correctness of inequality (58) and Lemma 5 follow directly by combining relations (59) and (61).

□

3.7 Impossible Galois Hash key values

The next three lemmas and proposition are about the statistical properties of the internal secrets of the MAGIC mode, specifically the Galois Hash key value $H \in \mathcal{H}$, and the random permutation $\pi^* \in \mathcal{P}$. The lemmas and proposition show how the statistical properties of H, π^* change as adversary $\mathcal{A}^*$ issues queries to the sign and verification oracles of the mode. The lemmas also establish probability distributions for H, π^*, which characterize the secrets at the time of completion of query $q^{[i]} \in Q$, $i \in [1, \{|Q_s| + |Q_v|]\}$. For the computation of the distributions, it is assumed that responses from queries $q^{[1]}, \ldots, q^{[i]}$ are available. It is also assumed that no security critical events occur up to, and including, query $q^{[i]}$.

Initially H and π^* are random uniformly distributed in the sets $\mathcal{H}$ and $\mathcal{P}$ respectively. Then, adversary $\mathcal{A}^*$ issues queries. The responses coming from oracles $\mathsf{MAC}()$ and $\mathsf{Verify}()$ leak information about H and π^*. As we show below, information is leaked even if no security critical events occur up to query $q^{[i]}$.

Indeed, if no MAC collisions are observed at the completion of sign queries up to $q^{[i]}$, and since blinding is performed by permutation π^*, all sign queries up to $q^{[i]}$ result in unique Galois Hash outputs. Such knowledge makes a subset of Galois Hash key values impossible. Impossible hash keys are all those which are roots of polynomial equations formed by equating Galois Hash outputs coming from every pair of issued sign queries.

Second, if no ECC function exploitation events are observed at the completion of verification queries up to $q^{[i]}$, then such knowledge leaks information about permutation π^*. Permutation π^* must be in a subset of $\mathcal{P}$ that includes only those permutations that do not cause procedure $\mathsf{Verify}()$ to abuse its error correcting capability. Specifically, the input-output mappings specified by π^* must be such that procedure $\mathsf{Verify}()$ does not abuse lines 8-9 or 12-13 of its pseudocode.

The existence of impossible hash keys is established by Lemma 6. Then, Proposition 2 shows that for every hash key value $H_{\mathcal{P}}$ which is not impossible, there exists a set of blinding permutations $\rho()$, which, together with the hash key $H_{\mathcal{P}}$, respond to adversary $\mathcal{A}^*$'s queries in the same way as the combination H, π^*, if no security critical events occur up to query $q^{[i]}$. Based on these two results, Lemmas 7 and 8 establish that after i queries complete, and if no security critical events occur up to $q^{[i]}$, secrets H and π^* remain uniformly distributed, taking values from subsets of $\mathcal{H}$, $\mathcal{P}$. Such subsets are computed based on knowledge of queries up to $q^{[i]}$.

Lemma 6: *On the existence of impossible Galois Hash key values, if no security critical events occur up to query* $q^{[i]}$. Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $i \in |Q_s| + |Q_v|$ be a query index, for which $q^{[1]}, \ldots, q^{[i]} \in Q$. Let's also consider that no security critical events occur up to query $q^{[i]}$, and knowledge about queries $q^{[1]}, \ldots, q^{[i]}$ and their associated responses $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ is available. Then, there exists a subset $\mathcal{H}_{\mathcal{I}}^{[i]} \subseteq \mathcal{H}$, which we refer to as the subset of "impossible" hash key values, associated with query index i and set $\mathcal{H}$, such that for every hash key value $H_{\mathcal{I}} \in \mathcal{H}_{\mathcal{I}}^{[i]}$, the probability that the secret hash key H of the mode is equal to $H_{\mathcal{I}}$ is 0:

$$\begin{gathered} \exists\, \mathcal{H}_{\mathcal{I}}^{[i]} \subseteq \mathcal{H} \;:\; \forall\, H_{\mathcal{I}} \in \mathcal{H}_{\mathcal{I}}^{[i]} \;: \\ \mathrm{Prob}[\, H_{\mathcal{I}} = H \mid q^{[1]}, \ldots, q^{[i]},\, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}),\, \mathcal{N}(1), \ldots, \mathcal{N}(i)] = 0 \end{gathered} \tag{62}$$

where $i \in |Q_s| + |Q_v|$. Furthermore the cardinality $|\mathcal{H}_{\mathcal{I}}^{[i]}|$ of the set $\mathcal{H}_{\mathcal{I}}^{[i]}$ satisfies:

$$|\mathcal{H}_{\mathcal{I}}^{[i]}| \leq n \cdot \binom{|Q_s^{[i]}|}{2} + \frac{n(n+1)}{2} \cdot |Q_v^{[i]}| \cdot |\mathcal{E}_{\mathcal{T}}| \tag{63}$$

where $|Q_s^{[i]}|$ is the number of sign queries issued by adversary $\mathcal{A}^*$ up to query $q^{[i]}$, $|Q_v^{[i]}|$ is the number of verification queries issued by $\mathcal{A}^*$ up to query $q^{[i]}$, and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$.

Proof of Lemma 6: For every pair of sign queries $q_s^{(j)}, q_s^{(j')} \in Q_s^{[i]}$, where indexes j and j', $j, j' \in [1, |Q_s^{[i]}|]$, indicate order of issue inside the set $Q_s^{[i]}$, it must hold that:

$$\text{Prob}[\mathsf{MAC}_H^{\pi^*()}(q_s^{(j)}) = \mathsf{MAC}_H^{\pi^*()}(q_s^{(j')}) \mid q_s^{(j)}, q_s^{(j')}, \mathcal{N}(1), \ldots, \mathcal{N}(i)] = 0 \tag{64}$$

This is because condition $\mathcal{N}(1), \ldots, \mathcal{N}(i)$ indicates that no MAC collisions occur up to query $q^{[i]}$.

We proceed by denoting query $q_s^{(j)}$ as $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)})$ and query $q_s^{(j')}$ as $(D^{(j')}, C_1^{(j')}, \ldots, C_n^{(j')})$. Using such notation, equation (64) can be written as:

$$\begin{aligned} \text{Prob}[\ &\pi^*(D^{(j)} + C_1^{(j)}H + \ldots + C_n^{(j)}H^n) = \\ &\quad \pi^*(D^{(j')} + C_1^{(j')}H + \ldots + C_n^{(j')}H^n) \mid\ q_s^{(j)}, q_s^{(j')}, \mathcal{N}(1), \ldots, \mathcal{N}(i)\ ] = 0 \end{aligned} \tag{65}$$

Next, we apply operator $\pi^{*^{-1}}()$ to both sides of the internal equation of (65) to obtain:

$$\begin{aligned} \text{Prob}[\ \ &(D^{(j)} + D^{(j')}) + (C_1^{(j)} + C_1^{(j')})H + \ldots + \\ &(C_n^{(j)} + C_1^{(j')})H^n = 0 \mid\ q_s^{(j)}, q_s^{(j')}, \mathcal{N}(1), \ldots, \mathcal{N}(i)\ ] = 0 \end{aligned} \tag{66}$$

The internal polynomial equation of (66) has r distinct roots, where $r \geq 0$ and $r \leq n$. We refer to such roots as $H_1^{(j,j')}, \ldots, H_r^{(j,j')}$. Then equation (66) can be written as:

$$\text{Prob}[H = H_1^{(j,j')}\ \vee\ \ldots\ \vee\ H = H_r^{(j,j')} \mid\ q_s^{(j)}, q_s^{(j')}, \mathcal{N}(1), \ldots, \mathcal{N}(i)\ ] = 0 \tag{67}$$

We define $\mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]}$ to be the set that includes all roots $H_1^{(j,j')}, \ldots, H_r^{(j,j')}$ of all internal polynomial equations of the form of (66), where the equations are formed for every pair of sign queries $q_s^{(j)}, q_s^{(j')} \in Q_s^{[i]}$.

From the definition of the set $\mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]}$, it holds that $\mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]} \subseteq \mathcal{H}_{\mathcal{I}}^{[i]}$. The cardinality of the set $\mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]}$ satisfies $|\mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]}| \leq n \cdot \binom{|Q_s^{[i]}|}{2}$. This is because there are $\binom{|Q_s^{[i]}|}{2}$ pairs of sign queries $q_s^{(j)}, q_s^{(j')} \in Q_s^{[i]}$ and the number of roots r of all internal polynomial equations of the form of (66), computed for every pair of sign queries does not exceed n.

Another security critical event, the absence of which indicates that some hash keys are impossible is the ECC function exploitation event. Indeed, let's consider a verification query $q_v^{(j)} = (D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}, T^{(j)}) \in Q_v^{[i]}$, with index $j \in [1, |Q_v^{[i]}|]$ denoting order of issue inside the set $Q_v^{[i]}$, and a sign query $q_s^{(j')} = (D^{(j')}, C_1^{(j')}, \ldots, C_n^{(j')}) \in Q_s^{[i]}$ such that the sign oracle response $\mathcal{O}(q_s^{(j')})$ to query $q_s^{(j')}$ is identical to the tag value $T^{(j)}$ contained in the verification query $q_v^{(j)}$, i.e., $\mathcal{O}(q_s^{(j')}) = T^{(j)}$.

As it is assumed that no ECC function exploitation event occurs at the completion of queries up to $q^{[i]}$, this assumption holds for query $q^{(j)}$ as well. We further assume that, at the completion of query $q^{(j)}$, no correction of ciphertext blocks takes place. As such, query $q^{(j)}$ is one for which, either procedure $\mathsf{Verify}_H^{\pi^*()}()$ corrects the tag value $T^{(j)}$, or just returns $(\mathsf{reject}, \perp, \perp)$. This further means that procedure $\mathsf{LocateError}()$ invoked by $\mathsf{Verify}_H^{\pi^*()}()$ on input $q^{(j)}$ returns the empty string "$\perp$". Under these assumptions, and given $\mathcal{O}(q_s^{(j')}) = T^{(j)}$, the following equation must hold:

$$\begin{aligned}\mathrm{Prob}[\ \big(D^{(j)} + C_1^{(j)}H + \ldots + C_n^{(j)}H^n + {\pi^*}^{-1}(\mathcal{O}(q_s^{(j')}))\big) \cdot H^{-k} = e \\ \mid q_v^{(j)}, q_s^{(j')}, \mathcal{O}(q_s^{(j')}) = T^{(j)},\ \mathcal{N}(1), \ldots, \mathcal{N}(i)\,] = 0\end{aligned} \tag{68}$$

for every $k \in [i, n]$ and $e \in \mathcal{E}_{\mathcal{T}}$. Substituting ${\pi^*}^{-1}(\mathcal{O}(q_s^{(j')}))$ with its equal $D^{(j')} + C_1^{(j')}H + \ldots + C_n^{(j')}H^n$, we obtain:

$$\begin{aligned}\mathrm{Prob}[\ (D^{(j)} + D^{(j')}) + (C_1^{(j)} + C_1^{(j')})H + \ldots + (C_n^{(j)} + C_1^{(j')})H^n \ = \ e \cdot H^k \\ \mid\ q_v^{(j)}, q_s^{(j')}, \mathcal{O}(q_s^{(j')}) = T^{(j)}, \mathcal{N}(1), \ldots, \mathcal{N}(i)\,] = 0\end{aligned} \tag{69}$$

The internal polynomial equation of (69) has degree n, and r distinct roots, $r \in [0, n]$. We define $\mathcal{H}_{\mathcal{I},\mathsf{EXPL}}^{[i]}$ to be the set that includes all roots of the internal polynomial equation of (69), where the equation applies to every pair consisting of a sign query $q_s^{(j')}$ and a verification query $q_v^{(j)}$ such that $\mathcal{O}(q_s^{(j')}) = T^{(j)}$, every $k \in [1, n]$, and every $e \in \mathcal{E}_{\mathcal{T}}$. The cardinality of the set $\mathcal{H}_{\mathcal{I},\mathsf{EXPL}}^{[i]}$ satisfies $|\mathcal{H}_{\mathcal{I},\mathsf{EXPL}}^{[i]}| \leq \frac{n(n+1)}{2} \cdot |Q_v^{[i]}||\mathcal{E}_{\mathcal{T}}|$. This is because there can be at most $|Q_v^{(i)}|$ such pairs and the number of roots of the internal polynomial equation of (69) computed for a specific pair does not exceed $\frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$.

We also note that the set $\mathcal{H}_{\mathcal{I},\mathsf{EXPL}}^{[i]}$ may be empty, specifically if the condition $\mathcal{O}(q_s^{(j')}) = T^{(j)}$ is never satisfied for any pair $q_s^{(j')}, q_v^{(j)}$. The proof of the first part of Lemma 6 follows by setting:

$$\mathcal{H}_{\mathcal{I}}^{[i]} \leftarrow \mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]} \cup \mathcal{H}_{\mathcal{I},\mathsf{EXPL}}^{[i]} \tag{70}$$

The proof of the second part of Lemma 6 follows by adding the cardinality bounds for the sets $\mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]}$ and $\mathcal{H}_{\mathcal{I},\mathsf{EXPL}}^{[i]}$.

□

Corollary 4: The cardinality $|\mathcal{H}_{\mathcal{I}}^{[i]}|$ of set $\mathcal{H}_{\mathcal{I}}^{[i]}$ defined in Lemma 6 satisfies:

$$|\mathcal{H}_{\mathcal{I}}^{[i]}| \le n \cdot \binom{|Q|}{2} + \frac{n(n+1)}{2} \cdot |Q| \cdot |\mathcal{E}_{\mathcal{T}}| \tag{71}$$

Corollary 5: Let's define as $\mathcal{H}_{\mathcal{P}}^{[i]}$ the set of all hash key values which are in $\mathcal{H}$ but not in $\mathcal{H}_{\mathcal{I}}^{[i]}$, i.e., $\mathcal{H}_{\mathcal{P}}^{[i]} = \mathcal{H} - \mathcal{H}_{\mathcal{I}}^{[i]}$. The sets $\mathcal{H}$ and $\mathcal{H}_{\mathcal{I}}^{[i]}$ are defined as in Lemma 6. We refer to such set as the set of "possible" hash key values associated with query index i and set $\mathcal{H}$. Then, the cardinality $|\mathcal{H}_{\mathcal{P}}^{[i]}|$ of set $\mathcal{H}_{\mathcal{P}}^{[i]}$ satisfies:

$$|\mathcal{H}_{\mathcal{P}}^{[i]}| \ \ge \ 2^N - \zeta - \eta|Q| - n\binom{|Q|}{2} \tag{72}$$

where $\zeta = \frac{n(n-1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|^2$, $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$. Corollary 5 follows from Corollary 1 and Lemma 6.

3.8 On the statistical properties of the secrets and internal state of the MAGIC mode

We proceed with our proof establishing that every element $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$ is equal to the secret hash key value H with non-zero, uniformly bounded probability, given the queries $q^{[1]}, \ldots, q^{[i]}$, the responses $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$, and $\mathcal{N}(1), \ldots, \mathcal{N}(i)$. To do this, we view the queries, the responses and the knowledge that no security critical events occur up to $q^{[i]}$ as constraints imposed on a family of MAGIC oracles and investigate which combinations of hash key values and blinding permutations satisfy the constraints.

Definition 10: *Set of constraint satisfying permutations associated with a hash key value* $H_{\mathcal{P}}^{[i]}$. Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $q^{[1]}, \ldots, q^{[i]}$ and $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ be a set of queries and their responses satisfying the constraints $\mathcal{N}(1), \ldots, \mathcal{N}(i)$. Let $\mathcal{H}_{\mathcal{P}}^{[i]}$ be the set of possible hash key values associated with query index i, set $\mathcal{H}$, and the constraints, as defined in Corollary 5. Finally, let $H_{\mathcal{P}}^{[i]}$ be an element of $\mathcal{H}_{\mathcal{P}}^{[i]}$. The set of "constraint satisfying permutations" $\mathcal{C}_{H_{\mathcal{P}}^{[i]}} \in \mathcal{P}$, associated with the constraints and element $H_{\mathcal{P}}^{[i]}$ is defined as:

$$
\begin{aligned}
\mathcal{C}_{H^{[i]}_{\mathcal{P}}} = \{\ & \rho \in \mathcal{P},\ \rho : \{0,1\}^N \to \{0,1\}^N\ : \\
& \big(\,\forall\, q_s^{(i_s)} \in Q_s^{[i]} :\ \ \mathsf{MAC}^{\rho()}_{H^{[i]}_{\mathcal{P}}}(q_s^{(i_s)}) = \mathsf{MAC}^{\pi^*()}_{H}(q_s^{(i_s)})\,\big) \\
& \wedge\ \big(\,\forall\, q_v^{(i_v)} = (D^{(i_v)}, C_1^{(i_v)}, \ldots, C_n^{(i_v)}, T^{(i_v)})\ \in Q_v^{[i]} : \\
& \mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}(q_v^{(i_v)}, T_{th}) = \begin{cases}
(\mathsf{accept}, \mathsf{Str}(C_1^{+}, \ldots, C_n^{+}), T^{(i_v)}), \quad \text{if} \\
\qquad (C_1^{(i_v)}, \ldots, C_n^{(i_v)}) \in \theta_{cipher}(C_1^{+}, \ldots, C_n^{+}) \\
\qquad \wedge\ \ T^{(i_v)} = \mathsf{MAC}^{\pi^*()}_{H}(D^{(i_v)}, C_1^{+}, \ldots, C_n^{+}) \\
(\mathsf{accept}, \mathsf{Str}(C_1^{(i_v)}, \ldots, C_n^{(i_v)}), T^{+}), \quad \text{if} \\
\qquad T^{(i_v)} = \vartheta^{\pi^*()}_{H}(T^{+}, T_{th}) \\
\qquad \wedge\ \ T^{+} = \mathsf{MAC}^{\pi^*()}_{H}(D^{(i_v)}, C_1^{(i_v)}, \ldots, C_n^{(i_v)}) \\
(\mathsf{reject}, \perp, \perp), \text{otherwise}
\end{cases} \\
& \big)\,\}
\end{aligned}
\tag{73}
$$

where $Q_s^{[i]}$ and $Q_v^{[i]}$ are the sets of sign and verification queries issued by adversary $\mathcal{A}^*$ up to query $q^{[i]}$, respectively.

The three cases for the responses coming from $\mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$, in the definition above, correspond to the events that the ciphertext blocks of query $q_v^{(i_v)}$ are corrected, that the tag value of $q_v^{(i_v)}$ is corrected, and that no error correction is performed on $q_v^{(i_v)}$. In none of the cases the ECC functionality of MAGIC is exploited.

Our proof strategy is to demonstrate that for every possible Galois Hash key value $H^{[i]}_{\mathcal{P}} \in \mathcal{H}^{[i]}_{\mathcal{P}}$, which may be different than the hash key H, there exists an associated non-empty set $\mathcal{C}_{H^{[i]}_{\mathcal{P}}}$ of equally probable permutations $\rho() \in \mathcal{C}_{H^{[i]}_{\mathcal{P}}}$, any of which, if combined with the hash key value $H^{[i]}_{\mathcal{P}}$ forms a pair of "alternative" MAGIC mode oracles $\mathsf{MAC}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$ and $\mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$, where these oracles provide identical responses to adversary $\mathcal{A}^*$'s queries $q^{[1]}, \ldots q^{[i]}$ as the attacked oracles, given that no security critical events occur up to query $q^{[i]}$. Once this is demonstrated, the security bounds of the MAGIC mode follow from the computed cardinalities of the set of possible Galois hash keys $\mathcal{H}^{[i]}_{\mathcal{P}}$, and of the sets $\mathcal{C}_{H^{[i]}_{\mathcal{P}}}$ of the blinding permutations associated with every possible hash key $H^{[i]}_{\mathcal{P}} \in \mathcal{H}^{[i]}_{\mathcal{P}}$.

Proposition 2: The set $\mathcal{C}_{H^{[i]}_{\mathcal{P}}} \in \mathcal{P}$ defined in Definition 10 is non-empty:

$$\mathcal{C}_{H_{\mathcal{P}}^{[i]}} \neq \emptyset \tag{74}$$

The implication Proposition 2 has on the security of the MAGIC mode is that, at the time query $q^{[i]}$ completes, and if no security critical events occur up to this query, then there is a plurality of combinations of hash key values and blinding permutations, any of which may be the pair (H, π^*) from adversary $\mathcal{A}^*$'s point of view.

Proof of Proposition 2: To prove Proposition 2, we suggest a procedure for selecting permutations from the set $\mathcal{P}$, so that the selected permutations form the non-empty set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ of Definition 10 above.

Preliminaries

To do this, we first consider the set $Q_s^{[i]}$ of sign queries issued up to $q^{[i]}$. We denote a sign query $q_s^{(j)} \in Q_s^{[i]}$ by $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)})$. Index $j \in [1, |Q_s^{[i]}|]$ indicates the order of issue of $q_s^{(j)}$ inside the set $Q_s^{[i]}$. For this query, and for a Galois Hash key value $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$, we also consider the output $g_{H_{\mathcal{P}}^{[i]}}^{(j)}$ of the Galois Hash transformation computed using key $H_{\mathcal{P}}^{[i]}$:

$$g_{H_{\mathcal{P}}^{[i]}}^{(j)} = D^{(j)} + C_1^{(j)} \cdot H_{\mathcal{P}}^{[i]} + \ldots + C_n^{(j)} \cdot {H_{\mathcal{P}}^{[i]}}^n \tag{75}$$

Next, we consider the set $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$ of mappings from Galois Hash transformation outputs $g_{H_{\mathcal{P}}^{[i]}}^{(j)}$ to sign query responses $\mathsf{MAC}_H^{\pi^*()} \leftarrow \mathcal{O}(q_s^{(j)})$ defined for every index $j \in [1, |Q_s^{[i]}|]$:

$$\mathcal{S}_{H_{\mathcal{P}}^{[i]}} = \{ g_{H_{\mathcal{P}}^{[i]}}^{(j)} \rightarrow \mathcal{O}(q_s^{(j)}) : \ q_s^{(j)} \in Q_s^{[i]}, \ j \in [1, |Q_s^{[i]}|] \ \} \tag{76}$$

As there are $|Q_s^{[i]}|$ elements in $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$, there are $(2^N - |Q_s^{[i]}|)!$ permutations in $\mathcal{P}$ which specify the mappings of the set $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$. However, we cannot conclusively say that all $(2^N - |Q_s^{[i]}|)!$ permutations are elements of the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$, because we have not yet taken into account the constraints imposed on the verification query responses in Definition 10.

On compatible decrypted tag values

To take these constraints into account, we consider the set $Q_v^{[i]}$ of verification queries issued up to $q^{[i]}$. We refer to a verification query $q_v^{(j)} \in Q_v^{[i]}$ as $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}, T^{(j)})$. Index $j \in [1, |Q_v^{[i]}|]$ indicates the order of issue of $q_v^{(j)}$ inside the set $Q_v^{[i]}$. We also use the notation $g_{H_{\mathcal{P}}^{[i]}}^{(j)}$ to refer to the Galois

Hash transformation output computed from authenticated data $D^{(j)}$ and ciphertext blocks $C_1^{(j)}, \ldots, C_n^{(j)}$, using the hash key value $H_{\mathcal{P}}^{[i]}$.

If query $q_v^{(j)}$ satisfies Definition 10, and if the query cannot be corrected by the MAGIC mode, then the response $\mathcal{O}(q_v^{(j)})$, coming from the verification oracle, should be equal to $(\mathsf{reject}, \perp, \perp)$. We investigate which decrypted tag values support this property, i.e., cause procedure $\mathsf{Verify}()$ to return $(\mathsf{reject}, \perp, \perp)$. For this purpose, we define set $\mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$ as the set of the decrypted tag values, which are "compatible" with query $q_v^{(j)} \in Q_v^{[i]}$ and the hash key value $H_{\mathcal{P}}^{[i]}$, given that query $q_v^{(j)}$ is a reject query, or does not abuse the ECC functionality of MAGIC:

$$\mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} = \{ v \in \{0,1\}^N : \, \mathsf{HW}(\, (g_{H_{\mathcal{P}}^{[i]}}^{(j)} + v) \cdot H_{\mathcal{P}}^{[i]^{-k}} \,) > T_{th}, \;\; k \in [1,n] \} \tag{77}$$

Set $\mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$ contains all those values v which, if considered as results of the decryption of the tag value $T^{(j)}$, lead to the computation of syndrome values $(g_{H_{\mathcal{P}}^{[i]}}^{(j)} + v)$ and error location indicators $(g_{H_{\mathcal{P}}^{[i]}}^{(j)} + v) \cdot H_{\mathcal{P}}^{[i]^{-k}}$, $k \in [1,n]$, that cause procedure $\mathsf{LocateError}()$ to return $\perp$. Thus, if a blinding permutation maps any of the elements of $\mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$ to the tag value $T^{(j)}$, it never causes procedure $\mathsf{Verify}()$ to execute lines 8-9 of its pseudocode, thus abusing the ECC functionality of the MAGIC mode. By the definition of $\mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$, the cardinality of this set satisfies:

$$|\mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}| \geq 2^N - \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}| \tag{78}$$

We also refer to sets of decrypted tag values that are compatible with a plurality of verification queries. We consider a subset of verification queries $Q_{v,\mathsf{EQT}j}^{[i]} \subseteq Q_v^{[i]}$, which contains all those queries from the set $Q_v^{[i]}$, the tag values of which are equal to the tag value of query $q_v^{(j)}$:

$$Q_{v,\mathsf{EQT}j}^{[i]} = \{ \, q_v^{(j')} = (D^{(j')}, C_1^{(j')}, \ldots, C_n^{(j')}, T^{(j')}), \; q_v^{(j')} \in Q_v^{[i]} \; : \; T^{(j')} = T^{(j)} \, \} \tag{79}$$

We define the set $\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$ of decrypted tag values that are compatible with the subset $Q_{v,\mathsf{EQT}j}^{[i]}$ of $Q_v^{[i]}$, and with the hash key value $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$, as the set:

$$\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} = \bigcap_{j' : \, q_v^{(j')} \in Q_{v,\mathsf{EQT}j}^{[i]}} \mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j')} \tag{80}$$

By the definitions of sets $\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ and $\mathcal{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}$, the cardinality of set $\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ satisfies:

$$|\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}| \;\geq\; 2^N - \frac{n(n+1)}{2} \cdot |Q^{[i]}_v| \cdot |\mathcal{E}_{\mathcal{T}}| \tag{81}$$

Furthermore, from the query budget constraint $|Q| < \frac{2^{N+1}}{n(n+1)|\mathcal{E}_{\mathcal{T}}|+4}$ discussed in Section 3.2 and from inequality (81), it is deduced that any set $\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ defined in (80) for $q^{(j)}_v \in Q^{[i]}_v$ and $H^{[i]}_{\mathcal{P}} \in \mathcal{H}^{[i]}_{\mathcal{P}}$ is non-empty.

Set $\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ contains all those values v which, if considered as results of the decryption of the tag value $T^{(j)}$, lead to the computation of syndromes $(g^{(j)}_{H^{[i]}_{\mathcal{P}}} + v)$ and error location indicators $(g^{(j)}_{H^{[i]}_{\mathcal{P}}} + v) \cdot {H^{[i]}_{\mathcal{P}}}^{-k}$, $k \in [1, n]$ that cause procedure $\mathsf{LocateError}()$ to return $\perp$, in the event there is a set $Q^{[i]}_{v,\mathsf{EQT}j}$ of verification queries which include the same tag $T^{(j)}$. Thus, a blinding permutation needs to only map an element of $\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ to the tag value $T^{(j)}$ to ensure that lines 8-9 of the $\mathsf{Verify}()$ pseudocode are never executed during the completion of the queries of the set $Q^{[i]}_{v,\mathsf{EQT}j}$.

On compatible encrypted Galois Hash values

Next, we define the set $\mathcal{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ of encrypted Galois Hash values, which are "compatible" with the hash key value $H^{[i]}_{\mathcal{P}}$ and query $q^{(j)}_v \in Q^{[i]}_v$:

$$\mathcal{U}^{(j)}_{H^{[i]}_{\mathcal{P}}} = \{u \in \{0,1\}^N : \mathsf{HW}(u + T^{(j)}) > T_{th}\} \tag{82}$$

Set $\mathcal{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ contains all those values u which, if considered as results of the encryption of the Galois Hash transformation output computed from authenticated data $D^{(j)}$ and ciphertext blocks $C^{(j)}_1, \ldots, C^{(j)}_n$, cause procedure $\mathsf{CanCorrectParity}()$ to return false. Thus, if a blinding permutation maps the Galois Hash transformation output $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$ computed from query $q^{(j)}_v$ to any of the elements of the set $\mathcal{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$, it never causes procedure $\mathsf{Verify}()$ to return $(\mathsf{accept}, \mathfrak{s}_1, \mathfrak{s}_2)$, for any pair of strings $\mathfrak{s}_1, \mathfrak{s}_2$. This is because procedure $\mathsf{Verify}()$ never executes lines 12-13 of its pseudocode. By the definition of $\mathcal{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$, the cardinality of this set satisfies:

$$|\mathcal{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}| = 2^N - |\mathcal{E}_{\mathcal{T}}| \tag{83}$$

We further refer to sets of encrypted Galois Hash values that are compatible with a plurality of verification queries. We consider a subset of verification queries $Q^{[i]}_{v,H^{[i]}_{\mathcal{P}},\mathsf{EQG}j} \subseteq Q^{[i]}_v$, which contains all those queries from the set $Q_v^{[i]}$, the Galois Hash transformation outputs of which are equal to the Galois Hash transformation output of query $q_v^{(j)}$, provided that this output is computed using key $H_{\mathcal{P}}^{[i]}$:

$$Q^{[i]}_{v,H_{\mathcal{P}}^{[i]},\mathsf{EQG}j} = \{\ q_v^{(j')} \in Q_v^{[i]}\ :\ g^{(j')}_{H_{\mathcal{P}}^{[i]}} = g^{(j)}_{H_{\mathcal{P}}^{[i]}}\ \} \tag{84}$$

We define the set $\mathsf{U}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ of decrypted tag values that are compatible with the subset $Q^{[i]}_{v,H_{\mathcal{P}}^{[i]},\mathsf{EQG}j}$ of $Q_v^{[i]}$, and with the hash key value $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$, as the set:

$$\mathsf{U}^{(j)}_{H_{\mathcal{P}}^{[i]}} = \bigcap_{j'\colon q_v^{(j')} \in Q^{[i]}_{v,H_{\mathcal{P}}^{[i]},\mathsf{EQG}j}} \mathcal{U}^{(j')}_{H_{\mathcal{P}}^{[i]}} \tag{85}$$

By the definitions of sets $\mathsf{U}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ and $\mathcal{U}^{(j)}_{H_{\mathcal{P}}^{[i]}}$, the cardinality of set $\mathsf{U}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ satisfies:

$$|\mathsf{U}^{(j)}_{H_{\mathcal{P}}^{[i]}}| \ \geq\ 2^N - |Q_v^{[i]}| \cdot |\mathcal{E}_{\mathcal{T}}| \tag{86}$$

Furthermore, from the query budget constraint $|Q| < \frac{2^{N+1}}{n(n+1)|\mathcal{E}_{\mathcal{T}}|+4}$ discussed in Section 3.2 and from inequality (86), it is deduced that any set $\mathsf{U}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ defined by (85) is non-empty.

The procedure SelectPermutations()

Now that we defined the sets $\mathsf{V}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ and $\mathsf{U}^{(j)}_{H_{\mathcal{P}}^{[i]}}$, associated with the verification query $q_v^{(j)} \in Q_v$, we propose a method for computing the elements of a non-empty set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ satisfying Definition 10. This is procedure SelectPermutations() below. Procedure SelectPermutations() invokes procedures ObtainVMappings(), ObtainUMappings(), CombineMappings() and MappingsToPermutations(). Procedures ObtainVMappings()and ObtainUMappings() in turn, invoke procedures IsCiphertextCorrectable() and IsCiphertagCorrectable().

Procedures ObtainVMappings() and ObtainUMappings() support the core of the functionality of SelectPermutations(). Procedures ObtainVMappings() and ObtainUMappings() accept as input a verification query $q_v^{(j)}$, a hash key value $H_{\mathcal{P}}^{[i]}$, the sets of sign and verification queries $Q_s^{[i]}$ and $Q_v^{[i]}$, the secrets of the MAGIC mode H and $\pi^*()$, the attacked sign oracle $\mathsf{MAC}_H^{\pi^*()}()$, and a set of tag values T_{taken} or Galois Hash outputs g_{taken} for which mappings have already been defined.

```
$\mathsf{IsCiphertextCorrectable}(q_v^{(j)}, \mathsf{MAC}_H^{\pi^*()}())$
1. $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}, T^{(j)}) \leftarrow q_v^{(j)}$
2. **if exists** $(C_1^+, \ldots, C_n^+) \in \{0,1\}^{n \cdot N}$ **such that**
3.     $(C_1^{(j)}, \ldots, C_n^{(j)}) \in \theta_{cipher}(C_1^+, \ldots, C_n^+)$
4.     **and** $T^{(j)} = \mathsf{MAC}_H^{\pi^*()}(D^{(i_v)}, C_1^+, \ldots, C_n^+)$
5.     **then**
6.         **return** $(\mathsf{true}, \mathsf{Str}(C_1^+, \ldots, C_n^+))$
7.     **else**
8.         **return** $(\mathsf{false}, \perp)$
```

```
$\mathsf{IsTagCorrectable}(q_v^{(j)}, H, \pi^*(), \mathsf{MAC}_H^{\pi^*()}())$
1. $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}, T^{(j)}) \leftarrow q_v^{(j)}$
2. **if exists** $T^+ \in \{0,1\}^N$ **such that**
3.     $T^{(j)} = \vartheta_H^{\pi^*()}(T^+, T_{th})$
4.     **and** $T^+ = \mathsf{MAC}_H^{\pi^*()}(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)})$
5.     **then**
6.         **return** $(\mathsf{true}, \mathsf{Str}(T^+))$
7.     **else**
8.         **return** $(\mathsf{false}, \perp)$
```

```
$\mathsf{ObtainVMappings}(q_v^{(j)}, H_{\mathcal{P}}^{[i]}, Q_s^{[i]}, Q_v^{[i]}, H, \pi^*(), \mathsf{MAC}_H^{\pi^*()}(), T_{\mathsf{taken}})$
1. $\mathcal{G}_{H_{\mathcal{P}}^{[i]}} \leftarrow \{g_{H_{\mathcal{P}}^{[i]}}^{(j')},\ \ q_s^{(j')} \in Q_s^{[i]},\ \ j' \in [1, |Q_s^{(i)}|]\}$
2. $\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} \leftarrow$ set computed from (80) on $q_v^{(j)}$, $Q_v^{[i]}$, $H_{\mathcal{P}}^{[i]}$
3. $(b, \mathfrak{s}) \leftarrow \mathsf{IsCiphertextCorrectable}(q_v^{(j)}, \mathsf{MAC}_H^{\pi^*()}())$
4. **if** $b = \mathsf{true}$
5.     **return** $\emptyset$
6. **if** $T^{(j)} \in T_{\mathsf{taken}}$
7.     **return** $\emptyset$
8. **else**
9.     $T_{\mathsf{taken}} \leftarrow T_{\mathsf{taken}} \cup \{T^{(j)}\}$
10.    **return** $\{v \rightarrow T^{(j)} : v \in (\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} - \mathcal{G}_{H_{\mathcal{P}}^{[i]}})\}$
```

```
$\mathsf{ObtainUMappings}(q_v^{(j)}, H_{\mathcal{P}}^{[i]}, Q_s^{[i]}, Q_v^{[i]}, H, \pi^*(), \mathsf{MAC}_H^{\pi^*()}(), g_{\mathsf{taken}})$
1. $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}, T^{(j)}) \leftarrow q_v^{(j)}$
2. $g_{H_{\mathcal{P}}^{[i]}}^{(j)} \leftarrow D^{(j)} + C_1^{(j)} \cdot H_{\mathcal{P}}^{[i]} + \ldots + C_n^{(j)} \cdot H_{\mathcal{P}}^{[i]^n}$
3. $\mathcal{T}^{[i]} \leftarrow \{t = \mathsf{MAC}_H^{\pi^*()}(q_s^{(j')}),\ q_s^{(j')} \in Q_s^{[i]},\ \ j' \in [1, |Q_s^{[i]}|]\}$
4. $\mathsf{U}_{H_{\mathcal{P}}^{[i]}}^{(j)} \leftarrow$ set computed from (85) on $q_v^{(j)}$, $Q_v^{[i]}$, $H_{\mathcal{P}}^{[i]}$
5. $(b, \mathfrak{s}) \leftarrow \mathsf{IsCiphertextCorrectable}(q_v^{(j)}, \mathsf{MAC}_H^{\pi^*()}())$
6. **if** $b = \mathsf{true}$
7.     **return** $\emptyset$
8. $(b, \mathfrak{s}) \leftarrow \mathsf{IsTagCorrectable}(q_v^{(j)}, H, \pi^*(), \mathsf{MAC}_H^{\pi^*()}())$
9. **if** $b = \mathsf{true}$
10.    **return** $\emptyset$
11. **if** $g_{H_{\mathcal{P}}^{[i]}}^{(j)} \in g_{\mathsf{taken}}$
```

```
12.     return $\emptyset$
13. else
14.     $g_{\mathsf{taken}} \leftarrow g_{\mathsf{taken}} \cup \{g^{(j)}_{H^{[i]}_{\mathcal{P}}}\}$
15.     return $\{g^{(j)}_{H^{[i]}_{\mathcal{P}}} \rightarrow u : u \in (\mathsf{U}^{(j)}_{H^{[i]}_{\mathcal{P}}} - \mathcal{T}^{[i]})\}$
```

```
CombineMappings($\{m_1, \ldots, m_{n_m}\}, n_m$)
1. $M_C \leftarrow \{\{\mu_1, \ldots, \mu_{n_m}\} :\ \mu_1 \in m_1,\ \mu_2 \in m_2,\ \ldots, \mu_{n_m} \in m_{n_m},$
   $\forall\ \nu, \xi \in M_C,\ \nu \neq \xi,$
   $\forall\ \{(v_1 \rightarrow u_1), \ldots, (v_{n_m} \rightarrow u_{n_m})\} \in M_C,$
     $\nexists\ (i_m, i'_m) : v_{i_m} = v_{i'_m} \vee u_{i_m} = u_{i'_m}\}$
2. return $M_C$
```

```
MappingsToPermutations($m$)
1. $M_P \leftarrow \{\pi \in \mathcal{P}:\quad \forall (u, v) \in m : \pi(u) = v\}$
2.     return $M_P$
```

```
SelectPermutations($H^{[i]}_{\mathcal{P}}$, $H$, $\pi^*()$, $\mathsf{MAC}^{\pi^*()}_H()$, $\mathsf{Verify}^{\pi^*()}_H()$, $Q^{[i]}_s$, $Q^{[i]}_v$)
1. $C_{H^{[i]}_{\mathcal{P}}} \leftarrow \emptyset$
2. $\mathcal{S}_{H^{[i]}_{\mathcal{P}}} \leftarrow$ set computed from (76) on $H^{[i]}_{\mathcal{P}}$, $\mathsf{MAC}^{\pi^*()}_H()$, and $Q^{[i]}_s$
3. $g_{\mathsf{taken}} \leftarrow \{g^{(j')}_{H^{[i]}_{\mathcal{P}}},\ \ q^{(j')}_s \in Q^{[i]}_s,\ \ j' \in [1, |Q^{(i)}_s|]\}$
4. $T_{\mathsf{taken}} \leftarrow \{t = \mathsf{MAC}^{\pi^*()}_H(q^{(j')}_s),\ q^{(j')}_s \in Q^{[i]}_s,\ \ j' \in [1, |Q^{[i]}_s|]\}$
5. $M_{\mathsf{COMPAT}} \leftarrow \emptyset$
6. $n_{\mathsf{COMPAT}} \leftarrow 0$
7. for $j \leftarrow 1$ to $|Q^{[i]}_v|$
8.     do $q^{(j)}_v \leftarrow$ j-th element of $Q^{[i]}_v$
9.         $\mu_j \leftarrow$ ObtainVMappings($q^{(j)}_v, H^{[i]}_{\mathcal{P}}, Q^{[i]}_s, Q^{[i]}_v, H, \pi^*(), \mathsf{MAC}^{\pi^*()}_H(), T_{taken}$)
10.        if $\mu_j \neq \emptyset$
11.            $M_{\mathsf{COMPAT}} \leftarrow M_{\mathsf{COMPAT}} \cup \{\mu_j\}$
12.            $n_{\mathsf{COMPAT}} \leftarrow n_{\mathsf{COMPAT}} + 1$
13.        $\mu_j \leftarrow$ ObtainUMappings($q^{(j)}_v, H^{[i]}_{\mathcal{P}}, Q^{[i]}_s, Q^{[i]}_v, H, \pi^*(), \mathsf{MAC}^{\pi^*()}_H(), g_{taken}$)
14.        if $\mu_j \neq \emptyset$
15.            $M_{\mathsf{COMPAT}} \leftarrow M_{\mathsf{COMPAT}} \cup \{\mu_j\}$
16.            $n_{\mathsf{COMPAT}} \leftarrow n_{\mathsf{COMPAT}} + 1$
17. $M_{\mathsf{DISTINCT}} \leftarrow$ CombineMappings($M_{\mathsf{COMPAT}}, n_{\mathsf{COMPAT}}$)
18. for $j \leftarrow 1$ to $|M_{\mathsf{DISTINCT}}|$
19.        do $\xi_j \leftarrow$ j-th element of $M_{\mathsf{DISTINCT}}$
20.            $M_{\mathsf{PERM},j} \leftarrow$ MappingsToPermutations($\mathcal{S}_{H^{[i]}_{\mathcal{P}}} \cup \xi_j$)
21.            $C_{H^{[i]}_{\mathcal{P}}} \leftarrow C_{H^{[i]}_{\mathcal{P}}} \cup M_{\mathsf{PERM},j}$
22. if $M_{\mathsf{DISTINCT}} = \emptyset$
23.     $C_{H^{[i]}_{\mathcal{P}}} \leftarrow$ MappingsToPermutations($\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$)
24. return $C_{H^{[i]}_{\mathcal{P}}}$
```

This is how procedure ObtainVMappings() works. First, in lines 1-2 it performs variable initializations. Then, in line 3 it invokes procedure Is-

CiphertextCorrectable(). Procedure IsCiphertextCorrectable() checks whether the ciphertext blocks of query $q^{(j)}$ are correctable by the MAGIC mode. If this is the case, ObtainVMappings() exits in line 5, returning $\emptyset$. This is because, if no ECC function exploitation event occurs up to $q^{[i]}$, and if IsCiphertextCorrectable() returns true, then the input query $q_v^{[j]}$ to ObtainVMappings() is a corrupted replay of a sign query $q_s^{(j')}$. The mappings associated with all sign queries are separately taken into account by procedure SelectPermutations(), which invokes ObtainVMappings().

Next, procedure ObtainVMappings() checks whether the input tag $T^{(j)}$ is in the set T_{taken}. If this is the case, the mappings associated with tag value $T^{(j)}$ have already been taken into account. Such mappings have either been computed for a different verification query $q_v^{(j')} \in Q_{v,\mathsf{EQT}j}^{[i]}$ that includes tag $T^{(j)}$, or for a different sign query $q_s^{(j')}$, the response of which is $T^{(j)}$. In either of the cases, ObtainVMappings() returns $\emptyset$ in line 7. If $T^{(j)}$ is not in the set T_{taken}, procedure ObtainVMappings() updates T_{taken} in line 9, and subsequently, forms mappings between elements of the set $\mathsf{V}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ and the tag $T^{(j)}$. All elements of $\mathsf{V}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ are considered in the mappings, apart from those of set $\mathcal{G}_{H_{\mathcal{P}}^{[i]}}$ computed in line 1. This is the set of the Galois Hash outputs associated with the sign queries. These values do not map to $T^{(j)}$, because every sign query returns an element of T_{taken}, the flow of procedure ObtainVMappings() has determined that $T^{(j)} \notin T_{\mathsf{taken}}$, and the blinding permutation is a bijective function.

Procedure ObtainUMappings() is similar to ObtainVMappings(). In lines 1-4 it performs variable initializations. Then, in lines 5 and 8 it invokes procedures IsCiphertextCorrectable() and IsTagCorrectable() to check whether $q_v^{(j)}$ is a corrupted replay of a sign query $q_s^{(j')} \in Q_s^{[i]}$. If this is the case, ObtainUMappings() exits returning $\emptyset$, for the same reasons procedure ObtainVMappings() exits. This is done in lines 7 and 10 respectively. The reason why procedure ObtainUMappings() invokes IsTagCorrectable() and ObtainVMappings() doesn't is because, if a verification query $q_v^{(j)}$ contains a correctable tag and if such tag is corrected, then for this query procedure LocateError() of the MAGIC verification oracle must return $\perp$. The permutations that cause procedure LocateError() to return $\perp$ are those that map elements of $\mathsf{V}^{(j)}_{H_{\mathcal{P}}^{[i]}}$ to tag $T^{(j)}$. So, in order to support such queries, procedure ObtainVMappings() must return a non-empty set of mappings originating from elements of $\mathsf{V}^{(j)}_{H_{\mathcal{P}}^{[i]}}$. In this case, a check on the output of IsTagCorrectable() would cause procedure ObtainVMappings() to incorrectly return $\emptyset$, and is thus omitted.

Next, ObtainUMappings() checks whether the Galois Hash output $g^{(j)}_{H_{\mathcal{P}}^{[i]}}$ computed from the input ciphertext is in the set g_{taken}. If this is the case, this means that the mappings associated with this Galois Hash output have already been computed as part of responding to a different sign or verification query. In this case, $\mathsf{ObtainUMappings}()$ returns $\emptyset$ in line 12. If $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$ is not in the set g_{taken}, procedure $\mathsf{ObtainUMappings}()$ updates g_{taken} in line 14, and subsequently, forms mappings between $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$ and elements of the set $\mathsf{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$. Set $\mathsf{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ contains the encrypted Galois Hash outputs that are compatible with query $q^{(j)}$ and $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$. As in the execution of procedure $\mathsf{ObtainVMappings}()$, there are some mappings which are excluded. These are mappings between the Galois Hash output $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$ and the tags returned by sign queries. The tags returned by sign queries are elements of the set $\mathcal{T}^{[i]}$ computed in line 3. These mappings are excluded, because each sign query is associated with a mapping between an element of g_{taken} and an element of $\mathcal{T}^{[i]}$, $g^{(j)}_{H^{[i]}_{\mathcal{P}}} \notin g_{\mathsf{taken}}$, and the blinding permutation is a bijective function.

Another procedure invoked by $\mathsf{SelectPermutations}()$ is $\mathsf{CombineMappings}()$. Procedure $\mathsf{CombineMappings}()$ accepts as input a set $\{m_1, \ldots, m_{n_m}\}$ of n_m sets of mappings $m_1, \ldots, m_{n_m}$. Each element $m_j, j \in [1, n_m]$, is a set of alternative mappings either from the same origin or to the same destination. The elements of each set m_j are returned via either invocations to $\mathsf{ObtainVMappings}()$ or invocations to $\mathsf{ObtainUMappings}()$. Procedure $\mathsf{CombineMappings}()$ returns a new set M_C. Each element $\{\mu_1, \ldots, \mu_{n_m}\}$ of M_C is a set containing n_m mappings, where each mapping μ_j, $j \in [1, n_m]$, is obtained from its corresponding set m_j.

The elements of the set M_C are all possible sets of mappings produced this way that satisfy two conditions. First, no two elements of M_C are identical. Second, the mappings of each element $\{\mu_1, \ldots, \mu_{n_m}\}$ are supported by bijective functions, i.e., no two mappings of $\{\mu_1, \ldots, \mu_{n_m}\}$ are from the same origin or to the same destination. Essentially, procedure $\mathsf{CombineMappings}$ rearranges the sets of mappings computed via invocations to $\mathsf{ObtainVMappings}()$ and $\mathsf{ObtainUMappings}()$, so that each returned set $\{\mu_1, \ldots, \mu_{n_m}\}$ contains at most two mappings associated with every verification query of $Q^{[i]}_v$. A first mapping is obtained via an invocation to $\mathsf{ObtainVMappings}()$ and a second mapping is obtained via an invocation to $\mathsf{ObtainUMappings}()$. Furthermore, as we establish below, the mappings of each set $\{\mu_1, \ldots, \mu_{n_m}\}$ correspond to bijective functions which, if employed as blinding permutations, never cause ECC function exploitation or blind forgery events to occur.

We proceed with the flow of procedure $\mathsf{SelectPermutations}()$. Initially, procedure $\mathsf{SelectPermutations}()$ performs variable initializations, in lines 1-6. Then, in lines 7-16, it executes a for-loop, iterating over every verification query of the set $Q^{[i]}_v$. For every verification query of the set $Q^{[i]}_v$, it invokes $\mathsf{ObtainVMappings}()$, in line 9, and $\mathsf{ObtainUMappings}()$, in line 13. The nonempty responses are included into set M_{COMPAT}, in lines 11 and 15. When the for-loop finishes in line 17, set M_{COMPAT} and its cardinality n_{COMPAT} are passed into procedure CombineMappings().

The last part of procedure SelectPermutations() is the for-loop of lines 18-21. In this loop, each set of mappings ξ_j returned by procedure CombineMappings() is augmented with the elements of the set $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$ computed in line 2. This is the set of mappings between Galois Hash outputs and sign query responses computed from (76), and associated with the sign queries of $Q_s^{[i]}$. Each augmented set $\mathcal{S}_{H_{\mathcal{P}}^{[i]}} \cup \xi_j$ is passed into procedure MappingsToPermutations(). No two sets passed into MappingsToPermutations() are the same. Procedure MappingsToPermutations() returns the permutations of the set $M_{\mathsf{PERM},j} \subseteq \mathcal{P}$ that support the mappings $\mathcal{S}_{H_{\mathcal{P}}^{[i]}} \cup \xi_j$ passed as input. The union of all sets of permutations returned by MappingsToPermutations() is the set $C_{H_{\mathcal{P}}^{[i]}}$ returned by procedure SelectPermutations(). Additionally, lines 22-23 cover the corner case where no mappings are obtained via invocations to ObtainVMappings() and ObtainUMappings(). This may happen, for instance, if all queries are sign queries.

Establishing that SelectPermutations() returns a non-empty response

The correctness of Proposition 2 follows from the description of procedure SelectPermutations(). In what follows, we show that the set $C_{H_{\mathcal{P}}^{[i]}}$ returned by SelectPermutations() is non-empty if at least one of $Q_s^{[i]}$ and $Q_v^{[i]}$ is non-empty, and the query budget constraint of section 3.2 are satisfied. If $Q_s^{[i]}$ is non-empty, then procedure MappingsToPermutations() is always invoked either in line 20 or in line 23 of procedure SelectPermutations() returning some non-empty set of permutations.

On the other hand, if $Q_s^{[i]}$ is empty and $Q_v^{[i]}$ is non-empty then all verification queries must be reject queries, according to the assumption that no security critical events occur up to $q^{[i]}$. For all reject queries that include the same tag value $T^{(j)}$, there is one for which the invocation to procedure ObtainVMappings() returns a non-empty set. This is the invocation for which the tag value $T^{(j)}$ passed is not in the set T_{taken}. The check is performed in line 6 of ObtainVMappings(). Similarly, for reject queries that include ciphertext blocks of the same Galois Hash combination $g^{(j)}_{H_{\mathcal{P}}^{[i]}}$, there is one for which the invocation to procedure ObtainUMappings() returns a non-empty set. This is the invocation for which the value $g^{(j)}_{H_{\mathcal{P}}^{[i]}}$ is not in the set g_{taken}. The check is performed in line 11 of procedure ObtainUMappings(). Based on these observations, we establish that SelectPermutations() returns a non-empty response, if $Q_s^{[i]}$ is empty and $Q_v^{[i]}$ is non-empty, by showing that procedure CombineMappings(), invoked by SelectPermutations() in line 17 indeed forms a non-empty response M_C, as defined in line 1 of its pseudocode. This is when accepting as input a non-empty set $\{m_1, \ldots, m_{n_m}\}$

of cardinality n_m. Set $\{m_1, \ldots, m_{n_m}\}$ contains the non-empty responses from the invocations to ObtainVMappings() and ObtainUMappings() made by SelectPermutations() in lines 9 and 13 respectively.

For this purpose, we make use of the budget constraint of Section 3.2. The budget constraint (46) of 3.2 can be written as:

$$2^N - \frac{n(n+1)}{2} \cdot |Q| \cdot |\mathcal{E}_{\mathcal{T}}| > 2|Q| \tag{87}$$

Inequality (87), when combined with inequalities (81) and (86) results in the relations $|\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}| \geq 2|Q|$ and $|\mathsf{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}| \geq 2|Q|$, which hold for every $q_v^{(j)} \in Q_v^{[i]}$. From the relations, it follows that the responses returned from the invocations to ObtainVMappings() and ObtainUMappings(), are not only non-empty, but have cardinality greater than $2|Q_v| + |Q_s|$, or when $Q_s^{[i]}$ is empty, greater than $2|Q|$.

Since these responses form the non-empty sets $m_1, \ldots, m_{n_m}$ passed as input to CombineMappings(), and $n_m \leq 2|Q|$, it can be deduced that procedure CombineMappings() succeeds in forming at least one set that satisfies the two conditions of its pseudocode. Indeed, the first condition, which dictates that every two returned sets of mappings should be different, is trivially satisfied if procedure CombineMappings() returns only the unique sets that satisfy its second condition. The second condition, which dictates that the mappings in each returned set should be the mappings of a bijective function, is also satisfied given the query budget constraint discussed.

The mappings of each set m_i, $i \in [1, n_m]$ from $\{m_1, \ldots, m_{n_m}\}$, which are obtained via invocations to ObtainVMappings(), are from origins which may be possibly overlapping with the origins of mappings of another set m_j, $j \in [1, n_m]$, $j \neq i$, also obtained via invocations to ObtainVMappings(). The mappings of sets m_i, m_j, however, should have different destinations. Similarly, the mappings of each set m_k, $k \in [1, n_m]$ from $\{m_1, \ldots, m_{n_m}\}$, obtained via invocations to ObtainUMappings() map origins to destinations which may be possibly overlapping with the destinations of mappings of another set m_l, $l \in [1, n_m]$, $k \neq l$, also obtained via invocations to ObtainUMappings(). The mappings of sets m_k, m_l should have different origins. Because of the possibility of these origin and destination overlaps, we need condition $|m_{i_m}| > 2|Q| \geq n_m$ to hold. It is easy to see that this condition guarantees that, for every $i_m \in [1, n_m]$, there is always a different source or destination to select from, when forming the mappings of at least one set $\{\mu_1, \ldots, \mu_{n_m}\}$ included in the response of CombineMappings(). This is true even if all sets of $\{m_1, \ldots, m_{n_m}\}$ contain mappings with origins or destinations which are in common between sets. Therefore, the response from SelectPermutations() is always non-empty.

Completing the proof

We complete the proof of Proposition 2, showing that all permutations returned by procedure $\mathsf{SelectPermutations}()$ satisfy Definition 10. First, all permutations returned by $\mathsf{SelectPermutations}()$ support the mappings of the set $\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$. This is the set that maps the Galois hash outputs computed from the ciphertext blocks of the queries of $Q^{[i]}_s$ to their corresponding responses coming from the oracle $\mathsf{MAC}^{\pi^*()}_H()$. The Galois hash outputs are computed using key $H^{[i]}_{\mathcal{P}}$. The permutations support the mappings of $\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$ because such mappings are included in every invocation to procedure $\mathsf{MappingsToPermutations}()$, in lines 20 and 23 of the pseudocode. Therefore, any permutation $\rho()$ in the set $C_{H^{[i]}_{\mathcal{P}}}$ returned by $\mathsf{SelectPermutations}()$, if combined with the hash key $H^{[i]}_{\mathcal{P}}$, forms an oracle $\mathsf{MAC}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$ that responds to the sign queries of the set $Q^{[i]}_s$ in an identical manner as the attacked oracle $\mathsf{MAC}^{\pi^*()}_H()$.

Next, we consider the verification queries of the set $Q^{[i]}_v$. As no security critical events occur up to query $q^{[i]}$, the queries of $Q^{[i]}_v$ are one of queries that include correctable ciphertext blocks, queries that include correctable tags, or reject queries.

If a verification query $q^{(j)}_v = (D^{(j)}, C^{(j)}_1, \dots, C^{(j)}_n, T^{(j)}) \in Q^{[i]}_v$ includes ciphertext blocks $C^{(j)}_1, \dots, C^{(j)}_n$ that are correctable, then such query is a corrupted replay of a sign query $q^{(j')}_s = (D^{(j)}, C^{(j')}_1, \dots, C^{(j')}_n) \in Q^{[i]}_s$. As the tag returned by $\mathsf{MAC}^{\pi^*()}_H()$ on input $(D^{(j)}, C^{(j')}_1, \dots, C^{(j')}_n)$ is $T^{(j)}$, every permutation $\rho() \in C_{H^{[i]}_{\mathcal{P}}}$ maps the Galois Hash transformation of $(D^{(j)}, C^{(j')}_1, \dots,$ $C^{(j')}_n)$ to $T^{(j)}$. Similarly, every verification oracle $\mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$, formed by combining $\rho()$ with the hash key $H^{[i]}_{\mathcal{P}}$, decrypts the supplied tag value $T^{(j)}$ to the Galois Hash of $(D^{(j)}, C^{(j')}_1, \dots, C^{(j')}_n)$. As only one block from $C^{(j')}_1, \dots, C^{(j')}_n$ differs from a corresponding block from $C^{(j)}_1, \dots, C^{(j)}_n$ by at most T_{th} bits, the flow of every procedure $\mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$ proceeds exactly as the flow of $\mathsf{Verify}^{\pi^*()}_H()$ successfully correcting blocks $C^{(j)}_1, \dots, C^{(j)}_n$.

If a verification query $q^{(j)}_v = (D^{(j)}, C^{(j)}_1, \dots, C^{(j)}_n, T^{(j)}) \in Q^{[i]}_v$ includes a tag value $T^{(j)}$ which is correctable, then such query is also a corrupted replay of a sign query $q^{(j')}_s \in Q^{[i]}_s$. In this case, the sign query $q^{(j')}_s$ includes the same authenticated data and ciphertext blocks $(D^{(j)}, C^{(j)}_1, \dots, C^{(j)}_n)$ as query $q^{(j)}_v$. The response coming from $\mathsf{MAC}^{\pi^*()}_H()$ on input $q^{(j')}_s$ is denoted by $T^{(j')}$. On input $q^{(j)}_v$, the verification oracle $\mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$ decrypts the supplied tag value $T^{(j)}$ to an element from the set $\mathsf{V}^{(j)}_{H^{[i]}_{\mathcal{P}}}$. Such decrypted tag value causes procedure $\mathsf{LocateError}()$ invoked by $\mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$ to return $\perp$. Thus, the flow of procedure $\mathsf{Verify}^{\rho()}_{H^{[i]}_{\mathcal{P}}}()$ reaches line 11. As the mapping between the Galois Hash value computed from $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)})$ and $T^{(j')}$ is supported by $\rho()$, being a mapping associated with a sign query, procedure $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$ invokes $\mathsf{CanCorrectParity}()$ with inputs $T^{(j)}$, $T^{(j')}$ and T_{th} in line 12. Since $T^{(j')}$ is the corrected version of tag $T^{(j)}$, the Hamming weight distance between $T^{(j)}$ and $T^{(j')}$ is bounded by T_{th}. Thus, procedure $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$ corrects the tag value $T^{(j)}$ in the same way as procedure $\mathsf{Verify}_{H}^{\pi^*()}()$ does.

If a verification query $q_v^{(j)} \in Q_v^{[i]}$ is rejected by oracle $\mathsf{Verify}_{H}^{\pi^*()}()$, it is also rejected by oracle $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$. On input $q_v^{(j)}$, the verification oracle $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$ decrypts the supplied tag value $T^{(j)}$ to an element from the set $\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$. Such decrypted tag value causes procedure $\mathsf{LocateError}()$ invoked by $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$ to return $\perp$. Thus, the flow of procedure $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$ reaches line 11. In this line, oracle $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$ encrypts the Galois Hash output, computed from the authenticated data and ciphertext blocks of $q_v^{(j)}$, to an element of the set $\mathsf{U}_{H_{\mathcal{P}}^{[i]}}^{(j)}$. Such encrypted Galois Hash value causes procedure $\mathsf{CanCorrectParity}()$ to return false. Thus oracle $\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}()$ returns $(\mathsf{reject}, \perp, \perp)$. We conclude that all permutations returned by SelectPermutations() satisfy Definition 10. Hence:

$$C_{H_{\mathcal{P}}^{[i]}} \subseteq \mathcal{C}_{H_{\mathcal{P}}^{[i]}} \tag{88}$$

Since $C_{H_{\mathcal{P}}^{[i]}} \neq \emptyset$, it must also hold that $\mathcal{C}_{H_{\mathcal{P}}^{[i]}} \neq \emptyset$. This completes the proof of Propositon 2.

□

Corollary 6: Let $\mathsf{MAC}_{H}^{\pi^*()}()$ and $\mathsf{Verify}_{H}^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $q^{[1]}, \ldots, q^{[i]}$ and $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ be a set of queries and their responses satisfying the constraints $\mathcal{N}(1), \ldots, \mathcal{N}(i)$. Let $\mathcal{H}_{\mathcal{P}}^{[i]}$ be a set of possible hash key values associated with query index i, set $\mathcal{H}$, and the constraints, as defined in Corollary 5. Let $H_{\mathcal{P}}^{[i]}$ an element of $\mathcal{H}_{\mathcal{P}}^{[i]}$. Finally let $\mathcal{C}_{H_{\mathcal{P}}^{[i]}} \in \mathcal{P}$ be the set of constraint satisfying permutations associated with the constraints and element $H_{\mathcal{P}}^{[i]}$, as defined by Definition 10. Then the response $C_{H_{\mathcal{P}}^{[i]}}$ of procedure $\mathsf{SelectPermutations}()$ on inputs $H_{\mathcal{P}}^{[i]}$, H, $\pi^*()$, $\mathsf{MAC}_{H}^{\pi^*()}(), \mathsf{Verify}_{H}^{\pi^*()}(), Q_s^{[i]}$ and $Q_v^{[i]}$ is equal to the set of constraint satisfying permutations $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$:

$$C_{H_{\mathcal{P}}^{[i]}} = \mathcal{C}_{H_{\mathcal{P}}^{[i]}} \tag{89}$$

where $Q_s^{[i]}$ and $Q_v^{[i]}$ are the sets of sign and verification queries issued by adversary $\mathcal{A}^*$ up to query $q^{[i]}$.

Proof of Corollary 6: It is sufficient to show that there are no permutations that are not returned by procedure $\mathsf{SelectPermutations}()$, and which satisfy Definition 10. Permutations that do not support the mappings of the set $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$, defined by (76), are not in the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ and are not returned by $\mathsf{SelectPermutations}()$ either. So the discussion will focus only on permutations that support the mappings of the $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$.

Suppose some permutation $\rho()$ satisfies $\rho() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ and $\rho() \notin C_{H_{\mathcal{P}}^{[i]}}$. If all queries are sign queries or verification queries that include correctable ciphertext blocks, then it is not possible for $\rho()$ to simultaneously satisfy $\rho() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ and $\rho() \notin C_{H_{\mathcal{P}}^{[i]}}$, as $\rho()$ supports the mappings of the set $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$ and, as such, is a member of both sets $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ and $C_{H_{\mathcal{P}}^{[i]}}$.

Now, let's suppose that, in addition to the above assumption, at least one verification query $q_v^{(j)} \in Q_v^{[i]}$ includes a tag $T^{(j)}$ which is correctable. Let's also assume that $T^{(j)}$ is not the response coming from a sign query to oracle $\mathsf{MAC}_H^{\pi^*()}()$. For this query, permutation $\rho()$ maps an element from the set $\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$ to $T^{(j)}$. The only elements from $\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$ which are excluded by procedure $\mathsf{SelectPermutations}()$ when forming mappings, are those values which are members of set $\mathcal{G}_{H_{\mathcal{P}}^{[i]}}$, containing the Galois Hash values computed from the ciphertext blocks of the sign queries. If $\rho()$ is not in the set $C_{H_{\mathcal{P}}^{[i]}}$, then $\rho()$ must be necessarily mapping an element $g \in \mathcal{G}_{H_{\mathcal{P}}^{[i]}}$ to $T^{(j)}$. In this case, $\rho()$ is not a bijective function, as element g maps to some value other than $T^{(j)}$, according to the hypothesis that $T^{(j)}$ is not a response coming from a sign query, and to $T^{(j)}$ simultaneously. On the other hand, if $T^{(j)}$ is a response coming from a sign query to oracle $\mathsf{MAC}_H^{\pi^*()}()$, then the mapping to $T^{(j)}$ is included in the set $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$. Since $\rho()$ supports all mappings of $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$, $\rho()$ must be in the set $C_{H_{\mathcal{P}}^{[i]}}$ as well as $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$.

Next, we assume that, in addition to all above assumptions, at least one verification query $q_v^{(j)} \in Q_v^{[i]}$ is a query rejected by oracle $\mathsf{Verify}_H^{\pi^*()}()$. We also assume that neither the Galois Hash value $g_{H_{\mathcal{P}}^{[i]}}^{(j)}$ computed using key $H_{\mathcal{P}}^{[i]}$ on the authenticated data and ciphertext blocks of $q_v^{(j)}$, nor the tag value $T^{(j)}$ of $q_v^{(j)}$ collide with any sign query. Permutation $\rho()$ must be mapping an element from the set $(\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} - \mathcal{G}_{H_{\mathcal{P}}^{[i]}})$ to $T^{(j)}$ for the reasons discussed above. Such mapping is also supported by every permutation in $C_{H_{\mathcal{P}}^{[i]}}$. Additionally, it must be mapping the Galois Hash value $g_{H_{\mathcal{P}}^{[i]}}^{(j)}$, associated with $q_v^{(j)}$, to an element from the set of compatible encrypted

Galois Hash values $\mathsf{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$. The only elements from $\mathsf{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$ which are excluded by procedure SelectPermutations() are those values of set $\mathcal{T}^{[i]}$, which includes the tags returned by oracle $\mathsf{MAC}^{\pi^*()}_H()$, when processing the sign queries of set $Q^{[i]}_s$. If $\rho()$ is not in the set $C_{H^{[i]}_{\mathcal{P}}}$, then $\rho()$ must be necessarily mapping $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$ to an element of $\mathcal{T}^{[i]}$. In this case $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$ would map to two different values simultaneously, on in $\mathcal{T}^{[i]}$ and one not in $\mathcal{T}^{[i]}$, according to the hypothesis that $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$ is not the Galois hash output of any sign query. If on the other hand, either the Galois Hash value $g^{(j)}_{H^{[i]}_{\mathcal{P}}}$, or the tag value $T^{(j)}$ of $q^{(j)}_v$ collide with a sign query, then the corresponding mappings must be included in the set $\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$. In these cases, $\rho()$ must be in the set $C_{H^{[i]}_{\mathcal{P}}}$ as well as $\mathcal{C}_{H^{[i]}_{\mathcal{P}}}$, since $\rho()$ supports the mappings of the $\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$.

□

Corollary 7: The cardinality of the set $\mathcal{C}_{H^{[i]}_{\mathcal{P}}} \in \mathcal{P}$ of Corollary 6 satisfies:

$$(2^N - |Q^{[i]}_s| - 2|Q^{[i]}_v|)! \;\le\; |\mathcal{C}_{H^{[i]}_{\mathcal{P}}}| \;\le\; (2^N - |Q^{[i]}_s|)! \qquad (90)$$

Proof of Corollary 7: A superset of $\mathcal{C}_{H^{[i]}_{\mathcal{P}}}$ is the set of all permutations that support the mappings of the set $\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$. As there are $(2^N - |Q^{[i]}_s|)!$ such permutations, it must hold that $|\mathcal{C}_{H^{[i]}_{\mathcal{P}}}| \le (2^N - |Q^{[i]}_s|)!$

The lower bound is established by observing the flow of procedure SelectPermutations(). The cardinality $|C_{H^{[i]}_{\mathcal{P}}}|$ of the response of SelectPermutations() is minimized when procedure CombineMappings() invoked by SelectPermutations() returns set M_C containing a single set of mappings. Furthermore, the cardinality $|C_{H^{[i]}_{\mathcal{P}}}|$ is minimized when such set of mappings has the highest possible number of elements, which is $2|Q^{[i]}_v|$. The number of permutations which support the $|Q^{[i]}_s|$ mappings of the set $\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$ and the $2|Q^{[i]}_v|$ of the response coming from procedure CombineMappings() is $(2^N - |Q^{[i]}_s| - 2|Q^{[i]}_v|)!$ Hence, it must hold that $|\mathcal{C}_{H^{[i]}_{\mathcal{P}}}| \ge (2^N - |Q^{[i]}_s| - 2|Q^{[i]}_v|)!$ This concludes the proof of Corollary 7.

□

Corollary 8: Let $\mathsf{MAC}^{\pi^*()}_H()$ and $\mathsf{Verify}^{\pi^*()}_H()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $q^{[1]}, \ldots, q^{[i]}$ and $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ be a set of queries and their responses satisfying the constraints $\mathcal{N}(1), \ldots, \mathcal{N}(i)$. Let $\mathcal{H}^{[i]}_{\mathcal{P}}$ be a set of possible hash key values associated with query index i, set $\mathcal{H}$, and the constraints, as defined in Corollary 5. Finally let $\mathcal{C}_{H^{[i]}_{\mathcal{P}}} \in \mathcal{P}$ be the set of constraint satisfying permutations associated with the constraints and every element $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$, as defined by Definition 10. Then the following two events are identical:

$$\begin{aligned} & \bigvee_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \left(H = H_{\mathcal{P}}^{[i]} \ \wedge \ \pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}} \right) \ = \\ & = \ q^{[1]}, \ldots, q^{[i]}, \ \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}), \ \mathcal{N}(1), \ldots, \mathcal{N}(i) \end{aligned} \tag{91}$$

Corollary 8 follows directly from Definition 10 and Proposition 2.

Based on Proposition 2 and Corollaries 6, 7 and 8, Lemma 7 below establishes that the secret hash key H of the MAGIC mode can take any value in the set $\mathcal{H}_{\mathcal{P}}^{[i]}$ with non-zero probability and, furthermore, the probability that H is equal to some element $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$ is uniformly bounded.

Lemma 7: *On the probability distribution of the hash key value H at the time query $q^{[i]}$ completes, if no security critical events occur up to this query.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $i \in |Q_s| + |Q_v|$ be a query index such that no security critical events occur up to query $q^{[i]}$, and knowledge about queries $q^{[1]}, \ldots, q^{[i]}$ and their associated responses $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ is available. Let also $\mathcal{H}_{\mathcal{P}}^{[i]}$ be a set of possible hash key values associated with query index i and set $\mathcal{H}$, as defined in Corollary 5. Then, the probability that the secret hash key H of the MAGIC mode is equal to some element $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$ is uniformly bounded across all elements of $\mathcal{H}_{\mathcal{P}}^{[i]}$:

$$\begin{aligned} & \mathrm{Prob}[\, H = H_{\mathcal{P}}^{[i]} \mid \ q^{[1]}, \ldots, q^{[i]}, \ \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}), \ \mathcal{N}(1), \ldots, \mathcal{N}(i)] \\ & \qquad \leq \ \frac{1}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}}, \ \forall \, H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]} \end{aligned} \tag{92}$$

where $\zeta = \frac{n(n-1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|^2$, $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$.

Proof of Lemma 7: To prove Lemma 7, we make use Corollaries 7 and 8 which, in turn, are based on Proposition 2. We begin the proof by considering the event $H = H_{\mathcal{P}}^{[i]}$ happening in conjunction with all possible mutually exclusive events $\pi^*() = \rho()$, $\rho \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}$:

$$\begin{aligned}
&\text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \mid \; q^{[1]}, \ldots, q^{[i]},\, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}),\, \mathcal{N}(1), \ldots, \mathcal{N}(i)] \\
&= \sum_{\rho() \,\in\, \mathcal{C}_{H_{\mathcal{P}}^{[i]}}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \;\wedge\; \pi^*() = \rho() \;\mid q^{[1]}, \ldots, \mathcal{N}(i)] = \\
&= \sum_{\rho() \,\in\, \mathcal{C}_{H_{\mathcal{P}}^{[i]}}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \;\wedge\; \pi^*() = \rho() \mid \bigvee_{h \,\in\, \mathcal{H}_{\mathcal{P}}^{[i]}} (H = h \;\wedge\; \pi^*() \in \mathcal{C}_h)\,]
\end{aligned} \tag{93}$$

where in the last step we made use of Corollary 8. We proceed by taking into account the fact that H and $\pi^*()$ are independently drawn.

$$\begin{aligned}
&\sum_{\rho() \,\in\, \mathcal{C}_{H_{\mathcal{P}}^{[i]}}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \;\wedge\; \pi^*() = \rho() \mid \bigvee_{h \,\in\, \mathcal{H}_{\mathcal{P}}^{[i]}} (H = h \;\wedge\; \pi^*() \in \mathcal{C}_h)\,] \\
&= \sum_{\rho() \,\in\, \mathcal{C}_{H_{\mathcal{P}}^{[i]}}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \mid \bigvee_{h \,\in\, \mathcal{H}_{\mathcal{P}}^{[i]}} (H = h \;\wedge\; \pi^*() \in \mathcal{C}_h)\,] \cdot \\
&\qquad\qquad \text{Prob}[\, \pi^*() = \rho() \mid H = H_{\mathcal{P}}^{[i]} \;\wedge\; (\textstyle\bigvee_{h \,\in\, \mathcal{H}_{\mathcal{P}}^{[i]}} (H = h \;\wedge\; \pi^*() \in \mathcal{C}_h))\,] \\
&= \sum_{\rho() \,\in\, \mathcal{C}_{H_{\mathcal{P}}^{[i]}}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \;\mid\; H \in \mathcal{H}_{\mathcal{P}}^{[i]}\,] \cdot \\
&\qquad\qquad \text{Prob}[\, \pi^*() = \rho() \mid H = H_{\mathcal{P}}^{[i]} \;\wedge\; \pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}\,] \\
&= \sum_{\rho() \,\in\, \mathcal{C}_{H_{\mathcal{P}}^{[i]}}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \;\mid\; H \in \mathcal{H}_{\mathcal{P}}^{[i]}\,] \;\cdot\; \text{Prob}[\, \pi^*() = \rho() \mid \pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}\,] \\
&= \quad \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \;\mid\; H \in \mathcal{H}_{\mathcal{P}}^{[i]}\,]
\end{aligned} \tag{94}$$

Next, we make use of the fact that H is uniformly drawn.

$$\text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \;\mid\; H \in \mathcal{H}_{\mathcal{P}}^{[i]}\,] = \frac{1}{|\mathcal{H}_{\mathcal{P}}^{[i]}|} \tag{95}$$

The correctness of Lemma 7 follows directly from (93), (94), (95) and Corollary 5.

□

Lemma 8: *On the probability distribution of the blinding permutation $\pi^*()$ at the time query $q^{[i]}$ completes, if no security critical events occur up to this query.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $i \in |Q_s| + |Q_v|$ be a query index such that no security critical events occur up to query $q^{[i]}$, and knowledge about queries $q^{[1]}, \ldots, q^{[i]}$ and their associated responses $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ is available. Then, the probability that the blinding permutation $\pi^*()$ of the MAGIC mode is equal to some permutation $\rho() \in \mathcal{P}$ is uniformly bounded across all elements of $\mathcal{P}$:

$$\begin{aligned}\text{Prob}[\,\pi^*() = \rho() \mid\ & q^{[1]}, \ldots, q^{[i]},\, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}),\, \mathcal{N}(1), \ldots, \mathcal{N}(i)] \\ \le\ & \frac{1}{(2^N - 2|Q|)!},\ \ \forall\, \rho() \in \mathcal{P}\end{aligned} \tag{96}$$

Proof of Lemma 8: Let $\mathcal{H}_{\mathcal{P}}^{[i]}$ be a set of possible hash key values associated with query index i and set $\mathcal{H}$, as defined in Corollary 5. We consider the event $\pi^*() = \rho()$ happening in conjunction with all mutually exclusive events $H = H_{\mathcal{P}}^{[i]}$, $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$. We also make use of Corollary 8:

$$\begin{aligned}&\text{Prob}[\,\pi^*() = \rho() \mid\ q^{[1]}, \ldots, q^{[i]},\, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}),\, \mathcal{N}(1), \ldots, \mathcal{N}(i)] \\ &= \sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \text{Prob}[\,H = H_{\mathcal{P}}^{[i]}\ \wedge\ \pi^*() = \rho() \mid \bigvee_{h \,\in\, \mathcal{H}_{\mathcal{P}}^{[i]}} (H = h\ \wedge\ \pi^*() \in \mathcal{C}_h)\,]\end{aligned} \tag{97}$$

Next, we make use of the fact that H and $\pi^*()$ are independently drawn:

$$\begin{aligned}&\sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \text{Prob}[\,H = H_{\mathcal{P}}^{[i]}\ \wedge\ \pi^*() = \rho() \mid \bigvee_{h \,\in\, \mathcal{H}_{\mathcal{P}}^{[i]}} (H = h\ \wedge\ \pi^*() \in \mathcal{C}_h)\,] \\ &= \sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \text{Prob}[\,H = H_{\mathcal{P}}^{[i]}\ \mid\ H \in \mathcal{H}_{\mathcal{P}}^{[i]}\,]\ \cdot\ \text{Prob}[\,\pi^*() = \rho() \mid \pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}\,]\end{aligned} \tag{98}$$

Since $\pi^*()$ is uniformly drawn and the cardinality of set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$, $i \in [1, |Q|]$ satisfies the bounds of Corollary 7, it must hold that:

$$\sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \mid H \in \mathcal{H}_{\mathcal{P}}^{[i]} \,] \cdot \text{Prob}[\, \pi^*() = \rho() \mid \pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}} \,]$$

$$\leq \sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \text{Prob}[\, H = H_{\mathcal{P}}^{[i]} \mid H \in \mathcal{H}_{\mathcal{P}}^{[i]} \,] \cdot \frac{1}{(2^N - |Q_s^{[i]}| - 2|Q_v^{[i]}|)!}$$

$$\leq \frac{1}{(2^N - 2|Q|)!} \tag{99}$$

where $Q_s^{[i]}$ and $Q_v^{[i]}$ are the sign and verification queries issued by adversary $\mathcal{A}^*$ up to query $q^{[i]}$. The correctness of Lemma 8 follows directly from relations (97), (98) and (99).

□

The next lemmas demonstrate that the internal state of the MAGIC mode is uniformly bounded as well.

Lemma 9: *On the probability distribution of a decrypted tag value* $\pi^{*^{-1}}(t)$*, associated with a tag value t that has never been queried before, given that no security critical events occur up to query* $q^{[i]}$. Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $i \in |Q_s| + |Q_v|$ be a query index such that no security critical events occur up to query $q^{[i]}$, and knowledge about queries $q^{[1]}, \ldots, q^{[i]}$ and their associated responses $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ is available. Let $Q_s^{[i]}$ and $Q_v^{[i]}$ be the sets of sign and verification queries issued by $\mathcal{A}^*$ up to query $q^{[i]}$. Finally, let t be a response from the sign oracle $\mathsf{MAC}_H^{\pi^*()}()$, such that there is no sign query $q_s^{(j)} \in Q_s^{[i]}$ for which $\mathcal{O}(q_s^{(j)}) = t$ and no verification query $q_v^{(j)} = (D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}, T^{(j)}) \in Q_v^{[i]}$ for which $T^{(j)} = t$. Then, the probability that the decrypted tag value $\pi^{*^{-1}}(t)$ is equal to some value g is uniformly bounded across all $g \in \{0,1\}^N$:

$$\text{Prob}[\, \pi^{*^{-1}}(t) = g \mid \; q^{[1]}, \ldots, q^{[i]},\, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}),\, \mathcal{N}(1), \ldots, \mathcal{N}(i)] \leq \frac{1}{2^N - 2|Q|},\ \forall\, g \in \{0,1\}^N \tag{100}$$

Proof of Lemma 9: Let $\mathcal{H}_{\mathcal{P}}^{[i]}$ be a set of possible hash key values associated with query index i and set $\mathcal{H}$, as defined in Corollary 5. For the proof of Lemma 9, we consider the event $\pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}$, $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$, happening in conjunction with all possible mutually exclusive events $\pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$, where $j \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$. By $\mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$ we mean the j-th set of permutations returned by procedure MappingsToPermutations(), when the permutations of the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ are computed. The total number of sets of permutations returned by procedure MappingsToPermutations() is denoted by $|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|$. When the set $Q_v^{[i]}$ of verification queries is non-empty, $\mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$ is equal to the set $M_{\mathsf{PERM},j}$ returned in line 20 of the pseudocode of procedure SelectPermutations(). Furthermore, $|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|$ is equal to the cardinality of the set M_{DISTINCT} returned by procedure CombineMappings() in line 17 of SelectPermutations(). When $Q_v^{[i]}$ is empty, $\mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$ is equal to the set returned by MappingsToPermutations() in line 23 of the pseudocode of SelectPermutations(), and $|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}| = 1$.

From the flow of procedure SelectPermutations(), it follows that each set $\mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$ includes all permutations that support, either the set of mappings denoted by $\mathcal{S}_{H_{\mathcal{P}}^{[i]}} \cup \xi_j$ in line 20 of the pseudocode of SelectPermutations(), or the set of mappings $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$. We refer to such mappings as $f_{H_{\mathcal{P}}^{[i]},j}$. It holds that $|f_{H_{\mathcal{P}}^{[i]},j}| \leq |Q_s^{[i]}| + 2|Q_v^{[i]}|$. We proceed with the proof by introducing the events $\pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$, $j \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ in the expression we need to bound.

$$\text{Prob}[\,{\pi^*}^{-1}(t) = g \mid q^{[1]}, \ldots, q^{[i]},\, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}),\, \mathcal{N}(1), \ldots, \mathcal{N}(i)] = \sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \sum_{j=1}^{|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|} \text{Prob}[\,\pi^*(g) = t \,\wedge\, H = H_{\mathcal{P}}^{[i]} \,\wedge\, \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j} \,|\, q^{[1]}, \ldots, \mathcal{N}(i)] \tag{101}$$

Next, we apply Corollary 8 and the fact that H, π^* are independently drawn:

$$\begin{aligned} &\sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \sum_{j=1}^{|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|} \text{Prob}[\,\pi^*(g) = t \,\wedge\, H = H_{\mathcal{P}}^{[i]} \,\wedge\, \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j} \,|\, q^{[1]}, \ldots, \mathcal{N}(i)] \\ &= \sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \sum_{j=1}^{|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|} \text{Prob}[\pi^*(g) = t \,|\, H = H_{\mathcal{P}}^{[i]} \,\wedge\, \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j}] \cdot \\ &\qquad \text{Prob}[\,H = H_{\mathcal{P}}^{[i]} \;|\; H \in \mathcal{H}_{\mathcal{P}}^{[i]}\,] \cdot \text{Prob}[\,\pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j} \,|\, \pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}\,] \end{aligned} \tag{102}$$

The right side of equality (102) can be bounded by a simpler expression that involves the maximum of $\mathrm{Prob}[\pi^*(g) = t \mid H = H_{\mathcal{P}}^{[i]} \wedge \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j}]$, computed over all $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$ and $\mathcal{F}_{H_{\mathcal{P}}^{[i]},j} \subseteq \mathcal{C}_{H_{\mathcal{P}}^{[i]}}$:

$$\sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \sum_{j=1}^{|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|} \mathrm{Prob}[\,\pi^*(g) = t \wedge H = H_{\mathcal{P}}^{[i]} \wedge \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j} \mid q^{[1]}, \ldots, \mathcal{N}(i)]$$

$$\leq \sum_{H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}} \sum_{j=1}^{|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|} \big(\max_{H_{\mathcal{P}}^{[i]},j} \mathrm{Prob}[\pi^*(g) = t \mid H = H_{\mathcal{P}}^{[i]} \wedge \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j}] \big) \cdot$$
$$\mathrm{Prob}[\,H = H_{\mathcal{P}}^{[i]} \mid H \in \mathcal{H}_{\mathcal{P}}^{[i]}\,] \cdot \mathrm{Prob}[\,\pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j} \mid \pi^*() \in \mathcal{C}_{H_{\mathcal{P}}^{[i]}}\,]$$

$$\leq \max_{H_{\mathcal{P}}^{[i]},j} \mathrm{Prob}[\pi^*(g) = t \mid H = H_{\mathcal{P}}^{[i]} \wedge \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j}] \tag{103}$$

Since t is neither a tag returned on any sign query of the set $Q_s^{[i]}$, nor a tag included in any verification query of the set $Q_s^{[i]}$, there is no mapping in any set $f_{H_{\mathcal{P}}^{[i]},j}$, $j \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ that has t as its destination. Therefore, from among the $(2^N - |f_{H_{\mathcal{P}}^{[i]},j}|)!$ permutations of any set $\mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$, the number of permutations that support both the mappings of $f_{H_{\mathcal{P}}^{[i]},j}$ and the mapping $g \to t$ is $(2^N - |f_{H_{\mathcal{P}}^{[i]},j}| - 1)!$ Hence:

$$\begin{aligned} &\max_{H_{\mathcal{P}}^{[i]},j} \mathrm{Prob}[\pi^*(g) = t \mid H = H_{\mathcal{P}}^{[i]} \wedge \pi^*() \in \mathcal{F}_{H_{\mathcal{P}}^{[i]},j}] \\ &= \max_{H_{\mathcal{P}}^{[i]},j} \frac{(2^N - |f_{H_{\mathcal{P}}^{[i]},j}| - 1)!}{(2^N - |f_{H_{\mathcal{P}}^{[i]},j}|)!} = \max_{H_{\mathcal{P}}^{[i]},j} \frac{1}{2^N - |f_{H_{\mathcal{P}}^{[i]},j}|} \\ &\leq \frac{1}{2^N - |Q_s^{[i]}| - 2|Q_v^{[i]}|} \leq \frac{1}{2^N - 2|Q|} \end{aligned} \tag{104}$$

The correctness of Lemma 9 follows from relations (102), (103) and (104).

□

A next lemma establishes that the decrypted tag value ${\pi^*}^{-1}(t)$ is associated with probability that is also uniformly bounded, in the case that the tag t is included in at least one verification query.

Lemma 10: *On the probability distribution of a decrypted tag value ${\pi^*}^{-1}(t)$, where t that has been queried before as part of a verification query, given that no security critical events occur up to $q^{[i]}$.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$

be sign and verification oracles of a MAGIC mode and $\mathcal{A}^*$ an adversary attacking these oracles, as in Lemma 5. Let $i \in |Q_s| + |Q_v|$ be a query index such that no security critical events occur up to query $q^{[i]}$, and knowledge about queries $q^{[1]}, \ldots, q^{[i]}$ and their associated responses $\mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]})$ is available. Let $Q_s^{[i]}$ and $Q_v^{[i]}$ be the sets of sign and verification queries issued by $\mathcal{A}^*$ up to query $q^{[i]}$. Let also t be a response from the sign oracle $\mathsf{MAC}_H^{\pi^*()}()$, such that there is no sign query $q_s^{(j)} \in Q_s^{[i]}$ for which $\mathcal{O}(q_s^{(j)}) = t$, and there is at least one verification query $q_v^{(j)} = (D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)}, T^{(j)}) \in Q_v^{[i]}$ for which $T^{(j)} = t$. Then, the probability that the decrypted tag value ${\pi^*}^{-1}(t)$ is equal to some value g is uniformly bounded across all $g \in \{0,1\}^N$:

$$\begin{aligned}&\text{Prob}[\, {\pi^*}^{-1}(t) = g \mid \ q^{[1]}, \ldots, q^{[i]},\, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}),\, \mathcal{N}(1), \ldots, \mathcal{N}(i)] \\ &\quad \le \frac{1}{2^N - (\eta+1)|Q|} \cdot \Big(1 + \frac{4 \cdot |Q|}{2^N - (\eta+5)|Q|}\Big)^{2|Q|} \ \forall\, g \in \{0,1\}^N\end{aligned} \tag{105}$$

where $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$.

Proof of Lemma 10: We assume that $H = H_{\mathcal{P}}^{[i]}$, for some $H_{\mathcal{P}}^{[i]} \in \mathcal{H}_{\mathcal{P}}^{[i]}$. For this hash key value, we consider procedure $\mathsf{SelectPermutations}()$, which computes the set of blinding permutations $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ that satisfy the constraints of the hypothesis. Since t is a tag included in a verification query, and since there is no sign query the response of which is t, then, the verification query which includes t is either a query, the tag of which is correctable, or a reject query. In both cases, there exists exactly one non-empty set of mappings m_{i_t} of index $i_t \in [1, n_m]$ from the set $\{m_1, m_2, \ldots, m_{n_m}\}$ passed as input to $\mathsf{CombineMappings}()$, which includes mappings the destination of which is t. Let's use the notation $q_v^{(j)}$ to refer to the verification query, the tag of which is t. The mappings of the set m_{i_t} originate from elements of the set $\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} - \mathcal{G}_{H_{\mathcal{P}}^{[i]}}$ returned from procedure $\mathsf{ObtainVMappings}()$. The set $\mathcal{G}_{H_{\mathcal{P}}^{[i]}}$ is associated with the sign queries and is computed in line 1 of $\mathsf{ObtainVMappings}()$. The set $\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$ is associated with the verification query $q_v^{(j)}$ and is computed in line 2 of $\mathsf{ObtainVMappings}()$.

The probability of the event ${\pi^*}^{-1}(t) = g$ and its equivalent $p^*(g) = t$ depends on whether $g \in \mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} - \mathcal{G}_{H_{\mathcal{P}}^{[i]}}$. If $g \notin \mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} - \mathcal{G}_{H_{\mathcal{P}}^{[i]}}$, then the probability that ${\pi^*}^{-1}(t) = g$ or $p^*(g) = t$ is zero, since the mapping $g \to t$ is not in m_{i_t}, and is thus not supported by any permutation of the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$. If, however, $g \in \mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)} - \mathcal{G}_{H_{\mathcal{P}}^{[i]}}$, then the probability that ${\pi^*}^{-1}(t) = g$ or $p^*(g) = t$ is non-zero. For this probability we can compute a bound.

On input $\{m_1, m_2, \ldots, m_{n_m}\}$, procedure CombineMappings() returns $|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|$ sets of mappings, where each returned set, after merging with $\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$, forms a set $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$. Such set has cardinality $|f_{H_{\mathcal{P}}^{[i]},I}| = |Q_s^{[i]}| + n_m \leq |Q_s^{[i]}| + 2|Q_v^{[i]}|$. The cardinality $|\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|$ and the sets $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ are defined in the proof of Lemma 9. $Q_s^{[i]}$ and $Q_v^{[i]}$ are the sign and verification queries issued by adversary $\mathcal{A}^*$ up to query $q^{[i]}$. Each set $f_{H_{\mathcal{P}}^{[i]},I}$ if passed into procedure MappingsToPermutations() produces $(2^N - |Q_s^{[i]}| - n_m)!$ permutations, all of which support the mappings of $f_{H_{\mathcal{P}}^{[i]},I}$.

The probability of the events $\pi^{*^{-1}}(t) = g$ and $p^*(g) = t$ is determined by the ratio of the number of permutations returned by prodecure MappingsToPermutations() that support the mapping $g \to t$, over the total number of permutations returned by procedure MappingsToPermutations(). This is also the ratio of the number of sets $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ that include the mapping $g \to t$, returned by procedure CombineMappings(), over the total number of sets returned by procedure CombineMappings().

To prove Lemma 10, we compute an upper bound for the number of sets $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ that include the mapping $g \to t$, and a lower bound for the total number of sets returned by procedure CombineMappings(). From the definition of CombineMappings(), it follows that the number of sets $f_{H_{\mathcal{P}}^{[i]},I}$ that include the mapping $g \to t$ is at most $|m_1| \cdot \ldots \cdot |m_{i_t-1}| \cdot |m_{i_t+1}| \cdot \ldots \cdot |m_{n_m}|$. This is because such sets are formed by selecting one mapping from each set in $\{m_1, m_2, \ldots, m_{n_m}\}$, in every possible way, such that the two conditions of CombineMappings() are satisfied, and the mapping selected from m_{i_t} is always $g \to t$.

A lower bound for the total number of sets returned by procedure CombineMappings() is obtained as follows: Let's assume that a mapping $u_1 \to v_1$ is selected from set m_1. In each of the sets $m_2, \ldots, m_{n_m}$ there may be at most one mapping with origin u_1 and at most one mapping with destination v_1. Such mappings cannot be in the same set $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ as mapping $u_1 \to v_1$, according to the second condition of CombineMappings(). Therefore, the number of mappings from $m_2, \ldots, m_{n_m}$, which can form sets $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ together with the mapping $u_1 \to v_1$ of m_1, is at least $|m_2| - 2, \ldots, |m_{n_m}| - 2$, respectively. Similarly, for each of the $|m_2| - 2$ mappings of m_2 and there are at least $|m_3| - 4, \ldots, |m_{n_m}| - 4$ mappings from $m_3, \ldots, m_{n_m}$, which together with the mapping from m_2 and $u_1 \to v_1$ form sets $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$. Using the same reasoning for sets $m_3, \ldots, m_{n_m}$, it follows that the number of sets $f_{H_{\mathcal{P}}^{[i]},I}$, $I \in [1, |\mathcal{P}_{H_{\mathcal{P}}^{[i]}}|]$ which include the mapping $u_1 \to v_1$ is at least $(|m_2| - 2) \cdot \ldots \cdot (|m_{n_m}| - 2n_m + 2)$. Since the bound holds for every mapping of m_1, the total number of sets

$f_{H^{[i]}_{\mathcal{P}},I}$, $I \in [1, |\mathcal{P}_{H^{[i]}_{\mathcal{P}}}|]$ returned by procedure CombineMappings() is at least $|m_1| \cdot (|m_2| - 2) \cdot \ldots \cdot (|m_{n_m}| - 2 \cdot n_m + 2)$, which, in turn, is greater than $|m_1| \cdot (|m_2| - 2 \cdot n_m) \cdot \ldots \cdot (|m_{n_m}| - 2 \cdot n_m)$. Furthermore, such lower bound can be computed using a different order for $m_1, m_2, \ldots, m_{n_m}$. Therefore, an alternative bound for the total number of sets returned by procedure CombineMappings(), which we can also use in the proof, is $(|m_1| - 2 \cdot n_m) \cdot \ldots \cdot (|m_{i_t - 1}| - 2 \cdot n_m) \cdot |m_{i_t}| \cdot (|m_{i_t+1}| - 2 \cdot n_m) \cdot \ldots \cdot (|m_{n_m}| - 2 \cdot n_m)$.

We proceed with the proof considering that the event $\pi^*(g) = t$ happens in conjunction with all mutually exclusive events $H = H^{[i]}_{\mathcal{P}}$, $H^{[i]}_{\mathcal{P}} \in \mathcal{H}^{[i]}_{\mathcal{P}}$. We also make use of Corollary 8:

$$\begin{aligned} &\text{Prob}[\, \pi^*(g) = t \mid q^{[1]}, \ldots, q^{[i]}, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}), \mathcal{N}(1), \ldots, \mathcal{N}(i)] \\ &= \sum_{H^{[i]}_{\mathcal{P}} \in \mathcal{H}^{[i]}_{\mathcal{P}}} \text{Prob}[\pi^*(g) = t \mid H = H^{[i]}_{\mathcal{P}} \wedge \pi^*() \in \mathcal{C}_{H^{[i]}_{\mathcal{P}}}\,] \cdot \\ &\qquad\qquad \text{Prob}[\, H = H^{[i]}_{\mathcal{P}} \mid H \in \mathcal{H}^{[i]}_{\mathcal{P}}\,] \\ &\le \max_{H^{[i]}_{\mathcal{P}}} \text{Prob}[\pi^*(g) = t \mid H = H^{[i]}_{\mathcal{P}} \wedge \pi^*() \in \mathcal{C}_{H^{[i]}_{\mathcal{P}}}\,] \end{aligned} \tag{106}$$

Inequality (106) can lead to the desired bound by introducing the cardinalities of the sets $f_{H^{[i]}_{\mathcal{P}},I}$, $I \in [1, |\mathcal{P}_{H^{[i]}_{\mathcal{P}}}|]$, which are returned by procedure CombineMappings() when the permutations of $\mathcal{C}_{H^{[i]}_{\mathcal{P}}}$ are computed:

$$\begin{aligned} &\max_{H^{[i]}_{\mathcal{P}}} \text{Prob}[\pi^*(g) = t \mid H = H^{[i]}_{\mathcal{P}} \wedge \pi^*() \in \mathcal{C}_{H^{[i]}_{\mathcal{P}}}\,] \\ &= \max_{H^{[i]}_{\mathcal{P}}} \frac{\sum_{I \in [1, |\mathcal{P}_{H^{[i]}_{\mathcal{P}}}|],\, g \to t \,\in\, f_{H^{[i]}_{\mathcal{P}},I}} |f_{H^{[i]}_{\mathcal{P}},I}|}{\sum_{I \in [1, |\mathcal{P}_{H^{[i]}_{\mathcal{P}}}|]} |f_{H^{[i]}_{\mathcal{P}},I}|} \end{aligned} \tag{107}$$

To further bound the right side of equation (107) we introduce the bounds for the enumerator and denominator discussed above:

$$\begin{aligned} &\max_{H^{[i]}_{\mathcal{P}}} \frac{\sum_{I \in [1, |\mathcal{P}_{H^{[i]}_{\mathcal{P}}}|],\, g \to t \,\in\, f_{H^{[i]}_{\mathcal{P}},I}} |f_{H^{[i]}_{\mathcal{P}},I}|}{\sum_{I \in [1, |\mathcal{P}_{H^{[i]}_{\mathcal{P}}}|]} |f_{H^{[i]}_{\mathcal{P}},I}|} \\ &\le \max_{H^{[i]}_{\mathcal{P}}} \frac{|m_1| \cdot \ldots \cdot |m_{i_t - 1}| \cdot |m_{i_t+1}| \cdot \ldots \cdot |m_{n_m}|}{(|m_1| - 2 \cdot n_m) \cdot \ldots \cdot |m_{i_t}| \cdot \ldots \cdot (|m_{n_m}| - 2 \cdot n_m)} \\ &= \max_{H^{[i]}_{\mathcal{P}}} \frac{1}{|m_{i_t}|} \cdot \prod_{\substack{J \in [1, n_m] \\ J \neq i_t}} \Big(1 + \frac{2 \cdot n_m}{|m_J| - 2 \cdot n_m}\Big) \end{aligned} \tag{108}$$

where $m_1, m_2, \ldots, m_{n_m}$ are the sets passed as input to procedure $\mathsf{CombineMappings}()$, when the permutations of $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ are computed, and m_{i_t} is the set that contains mappings to the tag value t. As each of the sets $m_1, m_2, \ldots, m_{n_m}$ is returned either by procedure $\mathsf{ObtainVMappings}()$, or procedure $\mathsf{ObtainUMappings}()$, and relations (81) and (86) hold, each of the cardinalities $|m_J|$, $J \in [1, n_m]$ satisfies $|m_J| \geq 2^N - (\eta + 1)|Q|$, where $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$. Taking into account that $n_m \leq 2|Q|$, we complete the proof of Lemma 10:

$$\begin{aligned} &\max_{H_{\mathcal{P}}^{[i]}} \frac{1}{|m_{i_t}|} \cdot \prod_{\substack{J \in [1, n_m] \\ J \neq i_t}} \Big(1 + \frac{2 \cdot n_m}{|m_J| - 2 \cdot n_m}\Big) \\ &\leq \frac{1}{2^N - (\eta+1)|Q|} \cdot \Big(1 + \frac{4 \cdot |Q|}{2^N - (\eta+5)|Q|}\Big)^{2|Q|} \end{aligned} \tag{109}$$

Lemma 10 follows directly from relations (106), (107), (108) and (109).

$\square$

Corollary 9: If $|Q| \leq \frac{2^{N/2}}{\sqrt{8}}$ and $\frac{2^N}{|Q|} \gg \eta+5$ the probability bound of Lemma 10 becomes:

$$\begin{aligned} &\text{Prob}[\, {\pi^*}^{-1}(t) = g \mid \; q^{[1]}, \ldots, q^{[i]}, \, \mathcal{O}(q^{[1]}), \ldots, \mathcal{O}(q^{[i]}), \, \mathcal{N}(1), \ldots, \mathcal{N}(i)] \\ &\leq \frac{\mathsf{e}}{2^N - (\eta+1)|Q|} + \frac{1}{2^N - (\eta+1)|Q|} \cdot O\Big(\frac{4 \cdot |Q|}{2^N - (\eta+5)|Q|}\Big) \\ &= \frac{\mathsf{e}}{2^N - (\eta+1)|Q|} + O(\frac{1}{2^{\frac{3N}{2}}}), \quad \forall \, g \in \{0,1\}^N \end{aligned} \tag{110}$$

where e is the base of the natural logarithm. Corollary 9 follows from the identity $(1 + \frac{1}{x})^x = e + O(\frac{1}{x})$, which is applied to the bound of Lemma 10 for $x \leftarrow \frac{2^N - (\eta+5)|Q|}{4|Q|}$.

3.9 Main results

The theorems of this section provide bounds for the probabilities of first occurrence of the four types of security critical events of the MAGIC mode, and for the advantage of adversary $\mathcal{A}^*$.

Theorem 3: *On the probability of first occurrence of a MAC collision event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode for which blinding is performed by a random permutation π^*. Let

$\mathcal{A}^*$ be a polynomial time algorithm playing a game the success of which is defined by relation (44), and where $\mathsf{E}_{K_B}() \leftarrow \pi^*()$. Then, the probability $F_{\mathcal{C}}(i)$ of the first occurrence of a MAC collision event $\mathcal{E}_{\mathcal{C}}(i)$, associated with query $q^{[i]} \in Q$ of index $i \in [1, |Q|]$ is bounded by:

$$F_{\mathcal{C}}(i) \; \leq \; \frac{n \cdot |Q|}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \tag{111}$$

where $\zeta = \frac{n(n-1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|^2$, $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$.

Proof of Theorem 3: From the definition of the MAC collision event, it holds that:

$$F_{\mathcal{C}}(i) = \sum_{j=1}^{|Q_s^{[i-1]}|} \text{Prob}[\mathsf{MAC}_H^{\pi^*()}(q_s^{(j)}) = \mathsf{MAC}_H^{\pi^*()}(q^{[i]}) \,|\, \mathcal{N}(1), \ldots, \mathcal{N}(i-1)] \tag{112}$$

where $Q_s^{[i-1]}$ is the set of sign queries issued by adversary $\mathcal{A}^*$ up to query $q^{[i-1]}$. We proceed by denoting any query $q_s^{(j)} \in Q_s^{[i-1]}$ as $(D^{(j)}, C_1^{(j)}, \ldots, C_n^{(j)})$ and any query $q^{[i]} \in Q$ as $(D^{[i]}, C_1^{[i]}, \ldots, C_n^{[i]})$. Using such notation, equation (112) can be written as:

$$\begin{aligned} F_{\mathcal{C}}(i) = \sum_{j=1}^{|Q_s^{[i-1]}|} & \text{Prob}[\, \pi^*(D^{(j)} + C_1^{(j)}H + \ldots + C_n^{(j)}H^n) = \\ & \pi^*(D^{[i]} + C_1^{[i]}H + \ldots + C_n^{[i]}H^n) \mid \mathcal{N}(1), \ldots, \mathcal{N}(i-1) \,] \end{aligned} \tag{113}$$

As in the proof of Lemma 7, we apply operator $\pi^{*^{-1}}()$ to both sides of the internal equation of (113) to obtain:

$$\begin{aligned} F_{\mathcal{C}}(i) = \sum_{j=1}^{|Q_s^{[i-1]}|} & = \text{Prob}[\; (D^{(j)} + D^{[i]}) + (C_1^{(j)} + C_1^{[i]})H + \ldots + \\ & (C_n^{(j)} + C_1^{[i]})H^n = 0 \mid \; \mathcal{N}(1), \ldots, \mathcal{N}(i-1) \,] \end{aligned} \tag{114}$$

The internal polynomial equation of (114) associated with any pair of queries $(q_s^{(j)}, q^i)$ has $r_{i,j}$ distinct roots, where $r_{i,j} \in [0, n]$. We refer to such roots as $H_1^{(i,j)}, \ldots, H_{r_{i,j}}^{(i,j)}$. Then equation (114) becomes:

$$F_{\mathcal{C}}(i) = \sum_{j=1}^{|Q_s^{[i-1]}|} \text{Prob}[H = H_1^{(i,j)} \vee \ldots \vee H = H_{r_{i,j}}^{(i,j)} \mid \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \tag{115}$$

We complete the proof of Theorem 3 applying Lemma 7 to equation (115):

$$\begin{aligned} F_{\mathcal{C}}(i) \ &\leq\ \sum_{j=1}^{|Q_s^{[i-1]}|} \frac{r_{i,j}}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \\ &\leq\ \frac{n \cdot |Q|}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \end{aligned} \tag{116}$$

□

Theorem 4: *On the probability of first occurrence of a Galois Hash computation event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode for which blinding is performed by a random permutation π^*. Let $\mathcal{A}^*$ be a polynomial time algorithm playing a game the success of which is defined by relation (44), and where $\mathsf{E}_{K_B}() \leftarrow \pi^*()$. Then, the probability $F_{\mathcal{G}}(i)$ of the first occurrence of a Galois Hash computation event $\mathcal{E}_{\mathcal{G}}(i)$, associated with query $q^{[i]} \in Q$ of index $i \in [1, |Q|]$ is bounded by:

$$F_{\mathcal{G}}(i)\ \leq\ \frac{n}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \tag{117}$$

where $\zeta = \frac{n(n-1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|^2$, $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$.

Proof of Theorem 4: We refer to query $q^{[J]} \in Q_s$, for which a Galois Hash output is computed, as $(D^{[J]}, C_1^{[J]}, \ldots, C_n^{[J]})$. Query $q^{[J]}$ includes at least one non-zero ciphertext block. From the event definition, it holds that:

$$\begin{aligned} F_{\mathcal{G}}(i) = \text{Prob}[\, & q^{[J]} \in Q_s;\ G_J \ \leftarrow\ \mathcal{A}^{*[i]};\ D^{[J]} + C_1^{[J]} \cdot H + \ldots + \\ & C_n^{[J]} \cdot H^n = G_J \mid \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \\ \leq \text{Prob}[\, & G_J \ \leftarrow\ \mathcal{A}^{*[i]};\ D^{[J]} + C_1^{[J]} \cdot H + \ldots + C_n^{[J]} \cdot H^n = G_J \\ & \mid q^{[J]} \in Q_s,\ \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \\ \leq \text{Prob}[\, & D^{[J]} + G_J + C_1^{[J]} \cdot H + \ldots + C_n^{[J]} \cdot H^n = 0 \\ & \mid q^{[J]} \in Q_s,\ \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \end{aligned} \tag{118}$$

As coefficients $C_1^{[J]}, \ldots, C_n^{[J]}$ are not all simultaneously zero, the internal polynomial equation of (118) has $r_{J,G_J} \in [0, n]$ distinct roots, which we refer to as $H_1^{[J]}, \ldots, H_{r_{J,G_J}}^{[J]}$. By introducing these roots into inequality (118) and by using Lemma 7, the inequality becomes:

$$\begin{aligned} F_{\mathcal{G}}(i) \;\leq\; & \text{Prob}[H = H_1^{[J]} \,\vee\, \ldots \,\vee\, H = H_{r_{J,G_J}}^{[J]} \\ & \quad \mid q^{[J]} \in Q_s,\, \mathcal{N}(1), \ldots, \mathcal{N}(i)\,] \\ \leq\; & \max_{q^{[J]},\, G_J} \frac{r_{J,G_J}}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \qquad (119) \\ \leq\; & \frac{n}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \end{aligned}$$

This completes the proof. □

Theorem 5: *On the probability of first occurrence of an ECC function exploitation event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode for which blinding is performed by a random permutation π^*. Let $\mathcal{A}^*$ be a polynomial time algorithm playing a game the success of which is defined by relation (44), and where $\mathsf{E}_{K_B}() \leftarrow \pi^*()$. Then, the probability $F_{\mathcal{X}}(i)$ of the first occurrence of an ECC function exploitation event $\mathcal{E}_{\mathcal{X}}(i)$, associated with query $q^{[i]} \in Q$ of index $i \in [1, |Q|]$ is bounded by:

$$F_{\mathcal{X}}(i) \;\leq\; \max\Big(\frac{n \cdot Z \cdot |\mathcal{E}_{\mathcal{T}}|}{2^N - (\eta+1)|Q|},\; \frac{n^2 \cdot |\mathcal{E}_{\mathcal{T}}|}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}}\Big) \qquad (120)$$

where $Z = (1 + \frac{4 \cdot |Q|}{2^N - (\eta+5)|Q|})^{2|Q|}$, $\zeta = \frac{n(n-1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|^2$, $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$. Furthermore, when $2 \leq |Q| \leq \frac{2^{N/2}}{\sqrt{8}}$, $\frac{2^N}{|Q|} \gg \eta + 5$ and $n \geq 3$ the bound for the probability of first occurrence $F_{\mathcal{X}}(i)$ becomes:

$$F_{\mathcal{X}}(i) \;\leq\; \frac{n^2 \cdot |\mathcal{E}_{\mathcal{T}}|}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \qquad (121)$$

Proof of Theorem 5: We refer to query $q^{[i]} \in Q$, which is a verification query, as $(D^{[i]}, C_1^{[i]}, \ldots, C_n^{[i]}, T^{[i]})$. From the definition of the ECC function exploitation event, it holds that:

$$\begin{aligned} F_{\mathcal{X}}(i) = \mathrm{Prob}[\, & q^{[i]} \leftarrow \mathcal{A}^{*[i-1]};\; q^{[i]} \in Q_v;\; \mathsf{Cond}_5 \,\vee\, \mathsf{Cond}_6; \\ & \mathsf{Verify}_H^{\pi^*()}(q^{[i]}) = (\mathsf{accept},\, \mathfrak{s}_1^{[i]},\, \mathfrak{s}_2^{[i]});\;\; \mathfrak{s}_1^{[i]} \neq \perp,\, \mathfrak{s}_2^{[i]} \neq \perp \\ & |\, \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \end{aligned} \tag{122}$$

where Cond_5 and Cond_6 refer to the conditions of (42) and (43) evaluated on $(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) \leftarrow q^{[i]}$, and for which $\mathsf{E}_{K_B}() \leftarrow \pi^*()$. We proceed with the proof, removing some of the conditions in $\mathcal{E}_{\mathcal{X}}(i)$ in order to bound $F_{\mathcal{X}}(i)$:

$$\begin{aligned} F_{\mathcal{X}}(i) \;\leq\; \mathrm{Prob}[\, & \mathsf{Verify}_H^{\pi^*()}(q^{[i]}) = (\mathsf{accept},\, \mathfrak{s}_1^{[i]},\, \mathfrak{s}_2^{[i]});\;\; \mathfrak{s}_1^{[i]} \neq \perp,\, \mathfrak{s}_2^{[i]} \neq \perp \\ & |\, q^{[i]} \in Q_v,\; \mathsf{Cond}_5 \,\vee\, \mathsf{Cond}_6,\, \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \end{aligned} \tag{123}$$

From the flow of procedure $\mathsf{Verify}_H^{\pi^*()}()$, it holds that:

$$\begin{aligned} F_{\mathcal{X}}(i) \;\leq\; & \sum_{k=1}^{n} \sum_{e_1 \in \mathcal{E}_{\mathcal{T}}} \mathrm{Prob}[\, (D^{[i]} + C_1^{[i]} \cdot H + \ldots + C_n^{[i]} \cdot H^n \\ & + \pi^{*^{-1}}(T^{[i]})) \cdot H^{-k} = e_1 \\ & |\, q^{[i]} \in Q_v,\; \mathsf{Cond}_5 \,\vee\, \mathsf{Cond}_6,\, \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \\ \leq\; & \sum_{k=1}^{n} \sum_{e_1 \in \mathcal{E}_{\mathcal{T}}} \mathrm{Prob}[\, \pi^{*^{-1}}(T^{[i]}) = D^{[i]} + C_1^{[i]} \cdot H \\ & + \ldots + C_n^{[i]} \cdot H^n + e_1 \cdot H^k \\ & |\, q^{[i]} \in Q_v,\; \mathsf{Cond}_5 \,\vee\, \mathsf{Cond}_6,\, \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \end{aligned} \tag{124}$$

If $T^{[i]}$ is not a tag returned by any sign query, the probability that $\pi^{*^{-1}}(T^{[i]})$ takes some value in $\{0,1\}^N$, is bounded according to either Lemma 9 or Lemma 10, given a set of adversary queries $q^{[1]}, \ldots, q^{[i-1]}$, and the fact that no security critical event occurs up to query $q^{[i-1]}$. The bound depends on whether $T^{[i]}$ is a tag included in a verification query issued before $q^{[-1]}$, or whether $T^{[i]}$ is not included in any verification query. Since the bound of Lemma 9 is always tighter than the bound of Lemma 10, we can use the bound of Lemma 10 to simplify (124) in this case. If there is a sign query, the response of which is $T^{[i]}$, the probability that $\pi^{*^{-1}}(T^{[i]})$ takes some value in $\{0,1\}^N$ is the probability of the first occurrence of a Galois Hash computation event. Such probability is bounded by (117). Therefore we can use the maximum of the bounds of (105) and (117) to simplify inequality (124):

$$\begin{aligned} F_{\mathcal{X}}(i) &\leq \sum_{k=1}^{n} \sum_{e_1 \in \mathcal{E}_{\mathcal{T}}} \max\Big(\frac{Z}{2^N-(\eta+1)|Q|}, \frac{n}{2^N-\zeta-\eta|Q|-n\binom{|Q|}{2}}\Big) \\ &\leq \max\Big(\frac{n \cdot Z \cdot |\mathcal{E}_{\mathcal{T}}|}{2^N-(\eta+1)|Q|}, \frac{n^2 \cdot |\mathcal{E}_{\mathcal{T}}|}{2^N-\zeta-\eta|Q|-n\binom{|Q|}{2}}\Big) \end{aligned} \tag{125}$$

where to simplify the expression for the bound of (105), we substituted $(1+\frac{4\cdot|Q|}{2^N-(\eta+5)|Q|})^{2|Q|}$ with Z. The second part of Theorem 5 directly follows from combining (125) with Corollary 9.

□

Theorem 6: *On the probability of first occurrence of a blind forgery event.* Let $\mathsf{MAC}_H^{\pi^*()}()$ and $\mathsf{Verify}_H^{\pi^*()}()$ be sign and verification oracles of a MAGIC mode for which blinding is performed by a random permutation π^*. Let $\mathcal{A}^*$ be a polynomial time algorithm playing a game the success of which is defined by relation (44), and where $\mathsf{E}_{K_B}() \leftarrow \pi^*()$. Then, the probability $F_{\mathcal{B}}(i)$ of the first occurrence of a blind forgery event $\mathcal{E}_{\mathcal{B}}(i)$, associated with query $q^{[i]} \in Q$ of index $i \in [1, |Q|]$ is bounded by the bound of Lemma 10:

$$F_{\mathcal{B}}(i) \leq \frac{1}{2^N-(\eta+1)|Q|} \cdot \Big(1+\frac{4\cdot|Q|}{2^N-(\eta+5)|Q|}\Big)^{2|Q|} \tag{126}$$

Furthermore, if $|Q| \leq \frac{2^{N/2}}{\sqrt{8}}$ and $\frac{2^N}{|Q|} \gg \eta+5$, the probability $F_{\mathcal{B}}(i)$ is bounded by the bound of Corollary 9:

$$F_{\mathcal{B}}(i) \leq \frac{\mathsf{e}}{2^N-(\eta+1)|Q|} + O(\frac{1}{2^{\frac{3N}{2}}}) \tag{127}$$

where e is the base of the natural logarithm and the term η is defined as in Theorem 5.

Proof of Theorem 6: We refer to verification query $q^{[i]} \in Q$ as $(D^{[i]}, C_1^{[i]}, \ldots, C_n^{[i]}, T^{[i]})$. Theorem 6 follows from the definition of the blind forgery event:

$$
\begin{aligned}
F_{\mathcal{B}}(i) &= \mathrm{Prob}[\, q^{[i]} \leftarrow \mathcal{A}^{*[i-1]};\ q^{[i]} \in Q_v;\ \mathsf{Cond}_5 \ \vee\ \mathsf{Cond}_6; \\
&\qquad\qquad \mathsf{Verify}_H^{\pi^*()}(q^{[i]}) = (\mathsf{accept}, \perp, \perp) \mid \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \\
&\le \mathrm{Prob}[\, \mathsf{Verify}_H^{\pi^*()}(q^{[i]}) = (\mathsf{accept}, \perp, \perp) \\
&\qquad\qquad \mid q^{[i]} \in Q_v,\ \mathsf{Cond}_5 \ \vee\ \mathsf{Cond}_6,\ \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,] \\
&= \mathrm{Prob}[\, {\pi^*}^{-1}(T^{[i]}) = D^{[i]} + C_1^{[i]} \cdot H + \ldots + C_n^{[i]} \cdot H^n \\
&\qquad\qquad \mid q^{[i]} \in Q_v,\ \mathsf{Cond}_5 \ \vee\ \mathsf{Cond}_6,\ \mathcal{N}(1), \ldots, \mathcal{N}(i-1)\,]
\end{aligned}
\tag{128}
$$

where Cond_5 and Cond_6 refer to the conditions of (42) and (43) evaluated on $(D^{(r)}, C_1^{(r)}, \ldots, C_n^{(r)}, T^{(r)}) \leftarrow q^{[i]}$, and for which $\mathsf{E}_{K_B}() \leftarrow \pi^*()$.

Since query $q^{[i]}$ satisfies at least one of these two conditions, $T^{[i]}$ cannot be a tag returned by a sign query issued before $q^{[i]}$. Hence, the probability of the event ${\pi^*}^{-1}(T^{[i]}) = D^{[i]} + C_1^{[i]} \cdot H + \ldots + C_n^{[i]}$, given the conditions of (128), is bounded by one of the bounds of Lemma 9 or Lemma 10. The first part of Theorem 6 follows from the observation that, between the two bounds, the bound of Lemma 9 is always tighter, so the bound of Lemma 10 is applicable in all cases of the conditions of (128). The second part of Theorem 6 directly follows from combining (128) with Corollary 9.

□

A next theorem establishes the security of the MAGIC mode against an adversary who plays the MAC forgery game, the success of which is defined by relation (44). This is the main result from our analysis.

Theorem 7: *On the security of the MAGIC mode against adversary $\mathcal{A}$.* Let $\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}()$ and $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}()$ be sign and verification oracles of a MAGIC mode. Let $\mathcal{A}$ be a polynomial time algorithm playing a game the success of which is defined by relation (44), and which is characterized by the query budget constraint of (46). Then, the advantage of adversary $\mathcal{A}$ is bounded by:

$$
\begin{aligned}
\mathbf{Adv}^{\mathcal{A}} \ \le\ & \frac{n \cdot (|Q|^2 + |Q|)}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}} \\
& + \max\Big(\frac{n \cdot Z \cdot |\mathcal{E}_{\mathcal{T}}| \cdot |Q|}{2^N - (\eta+1)|Q|}, \frac{n^2 \cdot |\mathcal{E}_{\mathcal{T}}| \cdot |Q|}{2^N - \zeta - \eta|Q| - n\binom{|Q|}{2}}\Big) \\
& + \frac{Z \cdot (|Q|+1)}{2^N - (\eta+1)|Q|} \ +\ \mathbf{Adv}^{\mathsf{E}_{K_B}()}
\end{aligned}
\tag{129}
$$

Table 1: $\mathbf{Adv}^{\mathcal{A}}$, computed from inequality (130), when $N = 128$, $n = 4$ and $T_{th} = 10$

$\lvert Q\rvert$	$\mathbf{Adv}^{\mathcal{A}}$	$\lvert Q\rvert$	$\mathbf{Adv}^{\mathcal{A}}$
2	2^{-75}	2^{32}	2^{-44}
8	2^{-73}	2^{40}	2^{-36}
16	2^{-72}	2^{48}	2^{-27}
64	2^{-70}	2^{56}	2^{-14}
256	2^{-68}	$2^{62.34}$	0.5

where $Z = (1 + \frac{4\cdot|Q|}{2^N-(\eta+5)|Q|})^{2|Q|}$, $\zeta = \frac{n(n-1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|^2$, $\eta = \frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$ and $|\mathcal{E}_{\mathcal{T}}| = \sum_{t=1}^{T_{th}} \binom{N}{t}$. Furthermore, when $2 \leq |Q| \leq \frac{2^{N/2}}{\sqrt{8}}$, $\frac{2^N}{|Q|} \gg \eta+5$ and $n \geq 3$ the bound for the advantage of adversary $\mathcal{A}$ becomes:

$$\begin{aligned}\mathbf{Adv}^{\mathcal{A}} \leq\ & \frac{n\cdot(|Q|^2+|Q|)+n^2\cdot|\mathcal{E}_{\mathcal{T}}|\cdot|Q|}{2^N-\zeta-\eta|Q|-n\binom{|Q|}{2}} + \frac{\mathsf{e}\cdot(|\mathsf{Q}|+1)}{2^N-(\eta+1)|Q|} \\ & + O\Big(\frac{|Q|}{2^{\frac{3N}{2}}}\Big) + \mathbf{Adv}^{\mathsf{E}_{K_B}()}\end{aligned} \tag{130}$$

where e is the base of the natural logarithm.

Proof of Theorem 7: Theorem 7 follows by directly combining the results from Lemma 5, Theorems 3, 4, 5 and 6, and relation (47). When combining Lemma 5 with the theorems, we take into account the fact that the probability term Prob$[\mathcal{W}^{\mathcal{A}^*}|\mathcal{N}^{\mathcal{A}^*}]$, which is present in Lemma 5, is the probability of the first occurrence of a blind forgery event. This is the probability of $\mathcal{A}^*$ winning, if no security critical events occur at the completion of all his queries. This probability term is also bounded according to Theorem 6.

□

3.10 Discussion

The recommended uses of the MAGIC mode are those for which the parameters n, N and T_{th} satisfy the conditions of inequality (130). If the advantage of adversary $\mathcal{A}$ is bounded according to inequality (130), then the MAGIC mode offers security in the order of $O(2^{N/2})$, with N being the tag size. Indeed, if the number of queries is close to the limit $\frac{2^{N/2}}{\sqrt{8}}$, and the conditions of (130) are satisfied, then $n \cdot |Q|^2$ becomes the dom-

Table 2: $\mathbf{Adv}^{\mathcal{A}}$, computed from inequality (130), when $N = 256$, $n = 8$ and $T_{th} = 20$

$\|Q\|$	$\mathbf{Adv}^{\mathcal{A}}$	$\|Q\|$	$\mathbf{Adv}^{\mathcal{A}}$
2	2^{-151}	2^{32}	2^{-120}
8	2^{-149}	2^{64}	2^{-88}
16	2^{-148}	2^{80}	2^{-72}
64	2^{-146}	2^{96}	2^{-56}
256	2^{-144}	$2^{125.84}$	0.5

Table 3: $\mathbf{Adv}^{\mathcal{A}}$, computed from inequality (130), when $N = 512$, $n = 16$ and $T_{th} = 40$

$\|Q\|$	$\mathbf{Adv}^{\mathcal{A}}$	$\|Q\|$	$\mathbf{Adv}^{\mathcal{A}}$
2	2^{-304}	2^{64}	2^{-241}
8	2^{-302}	2^{128}	2^{-177}
16	2^{-301}	2^{160}	2^{-145}
64	2^{-299}	2^{192}	2^{-113}
256	2^{-295}	$2^{253.34}$	0.5

inant term in the enumerator of $\frac{n\cdot(|Q|^2+|Q|)+n^2\cdot|\mathcal{E}_{\mathcal{T}}|\cdot|Q|}{2^N-\zeta-\eta|Q|-n\binom{|Q|}{2}}$. Similarly, $n \cdot \binom{|Q|}{2}$ becomes the dominant term which is subtracted from 2^N in the denominator of $\frac{n\cdot(|Q|^2+|Q|)+n^2\cdot|\mathcal{E}_{\mathcal{T}}|\cdot|Q|}{2^N-\zeta-\eta|Q|-n\binom{|Q|}{2}}$. This means that the number of queries required in order for the bound of the advantage of adversary $\mathcal{A}$ to become 0.5, is $O(2^{N/2})$. For the same number of queries the remaining terms of the bound of (130) are negligible.

Table 1 shows values for the bound of inequality (130), when $N = 128$, $n = 4$ and $T_{th} = 10$, as the number of queries $|Q|$ varies. The number of queries for which the advantage of adversary $\mathcal{A}$ is bounded by 0.5 is $2^{62.34}$, which is close to 2^{64} as expected. Furthermore, the advantage bound ranges between 2^{-75} and 2^{-68}, when the number of queries ranges between 2 and 256. Such tight query budgets characterize online attacks. In online attacks, attacked systems usually monitor oracle responses to verification queries. Increasing numbers of reject verification queries are perceived as indications of abnormal behavior, thus exposing the adversary and the attack.

Table 2 and 3 show values for the bound of inequality (130), when $N = 256$, $n = 8$ and $T_{th} = 20$, and when $N = 512$, $n = 16$ and $T_{th} = 40$. The number of queries for which the advantage of adversary $\mathcal{A}$ is bounded by 0.5 is $2^{125.84}$, and $2^{253.34}$ in the two tables, respectively. As in Table 1, these numbers are close to $2^{N/2}$ as expected. Furthermore, the advantage bound ranges between 2^{-151} and 2^{-144}, and between 2^{-151} and 2^{-144}, in the two tables, when the number of queries is small, ranging between 2 and 256. All numbers of queries in Tables 2 and 3, for which the advantage bound for $\mathcal{A}$ is computed, are lower than the query budget constraint of (46).

4 Implementation considerations and future work

The MAGIC mode as described in the paper can be efficiently implemented in hardware or software for several combinations of parameter values N, n and T_{th}. The main components of hardware implementations of procedures $\mathsf{MAC}()$ and $\mathsf{Verify}()$ of the mode are: (i) multipliers and adders in the field $\mathbb{F}_{2^N}$, (ii) bit counters, used in the Hamming weight test performed by procedure $\mathsf{Verify}()$, and (iii) block ciphers.

The implementation of such components is known and their cost reasonable. For example, we have built multipliers in the field $\mathbb{F}_{2^{128}}$, defined by the irreducible polynomial of the GCM mode [5], using Intel's ® 14 nm process technology, and found that each multiplier is associated with an area cost of 3,400 μm^2 and 46,575 logic gates. Similarly the area cost of a procedure performing a Hamming weight test over 128-bit values is 131.4 μm^2 and 1,800 logic gates, using the same process technology. Plaintext encryption and blinding can be performed by standard ciphers and modes such as AES [6], XTS [7] or GCM [5]. The choice of cipher and mode may depend on additional security requirements imposed by the application that uses the MAGIC mode.

Finally, we note that the test which decides whether a hash key value H, drawn uniformly from $\{0, 1\}^N$, is also in the set $\mathcal{H}$, can run in reasonable amount of time and be implemented with reasonable cost. This is a one time test performed when the secret H of the mode is initialized. The complexity of such test is $O(n \cdot |\mathcal{E}_{\mathcal{T}}|)$. This is because, there are $2n - 2$ powers of H to be tested, and each of those powers needs to be multiplied with $|\mathcal{E}_{\mathcal{T}}|$ error values e. The test is successful if the Hamming weight of all products is higher than the threshold T_{th}. Even a naive implementation of this test, using a circuit that iterates over all possible powers and values for e, can execute in reasonable time. For example, if $N = 128$, $n = 4$ and $T_{th} = 5$, the number of such iterations is 2^{30}, which can complete in the order of seconds by an efficient hardware implementation. Furthermore, future work on more efficient algorithms for this test may allow for higher values of the threshold T_{th} to be applicable, when the MAGIC mode is in use.

A different research direction for future work is to consider instantiations of the MAGIC mode, where the message authentication code consists of a plurality of independent MAGIC tags. In these instantiations, tags are computed on the same message blocks independently, and each tag is derived from a different pair of secrets H and $\mathsf{E}_{K_B}()$. These instantiations may be both secure and support improved error correcting capability. Indeed, it may be possible to perform error correction in more than one input block using such variants of MAGIC. Due to their bit linearity, multiple Galois Hash values, produced using different hash keys, may be able to correct errors in as many input blocks as the number of such hash values.

A last direction for future work is to consider MAGIC in the presence of errors, the distribution of which is biased toward values associated with specific patterns. In this case, the more generic Hamming weight test, discussed in this paper, may need to be replaced by other entropy tests which are specific to the distribution of the errors corrected by the MAGIC mode. The design and security analysis of such variants is yet to be done.

References

[1] *Secure Hash Standard*, Federal Information Processing Standards Publication FIPS PUB 180-4.

[2] *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*, Federal Information Processing Standards Publication FIPS PUB 202.

[3] *The Keyed-Hash Message Authentication Code (HMAC)*, Federal Information Processing Standards Publication FIPS PUB 198-1.

[4] *SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash and ParallelHash*, NIST Special Publication 800-185.

[5] M. Dworkin, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*, NIST Special Publication 800-38D.

[6] *Advanced Encryption Standard (AES)*, Federal Information Processing Standards Publication FIPS PUB 197.

[7] *Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices*, NIST Special Publication 800-38E.

[8] I. S. Reed, and G. Solomon, *Polynomial Codes over Certain Finite Fields*, Journal of the Society for Industrial and Applied Mathematics, vol. 8(2), pp. 300–304,1980.

[9] J. L. Massey, *Shift-register synthesis and BCH decoding*, IEEE Transactions on Information Theory, vol. 15(1): pp. 122–127, 1969.

[10] E. R. Berlekamp, *Algebraic Coding Theory (Revised ed.)*, Laguna Hills, CA, Aegean Park Press, 1984.

[11] V. Guruswami and M. Sudan, *Improved decoding of Reed–Solomon codes and algebraic geometry codes*, IEEE Transactions on Information Theory, vol. 45(6): pp. 1757–1767, 1999.

[12] R. Koetter and A. Vardy, *Algebraic soft-decision decoding of Reed–Solomon codes*, IEEE Transactions on Information Theory, vol. 49(11): pp. 2809–2825, 2003.

[13] H. Krawczyk, *LFSR-based Hashing and Authentication*, Proceedings of CRYPTO 1994, LNCS vol. 839, Springer Heidelberg (1994).

[14] C. G. Boncelet, *The NTMAC for Authentication of Noisy Messages*, IEEE Transactions on Information Forensics and Security vol. 1(1), pp. 35-42, 2006.

[15] Y Liu and C. G. Boncelet, *The CRC-NTMAC for Authentication of Noisy Messages*, IEEE Transactions on Information Forensics and Security vol. 1(4), pp. 517-523, 2006.

[16] Y. Liu and C. G. Boncelet, *The BCH-NTMAC for Authentication of Noisy Messages*, Proceedings of the 40th Annual Conference on Information Sciences and Systems, pp. 246-251, 2006.

[17] C. C. Y. Lam, G. Gong and S. A. Vanstone, *Message Authentication Codes with Error Correcting Capabilities*, Proceedings of ICICS 2002, LNCS vol. 2513, pp. 354-366, 2002.

[18] A. Sengupta, D. Saha, S. Ghosh, D. Mehta and D. R. Chowdhury, *AEC: A Practical Scheme for Authentication with Error Correction*, Proceedings of SPACE 2014, LNCS vol. 8804, pp. 155-170, 2014.

[19] M. Ayoob and W. Adi, *Improving System Reliability by Joint Usage of Hash Function Bits and Error Correction Coding*, Proceedings of the Sixth International Conference on Emerging Security Technologies, 2015.

[20] G. Saileshwar, P. J. Nair, P. Ramrakhyani, W. Elsasser and M. K. Qureshi, *SYNERGY: Rethinking Secure-Memory Design for Error-Correcting Memories*, Proceedings of the Sixth International Conference on Emerging Security Technologies, 2015.

[21] L. Carter and M. Wegman, *Universal classes of hash functions*, Journal of Computer and System Sciences, vol. 18, no. 2, pp. 143–154, 1979.

[22] B. Cogliati and Y. Seurin, *EWCDM: An Efficient, Beyond-Birthday Secure, Nonce-Misuse Resistant MAC*, Proceedings of CRYPTO 2016, pp.121-149, 2016.

[23] M. Bellare, O. Goldreich and A. Mityagin, *The power of verification queries in message authentication and authenticated encryption*, Cryptology ePrint Archive: Report 2004/309, 2004.

[24] M. Bellare, *New Proofs for NMAC and HMAC: Security without Collision-Resistance*, Proceedings of CRYPTO 2006, LNCS vol. 4117, Springer-Verlag, 2006.

[25] D. J. Bernstein, *The Poly1305-AES Message-Authentication Code*, Proceedgins of Fast Software Encryption, LNCS vol. 3557, Springer Verlag, pp. 32–49, 2005.

[26] S. Gueron and Y. Lindell, *GCM-SIV: Full Nonce Misuse-Resistant Authenticated Encryption at Under One Cycle per Byte*, Proceedings of the 2015 ACM Conference on Computer and Communications Security (ACM CCS) pp. 109-119, 2015.

[27] R. J. McEliece, *A Public-Key Cryptosystem Based On Algebraic Coding Theory*, DSN Progress Report 42-44, pp. 114–116, 1978.

[28] T. Johansson, *Contributions to unconditionally secure authentication*, Ph.D Thesis, KF-Sigma Publishers, Lund Technical University, 1994.

[29] T. Johansson, G. Kabatianskii and B. Smeets, *On the relation between A-codes and codes correcting independent errors*, Proceedings of Eurocrypt 1993, 1993.

[30] C. S. Ding and X. S. Wang, *A coding theory construction of new systematic authentication codes*, Theoretical Computer Science, vol. 330, pp. 81-99, 2005.

[31] C. Carlet, C. S. Ding and H. Niederreiter, *Authentication Schemes from Highly Nonlinear Functions*, Design Codes and Cryptography, vol. 40, pp. 71-79, 2006.

[32] C. S. Ding, T. Helleseth, T. Kløve and X. S. Wang, *A generic construction of Carthesian authentication codes*, IEEE Transactions on Information Theory, vol. 53(6): pp. 2229–2235, 2007.

[33] H. Tilborg, *Authentication Codes from Error-Correcting Codes; An Overview*, Enhancing Cryptographic Primitives with Techniques from Error Correcting Codes, B. Preneel et al. (Eds), IOS Press, 2009.

Appendix: Glossary of Terms

A.1 Terms introduced in Section 1

blinding: The process of concealing the Galois Hash output of the MAGIC mode by means of encryption.

$\mathsf{MAC}_K()$: A sign oracle that uses key K.

T: The tag produced by sign oracle $\mathsf{MAC}_K()$.

Q_s: A set of sign queries.

Q_v: A set of verification queries.

$\mathcal{A}$: A MAC adversary.

$\theta^{\mathsf{ECC}()}(M, T)$: The set of message-tag pairs which, after error correction performed by $\mathsf{ECC}()$ become (M, T).

$\Theta^{\mathsf{ECC}()}(Q_s)$: The set of message-tag pairs which are correctable to one of the sign queries of set Q_s.

MAGIC: Message Authentication, Galois Integrity and Correction.

A.2 Terms introduced in Section 2

n: The number of input blocks which are encrypted. It is considered fixed.

N: The block size in bits.

H: The hash key of the MAGIC mode.

$\mathbb{F}_{2^N}$: The finite field over which the Galois Hash computations of the mode are defined.

$\mathcal{H}$: The set of hash key values from which H is uniformly drawn.

$\mathsf{E}_{K_e,\mathcal{M}_e,i_e,N}()$: The cipher which encrypts the mode's input plaintext blocks. This cipher is also denoted by $\mathsf{E}_{K_e}()$.

K_e: The key used by the cipher $\mathsf{E}_{K_e}()$. This is a bit string of length l_{K_e}.

$\mathcal{M}_e$: The mode of the cipher $\mathsf{E}_{K_e}()$.

i_e: The initialization vector used by the cipher $\mathsf{E}_{K_e}()$. This is a bit string of length l_{i_e}.

$\mathsf{E}_{K_B,\mathcal{M}_B,i_B,N}()$: The cipher which performs blinding. This cipher encrypts the tag of the mode. It is also denoted by $\mathsf{E}_{K_B}()$.

K_B: The key used by the cipher $\mathsf{E}_{K_B}()$. This is a bit string of length l_{K_B}.

$\mathcal{M}_B$: The mode of the cipher $\mathsf{E}_{K_B}()$.

i_B: The initialization vector used by the cipher $\mathsf{E}_{K_B}()$. This is a bit string of length l_{i_B}.

$\mathcal{P}$: The set of all bijective functions from the set $\{0,1\}^N$ to the set $\{0,1\}^N$.

$\mathbf{Adv}^{\mathsf{E}_{K_B}()}$: The advantage of distinguishing the blinding cipher from a permutation randomly chosen from $\mathcal{P}$.

$\mathsf{Str}(C_1, C_2, \ldots)$: Operator returning the binary representation of the concatenated strings $C_1, C_2, \ldots$. If the input is a single string, the operator is omitted.

$\mathsf{MAC}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}$: Sign oracle of the MAGIC mode. It uses a Galois Hash key value H, plaintext encrypting cipher $\mathsf{E}_{K_e}()$ and blinding cipher $\mathsf{E}_{K_B}()$. This oracle is also denoted by $\mathsf{MAC}_H^{\mathsf{E}_{K_B}()}$.

$\mathsf{Verify}_H^{\mathsf{E}_{K_e}(),\mathsf{E}_{K_B}()}$: Verification oracle of the MAGIC mode. It uses a Galois Hash key value H, plaintext encrypting cipher $\mathsf{E}_{K_e}()$ and blinding cipher $\mathsf{E}_{K_B}()$. This oracle is also denoted by $\mathsf{Verify}_H^{\mathsf{E}_{K_B}()}$.

$M_1, M_2, \ldots, M_n$: Input plaintext blocks

$C_1, C_2, \ldots, C_n$: Ciphertext blocks that result from encrypting $M_1, M_2, \ldots, M_n$ using cipher $\mathsf{E}_{K_e}()$.

T_{th}: Hamming weight threshold.

S: A syndrome value.

S_i: An error location indicator, associated with index $i \in [1, n]$.

$\mathcal{H}_{\mathcal{E}}$: The set of excluded Galois Hash key values.

$\mathcal{E}_{\mathcal{T}}$: The set of values in $\{0,1\}^N$ the Hamming weight of which is at most T_{th}.

$\theta_{cipher}(C_1, C_2, \ldots, C_n, T_{th})$: The set of ciphertext blocks which are correctable to $C_1, C_2, \ldots, C_n$ by the MAGIC mode.

$\theta_{tag}(T, T_{th})$: The set of tag values which are probabilistically correctable to T by the MAGIC mode.

$\vartheta_H^{\mathsf{E}_{K_B}()}(T, T_{th})$: Subset of $\theta_{tag}(T, T_{th})$ that includes tag values which are deterministically correctable to T by the MAGIC mode. It depends on the Galois Hash key value H and the blinding cipher $\mathsf{E}_{K_B}()$.

$|\ |$: Cardinality of a set

$\xleftarrow{\$}$: sampling from a uniform distribution

$\neg$: Logical complement

$\vee$: Logical OR between conditions

$\wedge$, " , ", " ; ": Alternatives for logical AND between conditions

A.3 Terms introduced in Section 3

$q_s^{(i)}$: Sign query of index i. Index i indicates order of issue inside the set Q_s.

$q_v^{(i)}$: Verification query of index i. Index i indicates order of issue inside the set Q_v.

$q^{[i]}$: Sign or verification query of index i. Index i indicates order of issue inside the set $Q = Q_s \cup Q_v$.

$(D^{(i)}, C_1^{(i)}, \ldots, C_n^{(i)})$: The authenticated data and ciphertext blocks of a sign query. Index i indicates order of issue inside the set Q_s.

$(D^{[i]}, C_1^{[i]}, \ldots, C_n^{[i]})$: The authenticated data and ciphertext blocks of a sign query. Index i indicates order of issue inside the set $Q = Q_s \cup Q_v$.

$(D^{(i)}, C_1^{(i)}, \ldots, C_n^{(i)}, T^{(i)})$: The authenticated data, ciphertext blocks and tag of a verification query. Index i indicates order of issue inside the set Q_v.

$(D^{[i]}, C_1^{[i]}, \ldots, C_n^{[i]}, T^{[i]})$: The authenticated data, ciphertext blocks and tag of a verification query. Index i indicates order of issue inside the set $Q = Q_s \cup Q_v$.

up to query $q^{[i]}$: Reference to all queries of a set from 1 ro i, including query

$q^{[i]}$. The term "up until query $q^{[i]}$" has the same meaning.

$Q_s^{[i]}$: Set of all sign queries issued up to query $q^{[i]}$.

$Q_v^{[i]}$: Set of all verification queries issued up to query $q^{[i]}$.

$\mathcal{A}^{[i]}$: Same as adversary $\mathcal{A}$, but acts upon information coming only from queries $q^{[1]}, \ldots, q^{[i]}$ and their responses.

$\mathcal{O}()$: Oracle response coming from the attacked sign and verification oracles of the MAGIC mode.

$\mathcal{W}^{\mathcal{A}}$: The event of adversary $\mathcal{A}$ winning the MAC forgery game.

$\mathcal{A}^*$: Adversary attacking a MAGIC mode for which blinding is performed by a random permutation.

$\mathbf{Adv}^{\mathcal{A}^*}$: The advantage of adversary $\mathcal{A}^*$.

$\mathcal{W}^{\mathcal{A}^*}$: The event of adversary $\mathcal{A}^*$ winning the MAC forgery game.

π^*: A permutation randomly chosen from the set $\mathcal{P}$.

$\mathsf{MAC}_H^{\pi^*()}$: Sign oracle of the MAGIC mode, for which blinding is performed by random permutation $\pi^*()$.

$\mathsf{Verify}_H^{\pi^*()}$: Verification oracle of the MAGIC mode, for which blinding is performed by random permutation $\pi^*()$.

$\mathcal{E}_{\mathcal{C}}(i_{\mathcal{C}})$: A MAC collision event associated with a query of index $i_{\mathcal{C}}$. Index $i_{\mathcal{C}}$ indicates order of issue inside the set Q.

$\mathcal{E}_{\mathcal{G}}(i_{\mathcal{G}})$: A Galois Hash computation event associated with a query of index $i_{\mathcal{G}}$. Index $i_{\mathcal{G}}$ indicates order of issue inside the set Q.

$\mathcal{E}_{\mathcal{X}}(i_{\mathcal{X}})$: An ECC function exploitation event associated with a query of index $i_{\mathcal{X}}$. Index $i_{\mathcal{X}}$ indicates order of issue inside the set Q.

$\mathcal{E}_{\mathcal{B}}(i_{\mathcal{B}})$: A blind forgery event associated with a query of index $i_{\mathcal{B}}$. Index $i_{\mathcal{B}}$ indicates order of issue inside the set Q.

$F_{\mathcal{C}}(i_{\mathcal{C}})$: Probability of the first occurrence of a MAC collision event associated with a query of index $i_{\mathcal{C}}$. Index $i_{\mathcal{C}}$ indicates order of issue inside the set Q.

$F_{\mathcal{G}}(i_{\mathcal{G}})$: Probability of the first occurrence of a Galois Hash computation event associated with a query of index $i_{\mathcal{G}}$. Index $i_{\mathcal{G}}$ indicates order of issue inside the set Q.

$F_{\mathcal{X}}(i_{\mathcal{X}})$: Probability of the first occurrence of an ECC function exploitation event associated with a query of index $i_{\mathcal{X}}$. Index $i_{\mathcal{X}}$ indicates order of issue inside the set Q.

$F_{\mathcal{B}}(i_{\mathcal{B}})$: Probability of the first occurrence of a blind forgery event associated with a query of index $i_{\mathcal{B}}$. Index $i_{\mathcal{B}}$ indicates order of issue inside the set Q.

$\mathcal{N}(i)$: The event that no security critical events occur up to query $q^{[i]}$.

$\mathcal{N}^{\mathcal{A}^*}$: The event that no security critical events occur during the MAC forgery game played by $\mathcal{A}^*$.

$\mathcal{S}^{\mathcal{A}^*}$: The event that at least one security critical event occurs during the MAC forgery game played by $\mathcal{A}^*$. This is the complementary of the event $\mathcal{N}^{\mathcal{A}^*}$.

$\mathcal{H}_{\mathcal{I}}^{[i]}$: Set of impossible Galois Hash keys, determined from queries and oracle responses up to query $q^{[i]}$, and the knowledge that no security critical events occur up to $q^{[i]}$.

$\mathcal{H}_{\mathcal{I},\mathsf{COL}}^{[i]}$: Set of impossible Galois Hash keys, determined from queries and oracle responses up to query $q^{[i]}$, and the knowledge that no MAC collisions occur up to $q^{[i]}$.

$\mathcal{H}_{\mathcal{I},\mathsf{EXPL}}^{[i]}$: Set of impossible Galois Hash keys, determined from queries and oracle responses up to query $q^{[i]}$, and the knowledge that no ECC function exploitation events occur up to $q^{[i]}$.

ζ: Entity equal to $\frac{n(n-1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|^2$.

η: Entity equal to $\frac{n(n+1)}{2} \cdot |\mathcal{E}_{\mathcal{T}}|$.

$\mathcal{H}_{\mathcal{P}}^{[i]}$: Set of possible Galois Hash keys, determined from queries and oracle responses up to query $q^{[i]}$, and the knowledge that no security critical events occur up to $q^{[i]}$. This set is equal to $\mathcal{H} - \mathcal{H}_{\mathcal{I}}^{[i]}$.

$H_{\mathcal{P}}^{[i]}$: An element of $\mathcal{H}_{\mathcal{P}}^{[i]}$.

$\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$: The set of constraint satisfying permutations associated with the Galois hash key value $\mathcal{H}_{\mathcal{P}}^{[i]}$.

$\rho()$: A blinding permutation of the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$.

$\mathsf{MAC}_{H_{\mathcal{P}}^{[i]}}^{\rho()}$: An alternative sign oracle, different from the attacked one $\mathsf{MAC}_{H}^{\pi^*()}$. This oracle uses the Galois Hash key $H_{\mathcal{P}}^{[i]}$ and the blinding permutation $\rho()$ from the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$. It provides the same responses as oracle $\mathsf{MAC}_{H}^{\pi^*()}$ to adversary $\mathcal{A}^*$'s sign queries from $q^{[1]}, \ldots, q^{[i]}$.

$\mathsf{Verify}_{H_{\mathcal{P}}^{[i]}}^{\rho()}$: An alternative verification oracle, different from $\mathsf{Verify}_{H}^{\pi^*()}$. This oracle uses the Galois Hash key $H_{\mathcal{P}}^{[i]}$ and the blinding permutation $\rho()$ from the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$. It provides the same responses as oracle $\mathsf{Verify}_{H}^{\pi^*()}$ to adversary $\mathcal{A}^*$'s verification queries from $q^{[1]}, \ldots, q^{[i]}$.

$g_{H_{\mathcal{P}}^{[i]}}^{(j)}$: Galois Hash output computed on the authenticated data and ciphertext blocks of sign query $q_s^{(j)}$ or verification query $q_v^{(j)}$. The key used is $H_{\mathcal{P}}^{[i]}$.

$\mathcal{G}_{H_{\mathcal{P}}^{[i]}}^{(j)}$: The set of Galois Hash outputs computed on the authenticated data and ciphertext blocks of the sign queries of $Q_s^{[i]}$. The key used is $H_{\mathcal{P}}^{[i]}$.

$\mathcal{T}^{[i]}$: The set of responses coming from the attacked oracle $\mathsf{MAC}_{H}^{\pi^*()}$ to the sign queries of $Q_s^{[i]}$. The responses coming from the attacked oracle are referred to using the $\mathcal{O}()$ operator.

$\mathcal{S}_{H_{\mathcal{P}}^{[i]}}$: Set of mappings between the Galois Hash outputs computed on the authenticated data and ciphertext blocks of the sign queries of $Q_s^{[i]}$, and their corresponding responses coming from the attacked oracle $\mathsf{MAC}_{H}^{\pi^*()}$. The hash key used for producing the mappings is not H but $H_{\mathcal{P}}^{[i]}$.

$\mathcal{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$: Set of decrypted tag values compatible with query $q_v^{(j)}$ and the hash key value $H_{\mathcal{P}}^{[i]}$.

$Q_{v,\mathsf{EQT}j}^{[i]}$: Set of verification queries from the set $Q_v^{[i]}$ that include the same tag $T^{(j)}$ as query $q_v^{(j)} \in Q_v^{[i]}$.

$\mathsf{V}_{H_{\mathcal{P}}^{[i]}}^{(j)}$: Set of decrypted tag values compatible with all queries of set $Q_{v,\mathsf{EQT}j}^{[i]}$ and with the hash key value $H_{\mathcal{P}}^{[i]}$.

$\mathcal{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$: Set of encrypted Galois Hash outputs compatible with query $q^{(j)}_v$ and the hash key value $H^{[i]}_{\mathcal{P}}$.

$Q^{[i]}_{v,H^{[i]}_{\mathcal{P}},\mathsf{EQG}j}$: Set of verification queries from the set $Q^{[i]}_v$ that include authenticated data and ciphertext blocks producing the same Galois Hash output as query $q^{(j)}_v \in Q^{[i]}_v$. The hash key considered in the formation of the set is not H but $H^{[i]}_{\mathcal{P}}$.

$\mathsf{U}^{(j)}_{H^{[i]}_{\mathcal{P}}}$: Set of encrypted Galois Hash outputs compatible with all queries of the set $Q^{[i]}_{v,H^{[i]}_{\mathcal{P}},\mathsf{EQG}j}$ and with the hash key value $H^{[i]}_{\mathcal{P}}$.

$\{m_1, m_2, \ldots, m_{n_m}\}$: Set passed as input to procedure $\mathsf{CombineMappings}()$. Each element m_{i_m}, $i_m \in [1, n_m]$ is a set of mappings itself.

n_m: The cardinality of the set $\{m_1, m_2, \ldots, m_{n_m}\}$ passed as input to procedure $\mathsf{CombineMappings}()$.

$\{\mu_1, \mu_2, \ldots, \mu_{n_m}\}$: A single element of the set returned by procedure $\mathsf{CombineMappings}()$. This element is a set itself containing the mappings μ_1, μ_2, $\ldots$, μ_{n_m}. Mapping μ_1 is taken from set m_1. Mapping μ_2 is taken from set m_2, and so on.

M_{DISTINCT}: The set returned by procedure $\mathsf{CombineMappings}()$ in line 17 of the pseudocode of $\mathsf{SelectPermutations}()$.

ξ_j: The j-th element of set M_{DISTINCT}.

$f_{H^{[i]}_{\mathcal{P}},j}$: The j-th set passed as input to procedure $\mathsf{MappingsToPermutations}()$. This happens either in line 20 of procedure $\mathsf{SelectPermutations}()$, in which case $f_{H^{[i]}_{\mathcal{P}},j}$ is equal to $\mathcal{S}_{H^{[i]}_{\mathcal{P}}} \cup \xi_j$, or in line 23 of the same procedure, in which case, $f_{H^{[i]}_{\mathcal{P}},j}$ is equal to $\mathcal{S}_{H^{[i]}_{\mathcal{P}}}$. The computations of procedure $\mathsf{SelectPermutations}()$ are performed for the hash key value $H^{[i]}_{\mathcal{P}}$.

$|\mathcal{P}_{H^{[i]}_{\mathcal{P}}}|$: The number of sets of permutations returned by procedure $\mathsf{MappingsToPermutations}()$. If the set of sign queries $Q^{[i]}_s$ is non-empty, this number is equal to the cardinality of M_{DISTINCT}. If $Q^{[i]}_s$ is empty this number is equal to 1. The computations of the calling procedure $\mathsf{SelectPermutations}()$ are performed for the hash key value $H^{[i]}_{\mathcal{P}}$.

$\mathcal{F}_{H^{[i]}_{\mathcal{P}},j}$: The j-th set returned by procedure $\mathsf{MappingsToPermutations}()$. This set is either returned in line 20 of procedure $\mathsf{SelectPermutations}()$, in which case, it is also denoted by $M_{\mathsf{PERM},j}$, or in line 23 of the same procedure, in which case it is exactly equal to the set $C_{H_{\mathcal{P}}^{[i]}}$ returned by SelectPermutations(). The computations of procedure SelectPermutations() are performed for the hash key value $H_{\mathcal{P}}^{[i]}$.

$M_{\mathsf{PERM},j}$: Alternative notation for the set $\mathcal{F}_{H_{\mathcal{P}}^{[i]},j}$, which is used in line 20 of the pseudocode of procedure SelectPermutations().

$C_{H_{\mathcal{P}}^{[i]}}$: The set returned by procedure SelectPermutations(). Corollary 6 establishes that this set is equal to the set $\mathcal{C}_{H_{\mathcal{P}}^{[i]}}$ of the constraint satisfying permutations associated with the hash key $H_{\mathcal{P}}^{[i]}$, i.e., $C_{H_{\mathcal{P}}^{[i]}} = \mathcal{C}_{H_{\mathcal{P}}^{[i]}}$.

Z: Entity equal to $(1 + \frac{4\cdot|Q|}{2^N-(\eta+5)|Q|})^{2|Q|}$.

e: The base of the natural logarithm.