Breaking and Repairing GCM Security Proofs

Tetsu Iwata[1], Keisuke Ohashi[1], and Kazuhiko Minematsu[2]

[1] Nagoya University, Japan
iwata@cse.nagoya-u.ac.jp, k_oohasi@echo.nuee.nagoya-u.ac.jp
[2] NEC Corporation, Japan
k-minematsu@ah.jp.nec.com

Abstract. In this paper, we study the security proofs of GCM (Galois/Counter Mode of Operation). We first point out that a lemma, which is related to the upper bound on the probability of a counter collision, is invalid. Both the original privacy and authenticity proofs by the designers are based on the lemma. We further show that the observation can be translated into a distinguishing attack that invalidates the main part of the privacy proof. It turns out that the original security proofs of GCM contain a flaw, and hence the claimed security bounds are not justified. A very natural question is then whether the proofs can be repaired. We give an affirmative answer to the question by presenting new security bounds, both for privacy and authenticity. As a result, although the security bounds are larger than what were previously claimed, GCM maintains its provable security. We also show that, when the nonce length is restricted to 96 bits, GCM has better security bounds than a general case of variable length nonces.

Keywords: GCM, counter-example, distinguishing attack, proof of security.

1 Introduction

GCM (Galois/Counter Mode of Operation) is the authenticated encryption mode of blockciphers designed by McGrew and Viega [26,27]. The mode is based on the counter mode encryption and the polynomial hash function, and the designers presented proofs of security both for privacy and authenticity [26,27]. It was selected as the NIST recommended blockcipher mode in 2007 [15], and is widely used in practice, e.g., in [1,2,4,5,6,7,8,14,17,19,20,25,33,34].

The security of GCM has been extensively evaluated. Ferguson pointed out that a forgery is possible if the tag length is short [16]. Joux showed that a part of the secret key can be obtained if the nonce is reused [21]. Handschuh and Preneel discussed weak keys of GCM and presented generalizations of Joux's attack [18]. Saarinen pointed out that GCM has more weak keys than previously known, and used the weak keys for forgery attacks [32]. See also [31] for comprehensive discussions on various aspects on GCM.

Despite aforementioned attacks, it is widely considered that the provable security results of GCM are sound, in the sense that the previous attacks do

not contradict the claimed security bounds by the designers, and that no flaw in the proofs has been identified. Some of these attacks show the tightness of the security bounds, and others are outside the security model (e.g., nonce reuse). Therefore, there is no attack that undermines the security bounds or their proofs.

GCM uses the counter mode encryption, and the initial counter value is derived from a nonce, where there are two different ways to generate the initial counter value depending on the length of the nonce. When the nonce length is 96 bits, the initial counter value is the nonce padded with a constant. When the nonce length is not 96 bits, the polynomial hash function is applied to the nonce to obtain the initial counter value. In order to prove the security of GCM, one has to show that the probability of a counter collision is small. McGrew and Viega presented a lemma showing the upper bound on the probability in [27], which is the basis for both the privacy and authenticity proofs.

In this paper, we first point out that the claimed lemma cannot be valid; the probability of a counter collision is larger than claimed. We show concrete counter-examples of two distinct nonces that invalidate the lemma. It turns out that the original security proofs (both for privacy and authenticity) of GCM contain a flaw, and hence the claimed security bounds are not justified.

We next translate the above observation into a distinguishing attack. The attack is simple and efficient. However, from the practical perspective, the success probability of the attack is insignificantly small, and it does not contradict the security bounds by the designers. On the other hand, the success probability is large enough to invalidate the main part of the privacy proof. In more detail, there are three terms in the privacy bound of GCM. The first one comes from the difference between a random permutation and a random function, the second one is the main part of the privacy proof that bounds the distinguishing probability of ciphertexts of GCM based on a random function from random strings (of the same lengths as the ciphertexts), and the last one bounds the forgery probability. The success probability of our distinguishing attack is larger than the second term, invalidating the main part of the privacy proof. Consequently, the security of GCM is not supported by the proofs.

Then a very natural question is whether the proofs can be repaired, or more generally, whether the security of GCM can ever be proved. In order to answer the question, we first introduce a combinatorial problem of quantifying a cardinality of a certain set of bit strings. The problem belongs to one of the problems of counting the number of output differences with non-zero probability of S-functions [29], which presents tools to analyze ARX systems (e.g., see [23]). One possible approach to solve the problem is to follow [29] (or [22]). In this paper, we take another approach and present a solution to the problem by giving a recursive formula that quantifies the cardinality. Basing on the solution, we present new security bounds on GCM, both for privacy and authenticity.

As a result, although the security bounds are larger than what were previously claimed, we show that GCM maintains its provable security. We also present provable security results of GCM when the nonce length is restricted to 96 bits, in which case GCM has better security bounds than a general case.

2 Preliminaries

Let $\{0,1\}^*$ be the set of all bit strings, and for an integer $\ell \geq 0$, let $\{0,1\}^\ell$ be a set of ℓ-bit strings. For a bit string $X \in \{0,1\}^*$, $|X|$ is its length in bits, and $|X|_\ell = \lceil |X|/\ell \rceil$ is the length in ℓ-bit blocks. The empty string is denoted as ε. Let 0^ℓ and 1^ℓ denote the bit strings of ℓ zeros and ones, respectively. We use the prefix $\mathtt{0x}$ for the hexadecimal notation, e.g., $\mathtt{0x63}$ is $01100011 \in \{0,1\}^8$. We also write $(\mathtt{0x0})^\ell$ to mean $0^{4\ell}$. For a bit string X and an integer ℓ such that $|X| \geq \ell$, $\mathsf{msb}_\ell(X)$ is the most significant ℓ bits (the leftmost ℓ bits) of X, and $\mathsf{lsb}_\ell(X)$ is the least significant ℓ bits (the rightmost ℓ bits) of X. For $X, Y \in \{0,1\}^*$, we write $X \,\|\, Y$, (X, Y), or simply XY to denote their concatenation. For a bit string X whose length in bits is a multiple of ℓ, we write its partition into ℓ-bit strings as $(X[1], \ldots, X[x]) \xleftarrow{\ell} X$, where $X[1], \ldots, X[x] \in \{0,1\}^\ell$ are unique bit strings such that $X[1] \,\|\, \ldots \,\|\, X[x] = X$. For non-negative integers a and ℓ with $a \leq 2^\ell - 1$, let $\mathsf{str}_\ell(a)$ be its ℓ-bit binary representation, i.e., if $a = a_{\ell-1}2^{\ell-1} + \cdots + a_1 2 + a_0$ for $a_{\ell-1}, \ldots, a_1, a_0 \in \{0,1\}$, then $\mathsf{str}_\ell(a) = a_{\ell-1} \ldots a_1 a_0 \in \{0,1\}^\ell$. For a bit string $X = X_{\ell-1} \ldots X_1 X_0 \in \{0,1\}^\ell$, let $\mathsf{int}(X)$ be the integer $X_{\ell-1}2^{\ell-1} + \cdots + X_1 2 + X_0$. For a finite set $\mathcal{X}$, $\#\mathcal{X}$ denotes its cardinality, and $X \xleftarrow{\$} \mathcal{X}$ means the uniform sampling of an element from $\mathcal{X}$ and assigning it to X.

Throughout this paper, we fix a block length n and a blockcipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{K}$ is a non-empty set of keys. Unless otherwise specified, we let $n = 128$. We write E_K for the permutation specified by $K \in \mathcal{K}$, and $C = E_K(M)$ for the ciphertext of a plaintext $M \in \{0,1\}^n$ under the key $K \in \mathcal{K}$. The set of n-bit strings, $\{0,1\}^n$, is also regarded as $\mathrm{GF}(2^n)$, the finite field with 2^n elements. An n-bit string $a_{n-1} \ldots a_1 a_0 \in \{0,1\}^n$ corresponds to a formal polynomial $a(\mathtt{x}) = a_{n-1} + a_{n-2}\mathtt{x} + \cdots + a_1 \mathtt{x}^{n-2} + a_0 \mathtt{x}^{n-1} \in \mathrm{GF}(2)[\mathtt{x}]$. When $n = 128$, the irreducible polynomial used in GCM is $p(\mathtt{x}) = 1 + \mathtt{x} + \mathtt{x}^2 + \mathtt{x}^7 + \mathtt{x}^{128}$.

3 Specification of GCM

We follow [27,28] with some notational changes. GCM is parameterized by a blockcipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and a tag length τ, where $64 \leq \tau \leq n$. We write GCM$[E, \tau]$ for GCM that uses E and τ as parameters. Let GCM-$\mathcal{E}$ be the encryption algorithm and GCM-$\mathcal{D}$ be the decryption algorithm, which are defined in Fig. 1. GCM uses the counter mode encryption and the polynomial hash function over $\mathrm{GF}(2^n)$ as its subroutines. They are denoted CTR and GHASH and are defined in Fig. 2. Figure 3 illustrates the overall structure of GCM-$\mathcal{E}$.

GCM-$\mathcal{E}$ takes a key $K \in \mathcal{K}$, a nonce $N \in \{0,1\}^*$, an associated data $A \in \{0,1\}^*$, and a plaintext $M \in \{0,1\}^*$ as inputs, and returns a pair of a ciphertext $C \in \{0,1\}^*$ and a tag $T \in \{0,1\}^\tau$ as an output, where $1 \leq |N| \leq 2^{n/2} - 1$, $0 \leq |A| \leq 2^{n/2} - 1$, $0 \leq |M| \leq n(2^{32} - 2)$, and $|C| = |M|$. We write $(C, T) \leftarrow \text{GCM-}\mathcal{E}_K^{N,A}(M)$. GCM-$\mathcal{D}$ takes a key $K \in \mathcal{K}$, a nonce $N \in \{0,1\}^*$, an associated data $A \in \{0,1\}^*$, a ciphertext $C \in \{0,1\}^*$, and a tag $T \in \{0,1\}^\tau$ as inputs, and returns either a plaintext $M \in \{0,1\}^*$ or the symbol $\perp$ indicating that the inputs are invalid. We write $M \leftarrow \text{GCM-}\mathcal{D}_K^{N,A}(C, T)$ or $\perp \leftarrow \text{GCM-}\mathcal{D}_K^{N,A}(C, T)$.

Algorithm GCM-$\mathcal{E}_K^{N,A}(M)$	**Algorithm** GCM-$\mathcal{D}_K^{N,A}(C,T)$
1. $L \leftarrow E_K(0^n)$	1. $L \leftarrow E_K(0^n)$
2. **if** $\|N\| = 96$ **then** $I[0] \leftarrow N \,\|\, 0^{31}1$	2. **if** $\|N\| = 96$ **then** $I[0] \leftarrow N \,\|\, 0^{31}1$
3. **else** $I[0] \leftarrow \mathsf{GHASH}_L(\varepsilon, N)$	3. **else** $I[0] \leftarrow \mathsf{GHASH}_L(\varepsilon, N)$
4. $m \leftarrow \|M\|_n$	4. $\bar{T}^* \leftarrow E_K(I[0]) \oplus \mathsf{GHASH}_L(A, C)$
5. $S \leftarrow \mathsf{CTR}_K(I[0], m)$	5. $T^* \leftarrow \mathsf{msb}_\tau(\bar{T}^*)$
6. $C \leftarrow M \oplus \mathsf{msb}_{\|M\|}(S)$	6. **if** $T \neq T^*$ **then return** $\perp$
7. $\bar{T} \leftarrow E_K(I[0]) \oplus \mathsf{GHASH}_L(A, C)$	7. $m \leftarrow \|C\|_n$
8. $T \leftarrow \mathsf{msb}_\tau(\bar{T})$	8. $S \leftarrow \mathsf{CTR}_K(I[0], m)$
9. **return** (C, T)	9. $M \leftarrow C \oplus \mathsf{msb}_{\|C\|}(S)$
	10. **return** M

Fig. 1. The encryption and decryption algorithms of GCM

Algorithm $\mathsf{CTR}_K(I[0], m)$	**Algorithm** $\mathsf{GHASH}_L(A, C)$
1. **for** $j \leftarrow 1$ **to** m **do**	1. $a \leftarrow n\|A\|_n - \|A\|$
2. $\quad I[j] \leftarrow \mathsf{inc}(I[j-1])$	2. $c \leftarrow n\|C\|_n - \|C\|$
3. $\quad S[j] \leftarrow E_K(I[j])$	3. $X \leftarrow A \,\|\, 0^a \,\|\, C \,\|\, 0^c \,\|\, \mathsf{str}_{n/2}(\|A\|) \,\|\, \mathsf{str}_{n/2}(\|C\|)$
4. $S \leftarrow (S[1], S[2], \ldots, S[m])$	4. $(X[1], \ldots, X[x]) \xleftarrow{n} X$
5. **return** S	5. $Y \leftarrow 0^n$
	6. **for** $j \leftarrow 1$ **to** x **do**
	7. $\quad Y \leftarrow L \cdot (Y \oplus X[j])$
	8. **return** Y

Fig. 2. Subroutines used in the encryption and decryption algorithms

In the definition of CTR, for a bit string $X \in \{0,1\}^n$, $\mathsf{inc}(X)$ treats the least significant 32 bits (the rightmost 32 bits) of X as a non-negative integer, and increments this value modulo 2^{32}, i.e.,

$$\mathsf{inc}(X) = \mathsf{msb}_{n-32}(X) \,\|\, \mathsf{str}_{32}(\mathsf{int}(\mathsf{lsb}_{32}(X)) + 1 \bmod 2^{32}).$$

For $r \geq 0$, we write $\mathsf{inc}^r(X)$ to denote the r times iterative applications of inc on X, and $\mathsf{inc}^{-r}(X)$ to denote the r times iterative applications of the inverse function of inc on X. We use the convention that $\mathsf{inc}^0(X) = X$. In the definition of GHASH, the multiplication in line 7 is over $\mathrm{GF}(2^n)$. We remark that, if $|N| \neq 96$, we have $\mathsf{GHASH}_L(\varepsilon, N) = X[1] \cdot L^x \oplus \cdots \oplus X[x] \cdot L$, where $X = (X[1], \ldots, X[x]) = N \,\|\, 0^{n|N|_n - |N|} \,\|\, \mathsf{str}_n(|N|)$.

4 Security Definitions

An adversary is a probabilistic algorithm that has access to one or more oracles. Let $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}$ denote an adversary $\mathcal{A}$ interacting with oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots,$

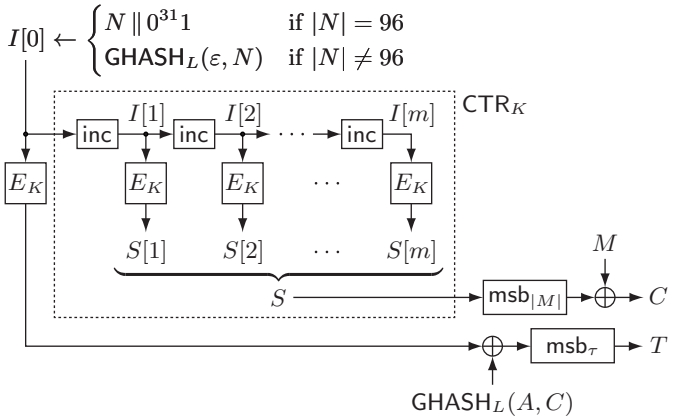

Fig. 3. The encryption algorithm of GCM

and $\mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2,\cdots} \Rightarrow 1$ denote the event that $\mathcal{A}$, after interacting with $\mathcal{O}_1, \mathcal{O}_2, \ldots$, outputs 1. The resources of $\mathcal{A}$ are measured in terms of time and query complexities. The time complexity includes the description size of $\mathcal{A}$, and we fix a model of computation and a method of encoding. The query complexity includes the number of queries, the total length of queries, and the maximum length of queries, and a more precise definition is given in each theorem statement.

Following [10,30], we consider two security notions for GCM: privacy and authenticity. For privacy, we consider an adversary $\mathcal{A}$ that has access to a GCM encryption oracle or a random-bits oracle. The GCM encryption oracle takes (N, A, M) and returns $(C, T) \leftarrow \text{GCM-}\mathcal{E}_K^{N,A}(M)$. The random-bits oracle, \$, takes (N, A, M) and returns $(C, T) \xleftarrow{\$} \{0,1\}^{|M|+\tau}$. We define

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{GCM}[E,\tau]}(\mathcal{A}) \stackrel{\mathrm{def}}{=} \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\text{GCM-}\mathcal{E}_K} \Rightarrow 1] - \Pr[\mathcal{A}^{\$} \Rightarrow 1],$$

where the first probability is defined over the randomness of $\mathcal{A}$ and the choice of K, and the last is over the randomness of $\mathcal{A}$ and the random-bits oracle. We assume that $\mathcal{A}$ is nonce-respecting: $\mathcal{A}$ does not make two queries with the same nonce.

For authenticity, we consider an adversary $\mathcal{A}$ that has access to GCM encryption and decryption oracles. The GCM decryption oracle takes (N, A, C, T) and returns $M \leftarrow \text{GCM-}\mathcal{D}_K^{N,A}(C, T)$ or $\perp \leftarrow \text{GCM-}\mathcal{D}_K^{N,A}(C, T)$. We define

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{GCM}[E,\tau]}(\mathcal{A}) \stackrel{\mathrm{def}}{=} \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\text{GCM-}\mathcal{E}_K,\text{GCM-}\mathcal{D}_K} \text{ forges}],$$

where the probability is defined over the randomness of $\mathcal{A}$ and the choice of K, and the adversary forges if the GCM decryption oracle returns a bit string (other than $\perp$) for a query (N, A, C, T), but (C, T) was not previously returned to $\mathcal{A}$ from the encryption oracle for a query (N, A, M). As in the privacy notion,

we assume that $\mathcal{A}$ is nonce-respecting: $\mathcal{A}$ does not make two queries to the encryption oracle with the same nonce. We remark that nonces used for the encryption queries can be used for decryption queries and vice-versa, and that the same nonce can be repeated for decryption queries. Without loss of generality, we assume that $\mathcal{A}$ does not make trivial queries: if the encryption oracle returns (C, T) for a query (N, A, M), then $\mathcal{A}$ does not make a query (N, A, C, T) to the decryption oracle, and $\mathcal{A}$ does not repeat a query to the decryption oracle.

In [27], McGrew and Viega analyzed the security of GCM, and there are differences between the above security notions. In [27], for privacy, the adversary has access to both the encryption and decryption oracles, while we chose to follow a more standard notion [10,30] where the privacy adversary has access to the encryption oracle only. Another difference is the assumption about the nonce reuse. In [27], the adversary is not allowed to reuse a nonce within decryption queries (but nonces used in encryption queries can be used in decryption queries and vice-versa), while our adversary can reuse nonces within decryption queries.

For the blockcipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, we consider the PRP notion [24]. Let $\mathrm{Perm}(n)$ be the set of all permutations on $\{0,1\}^n$. We say that P is a random permutation if $P \xleftarrow{\$} \mathrm{Perm}(n)$. We define

$$\mathbf{Adv}^{\mathrm{prp}}_{E}(\mathcal{A}) \stackrel{\mathrm{def}}{=} \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{E_K} \Rightarrow 1] - \Pr[P \xleftarrow{\$} \mathrm{Perm}(n) : \mathcal{A}^{P} \Rightarrow 1],$$

where the probabilities are defined over the randomness of $\mathcal{A}$, and the choices of K and P, respectively. We write GCM[$\mathrm{Perm}(n), \tau$] for GCM that uses a random permutation P as a blockcipher E_K, and we write the corresponding encryption and decryption algorithms as GCM-$\mathcal{E}_P$ and GCM-$\mathcal{D}_P$, respectively.

We also consider GCM that uses a random function as E_K, which is naturally defined as the invertibility of E_K is irrelevant in the definition of GCM. Let $\mathrm{Rand}(n)$ be the set of all functions from $\{0,1\}^n$ to $\{0,1\}^n$. We say that F is a random function if $F \xleftarrow{\$} \mathrm{Rand}(n)$, and write GCM[$\mathrm{Rand}(n), \tau$] for GCM that uses F as E_K. We write the corresponding encryption and decryption algorithms as GCM-$\mathcal{E}_F$ and GCM-$\mathcal{D}_F$, respectively.

5 Breaking GCM Security Proofs

5.1 Review of [27, Lemma 3], [27, Theorem 1], and [27, Theorem 2]

In this section, we first review a lemma in [27] that was used to derive the provable security results on GCM. Consider GCM[$\mathrm{Rand}(n), \tau$], GCM with E_K being a random function F, and the privacy notion for it. Let (N_1, A_1, M_1) and (N_2, A_2, M_2) be two encryption queries, where $N_1 \neq N_2$ and $|N_1|, |N_2| \neq 96$. Let $I_1[0] \leftarrow \mathsf{GHASH}_L(\varepsilon, N_1)$, and $I_1[j] \leftarrow \mathsf{inc}(I_1[j-1])$ for $1 \leq j \leq m_1$, where $m_1 = |M_1|_n$. Similarly, let $I_2[0] \leftarrow \mathsf{GHASH}_L(\varepsilon, N_2)$, and $I_2[j] \leftarrow \mathsf{inc}(I_2[j-1])$ for $1 \leq j \leq m_2$, where $m_2 = |M_2|_n$. If we have

$$\{I_1[0], I_1[1], \ldots, I_1[m_1]\} \cap \{I_2[0], I_2[1], \ldots, I_2[m_2]\} = \emptyset$$

and $I_i[j] \neq 0^n$ for $i = 1, 2$ and $0 \leq j \leq m_i$, then the two masks $S_1 = (S_1[1], \ldots, S_1[m_1])$ and $S_2 = (S_2[1], \ldots, S_2[m_2])$, produced from the counter mode encryption based on F, are uniformly distributed over $(\{0,1\}^n)^{m_1}$ and $(\{0,1\}^n)^{m_2}$, respectively. Furthermore, $F(I_1[0])$ and $F(I_2[0])$ are uniform random n-bit strings, and hence the probability distribution of strings returned from the encryption oracle is identical to that from the random-bits oracle.

Therefore, in order to prove the security of GCM, one has to show that the probability of a counter collision, $I_1[j_1] = I_2[j_2]$ for some $0 \leq j_1 \leq m_1$ and $0 \leq j_2 \leq m_2$, is small. We see that the event is equivalent to $\mathsf{inc}^{j_1}(\mathsf{GHASH}_L(\varepsilon, N_1)) = \mathsf{inc}^{j_2}(\mathsf{GHASH}_L(\varepsilon, N_2))$. Let $\mathsf{Coll}_L(r, N_1, N_2)$ denote the event

$$\mathsf{inc}^r(\mathsf{GHASH}_L(\varepsilon, N_1)) \oplus \mathsf{GHASH}_L(\varepsilon, N_2) = 0^n.$$

We need to bound $\Pr[L \xleftarrow{\$} \{0,1\}^n : \mathsf{Coll}_L(r, N_1, N_2)]$, where $r = j_1 - j_2$, which we write $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)]$ for simplicity. Since $-m_2 \leq r \leq m_1$ and $0 \leq m_1, m_2 \leq 2^{32} - 2$, the range of r is $-(2^{32} - 2) \leq r \leq 2^{32} - 2$.

In [27, Lemma 3], McGrew and Viega showed the following lemma (notation has been adapted to this paper).

Lemma 1 ([27]). *For any $-(2^{32}-2) \leq r \leq 2^{32}-2$, N_1, and N_2 such that $N_1 \neq N_2$, $|N_1|, |N_2| \neq 96$, and $|N_1|_n, |N_2|_n \leq \ell_N$, $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] \leq (\ell_N + 1)/2^n$.*

Based on the lemma, [27, Theorem 1] states that the privacy advantage of GCM[Perm$(n), \tau$], GCM with E_K being a random permutation P, is at most

$$\frac{0.5(\sigma/n + 2q)^2}{2^n} + \frac{2q(\sigma/n + 2q)\lceil \ell_N/n + 1 \rceil}{2^n} + \frac{q\lceil \ell/n + 1 \rceil}{2^\tau}, \tag{1}$$

and [27, Theorem 2] states that the authenticity advantage is at most

$$\frac{0.5(\sigma/n + 2q)^2}{2^n} + \frac{2q(\sigma/n + 2q + 1)\lceil \ell_N/n + 1 \rceil}{2^n} + \frac{q\lceil \ell/n + 1 \rceil}{2^\tau}, \tag{2}$$

where q is the maximum number of queries (either encryption or decryption queries), σ is the total length in bits of the plaintexts (either in encryption queries or returned from the decryption oracle), ℓ_N is the maximum length in bits of nonces in queries (either encryption or decryption queries), and ℓ is the maximum value of $|A_j| + |C_j|$, where A_j and C_j are the associated data and ciphertext, respectively, in the j-th query (either in decryption queries or returned from the encryption oracle). Note that the definitions of privacy and authenticity advantages are slightly different from ours, as explained in Sect. 4.

It is easy to see that the lemma is correct when $r = 0$: $\mathsf{Coll}_L(0, N_1, N_2)$ is the event $\mathsf{inc}^0(\mathsf{GHASH}_L(\varepsilon, N_1)) \oplus \mathsf{GHASH}_L(\varepsilon, N_2) = 0^n$, and we see that the left hand side is a non-trivial polynomial in L of degree at most $\ell_N + 1$ over GF(2^n), and hence there are at most $\ell_N + 1$ values of L that satisfy the equality.

However, for $r \neq 0$, it is not clear if $\mathsf{inc}^r(\mathsf{GHASH}_L(\varepsilon, N_1))$ is a polynomial of degree at most $\ell_N + 1$ even if this is the case for $\mathsf{GHASH}_L(\varepsilon, N_1)$, and the analysis for this case is missing in the proof of [27, Lemma 3]. The lemma is crucial in that it is used in the proofs for both privacy and authenticity.

5.2 Invalidating [27, Lemma 3]

Let $r = 1$, $N_1 = (\mathtt{0x0})^{17} \,\|\, \mathtt{0x2} = 0^{68} \,\|\, 0010$, and $N_2 = (\mathtt{0x0})^{17} \,\|\, \mathtt{0x6} = 0^{68} \,\|\, 0110$, where $|N_1| = |N_2| = 72$. Then $\mathsf{Coll}_L(r, N_1, N_2)$ is equivalent to

$$\mathsf{inc}^1(U_1 \cdot L^2 \oplus V \cdot L) \oplus (U_2 \cdot L^2 \oplus V \cdot L) = 0^n, \tag{3}$$

where $U_1 = (\mathtt{0x0})^{17} \,\|\, \mathtt{0x2} \,\|\, (\mathtt{0x0})^{14}$, $U_2 = (\mathtt{0x0})^{17} \,\|\, \mathtt{0x6} \,\|\, (\mathtt{0x0})^{14}$, and $V = (\mathtt{0x0})^{30} \,\|\, \mathtt{0x48}$. In binary, $U_1 = 0^{68} \,\|\, 0010 \,\|\, 0^{56}$, $U_2 = 0^{68} \,\|\, 0110 \,\|\, 0^{56}$, and $V = 0^{120} \,\|\, 01001000$. Now [27, Lemma 3] states that $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] \leq 2/2^n$, i.e., (3) has at most two solutions. However, one can verify (with the help of some software, e.g., [3]) that (3) has 32 solutions, which are listed in Appendix A. In other words, $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] \geq 32/2^n$ holds, and hence [27, Lemma 3] cannot be valid.

We present one more observation regarding the counter-example. Consider

$$\mathsf{inc}^2(U_1 \cdot L^2 \oplus V \cdot L) \oplus (U_2 \cdot L^2 \oplus V \cdot L) = 0^n, \tag{4}$$

$$\mathsf{inc}^4(U_1 \cdot L^2 \oplus V \cdot L) \oplus (U_2 \cdot L^2 \oplus V \cdot L) = 0^n, \tag{5}$$

where the values of U_1, U_2, and V are as above. Then one can verify that (4) has 31 solutions, and that (5) has 30 solutions, which are also listed in Appendix A. The 93 values of L are all distinct, and are also different from 0^n, which is a solution for $\mathsf{inc}^0(U_1 \cdot L^2 \oplus V \cdot L) \oplus (U_2 \cdot L^2 \oplus V \cdot L) = 0^n$. Therefore we have

$$\Pr_L\left[\bigvee_{r=0,1,2,4} \mathsf{Coll}_L(r, N_1, N_2)\right] \geq \frac{94}{2^n}. \tag{6}$$

We remark that we exclude the case $r = 3$ since $\mathsf{Coll}_L(3, N_1, N_2)$ has no solution. In Appendix A, we present other examples of (N_1, N_2) that satisfy (6) and also invalidate [27, Lemma 3].

We next show that the above observation is not merely spotting of a subtle error in the proofs of GCM. The observation can actually be translated into a distinguishing attack.

5.3 Distinguishing Attack

Consider GCM with E_K being a random function F, and the privacy notion for it. We remark that the analysis of this idealized version of GCM is essential since the main part of the privacy proof is the analysis of this case. Let N_1 and N_2 be as in Sect. 5.2, $A_1 = \varepsilon$, $M_1 = 0^{5n}$, $A_2 = \varepsilon$, and $M_2 = 0^n$.

Let $\mathcal{A}$ be an adversary that has access to an oracle $\mathcal{O}$ which is either the GCM encryption oracle or the random-bits oracle. $\mathcal{A}$ works as follows.

1. First, $\mathcal{A}$ makes two queries, (N_i, A_i, M_i) for $i = 1, 2$, and obtains $(C_i, T_i) \leftarrow \mathcal{O}(N_i, A_i, M_i)$.
2. Let $(C_1[1], \ldots, C_1[5]) \xleftarrow{n} C_1$ and output 1 if

$$C_1[1] = C_2 \text{ or } C_1[2] = C_2 \text{ or } C_1[3] = C_2 \text{ or } C_1[5] = C_2. \tag{7}$$

First, suppose that $\mathcal{O}$ is the GCM encryption oracle. If $\mathsf{Coll}_L(0, N_1, N_2) \vee \mathsf{Coll}_L(1, N_1, N_2) \vee \mathsf{Coll}_L(2, N_1, N_2) \vee \mathsf{Coll}_L(4, N_1, N_2)$, then we see that $\mathcal{A}$ outputs 1. Otherwise the probability distributions of $C_1[1]$, $C_1[2]$, $C_1[3]$, $C_1[5]$, and C_2 are exactly the same as those of returned by the random-bits oracle. In particular, (7) is satisfied with the same probability for the GCM encryption oracle and for the random-bits oracle. Therefore, we have

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{GCM}[\mathrm{Rand}(n),\tau]}(\mathcal{A}) = \Pr[\mathcal{A}^{\mathrm{GCM}\text{-}\mathcal{E}_F} \Rightarrow 1] - \Pr[\mathcal{A}^{\$} \Rightarrow 1] \geq \frac{94}{2^n}. \tag{8}$$

Now using the notation of (1) and (2), our adversary has the following query complexity: $q = 2$, $\sigma = 6n$, $\ell_N = 72$, and $\ell = 5n$. Then (1) is $50/2^n + 80/2^n + 12/2^\tau = 130/2^n + 12/2^\tau$. Therefore, the attack does not contradict the claimed privacy bound (1).

However, (1) allows the use of the GCM decryption oracle, and rounding up the details makes it sufficiently large so that our attack is tolerated in appearance. Now if we take a closer look at (1), the second term, $80/2^n$, is the main part of the privacy proof that bounds $\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{GCM}[\mathrm{Rand}(n),\tau]}(\mathcal{A})$, while the first term is from the application of the PRP/PRF switching lemma [11] and the last term bounds the forgery probability due to the use of the decryption oracle. Therefore, the above attack does invalidate the main part of the privacy proof, and we also see that it invalidates the second term of (2), which is $88/2^n$.

We have already shown that the claimed bound on $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)]$ is invalid, which implies that the claimed security bounds, (1) and (2), are not justified. Furthermore, the above attack invalidates the main part of the privacy proof. Therefore, at this point, it is fair to conclude that the security of GCM is not supported by the proofs. We note that, although our attack does not work with 96-bit nonces, this statement holds even in this case since (1) and (2) cover a general case including the case that the nonce length is restricted to 96 bits.

5.4 Remarks

Our attack is efficient. It uses two oracle calls, and the lengths of the queries are short. However, the success probability, although large enough to invalidate the second terms in (1) and (2), is insignificantly small in practice and it has a limited practical implication. We also note that many standards require or recommend using GCM with 96-bit nonces, in which case the attack does not work. Indeed, in many RFCs, such as RFC 4106 (IPsec) [34], 5647 (SSH) [20], 5288 (SSL) [33], the nonce length is fixed to 96 bits, and RFC 5084 [19] and 5116 [25] recommend 96-bit nonces. For IEEE standards, some strictly require 96-bit nonces (e.g. IEEE 802.1AE [5]) and some do not (e.g. IEEE P1619.1 [6]). There are cases where non-96-bit nonces are allowed, including NIST SP 800-38D [15], ISO/IEC 19772 [7], PKCS #11 [4], and most software libraries (e.g., Gladman's code [17], CRYPTO++ [14], Java SE [2], and BouncyCastle [1]). Finally, NSA Suite B Cryptography includes GCM with a strong recommendation (but not mandatory; see e.g. [8]) of using it with 96-bit nonces.

We emphasize that, even when non-96-bit nonces are allowed, our attack has a limited practical implication. However, it does undermine the provable security of GCM, making its provable security open. A very natural question is then whether the security of GCM can ever be proved. In the following sections, we present an affirmative answer to this question.

6 Towards Repairing the Proofs

6.1 Combinatorial Problem

To prove the security of GCM, the first step is to derive the upper bound on $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)]$. The obvious bound is $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] \leq (\ell_N + 1)/2^{n-32}$, which can be shown by ignoring the least significant 32 bits. In this section, we derive a better upper bound on $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)]$, and we first introduce a combinatorial problem for this goal.

For $0 \leq r \leq 2^{32} - 1$, let

$$\mathbb{Y}_r \stackrel{\text{def}}{=} \left\{\mathsf{str}_{32}(\mathsf{int}(Y) + r \bmod 2^{32}) \oplus Y \mid Y \in \{0,1\}^{32}\right\}. \tag{9}$$

We also let $\alpha_r \stackrel{\text{def}}{=} \#\mathbb{Y}_r$ and $\alpha_{\max} \stackrel{\text{def}}{=} \max\{\alpha_r \mid 0 \leq r \leq 2^{32} - 1\}$. For given r, it is not hard to experimentally derive the value of α_r by exhaustively evaluating $\mathsf{str}_{32}(\mathsf{int}(Y) + r \bmod 2^{32}) \oplus Y$ for all $Y \in \{0,1\}^{32}$. For example, we have

$$\alpha_0 = 1, \alpha_1 = 32, \alpha_2 = 31, \alpha_3 = 61, \alpha_4 = 30, \alpha_5 = 89, \ldots$$

and the problem is to identify $\alpha_{\max}$.

6.2 Relation to the Security of GCM

We show that identifying α_r gives the upper bound on $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)]$.

Lemma 2. *For any $0 \leq r \leq 2^{32}-1$, N_1, and N_2 such that $N_1 \neq N_2$, $|N_1|, |N_2| \neq 96$, and $|N_1|_n, |N_2|_n \leq \ell_N$, $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] \leq \alpha_r(\ell_N + 1)/2^n$.*

Proof. Let $Y_1, \ldots, Y_{\alpha_r} \in \{0,1\}^{32}$ be the α_r elements of $\mathbb{Y}_r$. Let $\mathcal{Y}_1, \ldots, \mathcal{Y}_{\alpha_r} \subseteq \{0,1\}^{32}$ be the α_r disjoint subsets of $\{0,1\}^{32}$ such that

$$\mathcal{Y}_j = \left\{Y \in \{0,1\}^{32} \mid \mathsf{str}_{32}(\mathsf{int}(Y) + r \bmod 2^{32}) \oplus Y = Y_j\right\}$$

for $1 \leq j \leq \alpha_r$, $\mathcal{Y}_1 \cup \cdots \cup \mathcal{Y}_{\alpha_r} = \{0,1\}^{32}$, and $\mathcal{Y}_j \cap \mathcal{Y}_{j'} = \emptyset$ for $1 \leq j < j' \leq \alpha_r$. Observe that if $Y \in \mathcal{Y}_j$, then $\mathsf{str}_{32}(\mathsf{int}(Y) + r \bmod 2^{32})$ can be replaced with $Y \oplus Y_j$.

For $1 \leq j \leq \alpha_r$, let D_j be the event $\mathsf{Coll}_L(r, N_1, N_2) \wedge \mathsf{lsb}_{32}(\mathsf{GHASH}_L(\varepsilon, N_1)) \in \mathcal{Y}_j$. Since $\mathsf{D}_1, \ldots, \mathsf{D}_{\alpha_r}$ are disjoint events, we have

$$\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] = \sum_{1 \leq j \leq \alpha_r} \Pr_L[\mathsf{D}_j]. \tag{10}$$

Recall that $\mathsf{Coll}_L(r, N_1, N_2)$ is the event $\mathsf{inc}^r(\mathsf{GHASH}_L(\varepsilon, N_1)) \oplus \mathsf{GHASH}_L(\varepsilon, N_2) = 0^n$, and since $\mathsf{lsb}_{32}(\mathsf{GHASH}_L(\varepsilon, N_1)) \in \mathcal{Y}_j$, $\mathsf{inc}^r(\mathsf{GHASH}_L(\varepsilon, N_1))$ can be replaced with $\mathsf{GHASH}_L(\varepsilon, N_1) \oplus (0^{n-32} \,\|\, Y_j)$, implying that the event D_j is equivalent to

$$\mathsf{GHASH}_L(\varepsilon, N_1) \oplus \mathsf{GHASH}_L(\varepsilon, N_2) \oplus (0^{n-32} \,\|\, Y_j) = 0^n \tag{11}$$

and $\mathsf{lsb}_{32}(\mathsf{GHASH}_L(\varepsilon, N_1)) \in \mathcal{Y}_j$. We see that (11) is a non-trivial equation in L of degree at most $\ell_N + 1$ over $\mathrm{GF}(2^n)$, and hence it has at most $\ell_N + 1$ solutions. From (10), we obtain the lemma. □

6.3 Deriving α_r and $\alpha_{\max}$

The problem introduced in Sect. 6.1 can be solved by exhaustively evaluating (9) for all $0 \le r \le 2^{32} - 1$, which is computationally costly. Another possible approach is to follow the framework in [29] (or to use tools in [22]).

Instead of following these approaches, in this section, we present a recursive formula to efficiently compute α_r. Let $r \ge 0$ be a given integer, ℓ be the number of runs of ones in the binary representation of r (e.g., if r is 00110101 in binary, then $\ell = 3$), and v_ℓ be an integer with $r \le 2^{v_\ell} - 1$. Suppose $\mathsf{str}_{v_\ell}(r) = 0^{s_\ell} 1^{t_\ell} \dots 0^{s_1} 1^{t_1} 0^{s_0}$, where $s_{\ell-1}, \dots, s_1 \ge 1$, $s_\ell, s_0 \ge 0$, $t_\ell, \dots, t_1 \ge 1$, and $v_\ell = (s_\ell + \dots + s_1 + s_0) + (t_\ell + \dots + t_1)$.

Define

$$A_\ell \stackrel{\text{def}}{=} \#\left\{\mathsf{str}_{v_\ell}(\mathsf{int}(Y) + r \bmod 2^{v_\ell}) \oplus Y \mid Y \in \{0,1\}^{v_\ell}\right\}.$$

Note that, when $v_\ell = 32$, α_r is equal to A_ℓ. The next proposition gives an efficient recursive formula to compute A_ℓ.

Proposition 1. *For any $\ell \ge 1$,*

$$A_\ell = \begin{cases} t_\ell A_{\ell-1} + B_{\ell-1} & \text{if } s_\ell = 0, \\ s_\ell B_\ell + A_{\ell-1} & \text{if } s_\ell \ge 1, \end{cases} \tag{12}$$

where $B_j = t_j A_{j-1} + B_{j-1}$ for $1 \le j \le \ell$, $A_j = s_j B_j + A_{j-1}$ for $1 \le j \le \ell - 1$, $A_0 = 1$, and $B_0 = 0$.

An elementary proof of the proposition is given in the full version of this paper. Given Proposition 1, it is not hard to experimentally derive $\alpha_{\max}$ by directly evaluating (12). We have $\alpha_{\max} = 3524578$, where $\alpha_r = \alpha_{\max}$ holds when $r =$ `0x2aaaaaab`, `0xaaaaaaab`, `0x55555555`, and `0xd5555555`.

From Lemma 2 and since $3524578 \le 2^{22}$, we obtain the following corollary.

Corollary 1. *For any $0 \le r \le 2^{32} - 1$, N_1, and N_2 such that $N_1 \ne N_2$, $|N_1|, |N_2| \ne 96$, and $|N_1|_n, |N_2|_n \le \ell_N$, $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] \le 2^{22}(\ell_N + 1)/2^n$.*

If the upper bound of r is smaller than $2^{32} - 1$, then depending on the value, the constant, 2^{22}, in Corollary 1 can be replaced by a smaller constant. Specifically, there are 303 values of $1 \le r \le 2^{32} - 1$ that satisfy $\max\{\alpha_0, \dots, \alpha_{r-1}\} < \alpha_r$, and

Table 1. List of (r, α_r) (left) and the relation between $r_{\max}$ and $\beta(r_{\max})$ (right)

r	α_r
0x00000001	32
0x00000003	61
0x00000005	89
0x0000000b	143
0x00000025	294
0x00000055	538
0x0000012b	1115
0x00000455	2113
0x00000955	4124
0x000024ab	8579
0x00005555	17389
0x00012aab	34702
0x00049555	69742
0x000aaaab	138117
0x00255555	262471
0x00955555	559000
0x02555555	1127959
0x0a555555	2116814

Range of $r_{\max}$	$\beta(r_{\max})$
0x00000001–0x00000002	2^5
0x00000003–0x00000004	2^6
0x00000005–0x0000000a	2^7
0x0000000b–0x00000024	2^8
0x00000025–0x00000054	2^9
0x00000055–0x0000012a	2^{10}
0x0000012b–0x00000454	2^{11}
0x00000455–0x00000954	2^{12}
0x00000955–0x000024aa	2^{13}
0x000024ab–0x00005554	2^{14}
0x00005555–0x00012aaa	2^{15}
0x00012aab–0x00049554	2^{16}
0x00049555–0x000aaaaa	2^{17}
0x000aaaab–0x00255554	2^{18}
0x00255555–0x00955554	2^{19}
0x00955555–0x02555554	2^{20}
0x02555555–0x0a555554	2^{21}
0x0a555555–0xffffffff	2^{22}

a list of the 303 values of (r, α_r) can be used to obtain a smaller constant. The list can be found in the full version of this paper. Table 1 (left) is the excerpt from the list for better readability and usability. For each $i = 5, 6, \ldots, 22$, Table 1 (left) lists the minimum value of r, among the 303 values, such that $2^{i-1} < \alpha_r \leq 2^i$.

Table 1 (right) is obtained from Table 1 (left) and shows the relation between the range of $r_{\max}$ and the corresponding constant $\beta(r_{\max})$, where $r_{\max}$ is the upper bound of r and $\beta(r_{\max})$ is the upper bound of α_r. For example, if $r_{\max} = 2^{10} =$ 0x400, then it is within the range of 0x0000012b–0x00000454 and hence $\beta(r_{\max}) = 2^{11}$. See also Fig. 4 for a graph showing the relation between $r_{\max}$ and $\beta(r_{\max})$.

We have the following corollary.

Corollary 2. *For any* $0 \leq r \leq r_{\max}$, N_1, *and* N_2 *such that* $N_1 \neq N_2$, $|N_1|, |N_2| \neq 96$, *and* $|N_1|_n, |N_2|_n \leq \ell_N$, $\Pr_L[\mathsf{Coll}_L(r, N_1, N_2)] \leq \beta(r_{\max})(\ell_N + 1)/2^n$.

We note that for $r_{\max} = 0$, from $\alpha_0 = 1$, $\beta(r_{\max})$ is defined to be 1.

7 Repairing GCM Security Proofs

In this section, basing on the results of the previous section, we present new security bounds on GCM. We also present overviews of the proofs.

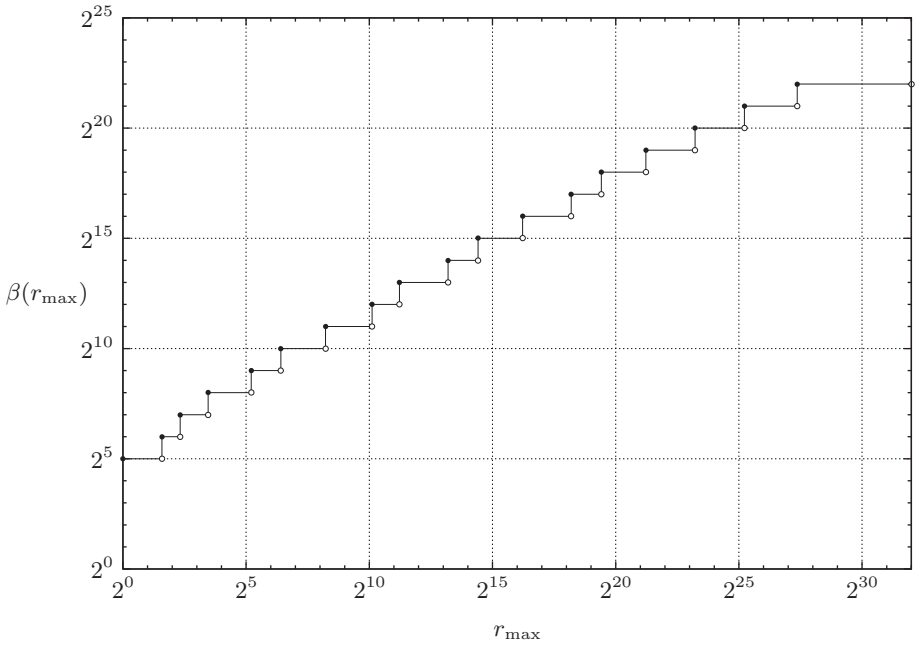

Fig. 4. Relation between $r_{\max}$ and $\beta(r_{\max})$

7.1 Privacy Result

In the privacy result, if $\mathcal{A}$ makes q queries $(N_1, A_1, M_1), \ldots, (N_q, A_q, M_q)$, then the total plaintext length is $m_1 + \cdots + m_q$, and the maximum nonce length is $\max\{n_1, \ldots, n_q\}$, where $|N_i|_n = n_i$ and $|M_i|_n = m_i$. We have the following information theoretic result.

Theorem 1. *Let* $\mathrm{Perm}(n)$ *and* τ *be the parameters of GCM. Then for any* $\mathcal{A}$ *that makes at most* q *queries, where the total plaintext length is at most* σ *blocks and the maximum nonce length is at most* ℓ_N *blocks,*

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{GCM}[\mathrm{Perm}(n),\tau]}(\mathcal{A}) \leq \frac{0.5(\sigma + q + 1)^2}{2^n} + \frac{2^{22} q(\sigma + q)(\ell_N + 1)}{2^n}.$$

Observe that the security bound is essentially the same as the original one (1), except that we have a constant, 2^{22}, in the second term, and we do not have a term that corresponds to the forgery probability.

We next present a corollary showing that GCM has a better security bound if the nonce length is restricted to 96 bits.

Corollary 3. *Assume that the nonce length is restricted to 96 bits. Then,*

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{GCM}[\mathrm{Perm}(n),\tau]}(\mathcal{A}) \leq \frac{0.5(\sigma + q + 1)^2}{2^n}.$$

Let E be a blockcipher secure in the sense of the PRP notion. Then the corresponding complexity theoretic results, where E is used in place of $\mathrm{Perm}(n)$, can be obtained by a standard argument (see e.g. [9]).

In the next section, we present an intuitive proof overview of Theorem 1. A complete proof is presented in the full version of this paper. A proof of Corollary 3 is obtained by modifying the proof of Theorem 1, and is omitted.

7.2 Proof Overview of Theorem 1

Suppose that $\mathcal{A}$ has access to the GCM encryption oracle. We first replace the random permutation P by a random function F. The difference between the two advantage functions is at most $(\sigma+q+1)^2/2^{n+1}$ from the PRP/PRF switching lemma [11]. Next, suppose that $\mathcal{A}$ has made $i-1$ queries $(N_1, A_1, M_1), \ldots, (N_{i-1}, A_{i-1}, M_{i-1})$, obtained $(C_1, T_1), \ldots, (C_{i-1}, T_{i-1})$, and is now making the i-th query (N_i, A_i, M_i). For $1 \leq j \leq i$, let $\mathcal{I}_j = \{I_j[0], I_j[1], \ldots, I_j[m_j]\}$ be a set of n-bit strings used as the inputs of F during the computation of (C_j, T_j) for the query (N_j, A_j, M_j) (other than 0^n used to generate L). Observe that, if $I_i[0], I_i[1], \ldots, I_i[m_i]$ are not previously used, then (C_i, T_i) is a random string of $|M_i|+\tau$ bits. That is, unless

$$\mathcal{I}_i \cap (\{0^n\} \cup \mathcal{I}_1 \cup \cdots \cup \mathcal{I}_{i-1}) \neq \emptyset \tag{13}$$

holds for some $1 \leq i \leq q$, the distribution of the output of the GCM encryption oracle (based on the random function F) is identical to that of the random-bits oracle, and hence $\mathcal{A}$ is unable to distinguish between the two oracles. It can be shown that the probability of (13) is at most $2^{22}q(\sigma+q)(\ell_N+1)/2^n$ by using Corollary 1. The result is obtained by summing up the two above-mentioned probabilities.

7.3 Authenticity Result

If $\mathcal{A}$ makes q encryption queries $(N_1, A_1, M_1), \ldots, (N_q, A_q, M_q)$ and q' decryption queries $(N'_1, A'_1, C'_1, T'_1), \ldots, (N'_{q'}, A'_{q'}, C'_{q'}, T'_{q'})$, then the total plaintext length is $m_1+\cdots+m_q$, the maximum nonce length is $\max\{n_1, \ldots, n_q, n'_1, \ldots, n'_{q'}\}$, and the maximum input length is $\max\{a_1+m_1, \ldots, a_q+m_q, a'_1+m'_1, \ldots, a'_{q'}+m'_{q'}\}$, where $|N_i|_n = n_i$, $|A_i|_n = a_i$, $|M_i|_n = m_i$, $|N'_i|_n = n'_i$, $|A'_i|_n = a'_i$, and $|C'_i|_n = m'_i$. We have the following information theoretic result.

Theorem 2. *Let* $\mathrm{Perm}(n)$ *and* τ *be the parameters of GCM. Then for any* $\mathcal{A}$ *that makes at most* q *encryption queries and* q' *decryption queries, where the total plaintext length is at most* σ *blocks, the maximum nonce length is at most* ℓ_N *blocks, and the maximum input length is at most* ℓ_A *blocks,*

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{GCM}[\mathrm{Perm}(n),\tau]}(\mathcal{A}) \leq \frac{0.5(\sigma+q+q'+1)^2}{2^n} + \frac{2^{22}(q+q'+1)(\sigma+q)(\ell_N+1)}{2^n} + \frac{q'(\ell_A+1)}{2^\tau}. \tag{14}$$

As in the privacy result, the bound is essentially the same as the original one (2), except that we have a constant, 2^{22}, in the second term. The next corollary shows that we have a better security bound if the nonce length is restricted to 96 bits.

Corollary 4. *Assume that the nonce length is restricted to 96 bits. Then,*

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{GCM}[\mathrm{Perm}(n),\tau]}(\mathcal{A}) \le \frac{0.5(\sigma+q+q'+1)^2}{2^n} + \frac{q'(\ell_A+1)}{2^\tau}.$$

The corresponding complexity theoretic results can be obtained based on the PRP notion of a blockcipher E by a standard argument (see e.g. [9]).

We present an intuitive proof overview of Theorem 2 in the next section, and a complete proof is presented in the full version of this paper. A proof of Corollary 4 can be obtained from the proof of Theorem 2, and is omitted.

7.4 Proof Overview of Theorem 2

We replace the random permutation P by a random function F. From the PRP/PRF switching lemma [11], we have a term $(\sigma+q+q'+1)^2/2^{n+1}$. We then consider the probability of a counter collision as in the privacy proof, but this time, we consider the counter values used for decryption queries as well. The probability can be shown to be at most $2^{22}(q+q'+1)(\sigma+q)(\ell_N+1)/2^n$ by using Corollary 1. Under the condition that there is no counter collision, the adversary is essentially asked to forge a message authentication code $(N,A,C) \to F(g(N)) \oplus \mathsf{GHASH}_L(A,C)$, where $g(N) = N \,\|\, 0^{31}1$ if $|N| = 96$, and $g(N) = \mathsf{GHASH}_L(\varepsilon, N)$ if $|N| \neq 96$. The probability can be shown to be at most $q'(\ell_A+1)/2^\tau$, and we obtain the theorem by summing up the three probabilities.

7.5 Better Security Bounds

Use of Corollary 2. Suppose that, either in privacy or authenticity notions, $\mathcal{A}$ makes q encryption queries $(N_1,A_1,M_1),\ldots,(N_q,A_q,M_q)$. Let ℓ_M be the maximum plaintext length, which is $\max\{m_1,\ldots,m_q\}$, where $|M_i|_n = m_i$. Theorem 1 and Theorem 2 assume $\ell_M = 2^{32}-2$ and use Corollary 1 to obtain the results. However, if ℓ_M is known to be smaller, then Corollary 2 can be used to obtain better bounds. Specifically, in Theorem 1 and Theorem 2, if $\ell_M \le r_{\max}$, then the constant becomes $\beta(r_{\max})$ instead of 2^{22}. For example, if ℓ_M is 2^{10}, then from Table 1 and by following the argument in Sect. 6.3, the constant becomes 2^{11}.

Use of [12, Theorem 2.3]. Using Bernstein's result [12, Theorem 2.3], we can further improve the authenticity bound (but not the privacy bound). For positive integer a, let $\delta_n(a) = (1-(a-1)/2^n)^{-a/2}$. With the same notation as in Theorem 2, the right hand side of (14) can be improved to the following bound.

$$\left[\frac{2^{22}(q+q'+1)(\sigma+q)(\ell_N+1)}{2^n} + \frac{q'(\ell_A+1)}{2^\tau}\right] \cdot \delta_n(\sigma+q+q'+1)$$

Note that when $a \ll 2^n$ we have $\delta_n(a) \approx (1+a^2/2^{n+1})$. It was shown that $\delta_n(a) \le 1.7$ when $a \le 2^{64}$ and $n \ge 128$ [13]. Hence, for example, if $1 < q' \le q \le \sigma$, $n = \tau = 128$, and $\sigma+q+q' < 2^{64}$, we obtain the bound $(17{\cdot}2^{22}q\sigma\ell_N + 4q'\ell_A)/2^{128}$.

8 Conclusion

In this paper, we studied the security proofs of GCM. We first pointed out that the proofs contain a flaw, and translated the observation into a distinguishing attack that invalidates the main part of the privacy proof. We then showed that the proofs can be repaired by presenting new privacy and authenticity bounds. The bounds are larger than what were previously claimed in a general case of variable length nonces, but they are smaller when the nonce length is restricted to 96 bits. Many standards require or recommend using GCM with 96-bit nonces for efficiency reasons. Our results suggest that restricting GCM to 96-bit nonces is recommended from the provable security perspective as well. This follows [31], where GCM with 96-bit nonces is recommended as the use of variable length nonces increases the proof complexity and the proofs are infrequently verified.

We remark that since our attack only invalidates the second terms of (1) and (2), it does not exclude a possibility that the original security bounds, (1) and (2), can still be proved, and it would be interesting to see if our security bounds can be improved.

Acknowledgments. The authors would like to thank Masato Aikawa for discussions at the initial phase of this work, Yasushi Osaki and Xiangyu Quan for helping searching the counter-examples given in Sect. 5.1 and in Appendix A, Nicky Mouha for verifying the values of r that give $\alpha_r = \alpha_{\max}$ presented in Sect. 6.3, and participants of Dagstuhl Seminar 12031, Symmetric Cryptography, and the anonymous CRYPTO 2012 reviewers for helpful comments. The work by Tetsu Iwata was supported in part by MEXT KAKENHI, Grant-in-Aid for Young Scientists (A), 22680001.

References

1. Bouncy Castle, `http://www.bouncycastle.org/` (accessed on May 26, 2012)
2. Java Platform, Standard Edition 7, `http://docs.oracle.com/javase/7/docs/` (accessed on May 26, 2012)
3. Risa/Asir, `http://www.math.kobe-u.ac.jp/Asir/asir.html` (accessed on May 26, 2012)
4. PKCS #11 v2.20: Cryptographic Token Interface Standard. PKCS #11 v2.20 (2004), `http://www.rsa.com/rsalabs/node.asp?id=2133` (accessed on May 31, 2012)
5. IEEE Standard for Local and Metropolitan Area Networks Media Access Control (MAC) Security. IEEE Std 802.1AE-2006 (2006)
6. IEEE Standard for Authenticated Encryption with Length Expansion for Storage Devices. IEEE Std 1619.1-2007 (2007)
7. Information Technology — Security Techniques — Authenticated Encryption, ISO/IEC 19772:2009. International Standard ISO/IEC 19772 (2009)
8. National Security Agency, Internet Protocol Security (IPsec) Minimum Essential Interoperability Requirements, IPMEIR Version 1.0.0 Core (2010), `http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml`

9. Bellare, M., Kilian, J., Rogaway, P.: The Security of the Cipher Block Chaining Message Authentication Code. J. Comput. Syst. Sci. 61(3), 362–399 (2000)
10. Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In: Okamoto, T. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 1976, pp. 531–545. Springer (2000)
11. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 4004, pp. 409–426. Springer (2006)
12. Bernstein, D.J.: Stronger Security Bounds for Permutations (2005), http://cr.yp.to/papers.html (accessed on May 31, 2012)
13. Black, J., Halevi, S., Krawczyk, H., Krovetz, T., Rogaway, P.: UMAC Security Bound from PRP-Advantage (2005), http://fastcrypto.org/umac/umac_security.pdf (accessed on May 31, 2012)
14. Dai, W.: Crypto++ Library, http://www.cryptopp.com/ (accessed on May 26, 2012)
15. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D (2007)
16. Ferguson, N.: Authentication Weaknesses in GCM. Public Comments to NIST (2005), http://csrc.nist.gov/groups/ST/toolkit/BCM/comments.html
17. Gladman, B.: http://www.gladman.me.uk/ (accessed on May 26, 2012)
18. Handschuh, H., Preneel, B.: Key-Recovery Attacks on Universal Hash Function Based MAC Algorithms. In: Wagner, D. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 5157, pp. 144–161. Springer (2008)
19. Housley, R.: Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS). IETF RFC 5084 (2007)
20. Igoe, K.M., Solinas, J.A.: AES Galois Counter Mode for the Secure Shell Transport Layer Protocol. IETF RFC 5647 (2009)
21. Joux, A.: Authentication Failures in NIST version of GCM. Public Comments to NIST (2006), http://csrc.nist.gov/groups/ST/toolkit/BCM/comments.html
22. Leurent, G.: ARXtools: A Toolkit for ARX Analysis. The Third SHA-3 Candidate Conference (2012), http://csrc.nist.gov/groups/ST/hash/sha-3/Round3/March2012/index.html
23. Leurent, G., Thomsen, S.S.: Practical Near-Collisions on the Compression Function of BMW. In: Joux, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 6733, pp. 238–251. Springer (2011)
24. Luby, M., Rackoff, C.: How to Construct Pseudorandom Permutations from Pseudorandom Functions. SIAM J. Comput. 17(2), 373–386 (1988)
25. McGrew, D.A.: An Interface and Algorithms for Authenticated Encryption. IETF RFC 5116 (2008)
26. McGrew, D.A., Viega, J.: The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT. Lecture Notes in Computer Science, vol. 3348, pp. 343–355. Springer (2004)
27. McGrew, D.A., Viega, J.: The Security and Performance of the Galois/Counter Mode of Operation (Full Version). Cryptology ePrint Archive, Report 2004/193 (2004), http://eprint.iacr.org/
28. McGrew, D.A., Viega, J.: The Galois/Counter Mode of Operation (GCM). Submission to NIST (2005), http://csrc.nist.gov/groups/ST/toolkit/BCM/modes_development.html

Table 2. List of solutions of (3)

0x7f6db6d2db6db6db6db6db6492492492	0x7f6db6dadb6db6db6db6db6492492492
0x81b6db776db6db6db6db6dadb6db6db6	0x81b6db676db6db6db6db6dadb6db6db6
0xbe00003c000000000000003ffffffff	0xbe00001c000000000000003ffffffff
0xc16db6aadb6db6db6db6db1b6db6db6d	0xc16db6eadb6db6db6db6db1b6db6db6d
0x3fb6db876db6db6db6db6d5249249249	0x3fb6db076db6db6db6db6d5249249249
0x000001dc00000000000001c000000000	0x000000dc00000000000001c000000000
0x7f6db56adb6db6db6db6d8e492492492	0x7f6db76adb6db6db6db6d8e492492492
0x81b6dc076db6db6db6db6aadb6db6db6	0x81b6d8076db6db6db6db6aadb6db6db6
0xbe000edc0000000000000e3ffffffff	0xbe0006dc0000000000000e3ffffffff
0xc16dab6adb6db6db6db6c71b6db6db6d	0xc16dbb6adb6db6db6db6c71b6db6db6d
0x3fb6e0076db6db6db6db555249249249	0x3fb6c0076db6db6db6db555249249249
0x000076dc00000000000071c000000000	0x000036dc00000000000071c000000000
0x7f6d5b6adb6db6db6db638e492492492	0x7f6ddb6adb6db6db6db638e492492492
0x81b700076db6db6db6daaaadb6db6db6	0x81b600076db6db6db6daaaadb6db6db6
0xbe03b6dc0000000000038e3ffffffff	0xbe01b6dc0000000000038e3ffffffff
0xc16adb6adb6db6db6db1c71b6db6db6d	0x00000004000000000000000000000000

29. Mouha, N., Velichkov, V., Cannière, C.D., Preneel, B.: The Differential Analysis of S-Functions. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 6544, pp. 36–56. Springer (2010)
30. Rogaway, P.: Authenticated-Encryption with Associated-Data. In: Atluri, V. (ed.) ACM Conference on Computer and Communications Security. pp. 98–107. ACM (2002)
31. Rogaway, P.: Evaluation of Some Blockcipher Modes of Operation. Investigation Reports on Cryptographic Techniques in FY 2010 (2011), `http://www.cryptrec.go.jp/english/` (accessed on May 31, 2012)
32. Saarinen, M.J.O.: Cycling Attacks on GCM, GHASH and Other Polynomial MACs and Hashes. Pre-proceedings of FSE 2012 (2012)
33. Salowey, J., Choudhury, A., McGrew, D.A.: AES Galois Counter Mode (GCM) Cipher Suites for TLS. IETF RFC 5288 (2008)
34. Viega, J., McGrew, D.A.: The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP). IETF RFC 4106 (2005)

A Solutions of (3), (4), and (5), and Examples of (N_1, N_2) Satisfying (6)

In Table 2, Table 3, and Table 4, we show a list of values of L that satisfy (3), (4), and (5), respectively. We see that the 93 values from these lists are all distinct, and are different from 0^n.

The counter-example presented in Sect. 5.2 was found by experimentally searching over the values of U_1, U_2, and V. We started by searching over random U_1, U_2, and V, and found that the values of the form $U_1 = 0^{8i} \| X \| 0^{n-8-8i}$, $U_2 = 0^{8i} \| Y \| 0^{n-8-8i}$, and $V \in \{0,1\}^n$ have many examples that satisfy (6), where $X, Y \in \{0,1\}^8$, $0 \le i \le 15$, and $\mathsf{int}(V) = 8j$ for some $i+1 \le j \le 16$ but

Table 3. List of solutions of (4)

0x7f6db6d6db6db6db6db6db6492492492	0x7f6db6dedb6db6db6db6db6492492492
0x81b6db736db6db6db6db6dadb6db6db6	0x81b6db636db6db6db6db6dadb6db6db6
0xbe0000380000000000000003ffffffff	0xbe0000180000000000000003ffffffff
0xc16db6aedb6db6db6db6db1b6db6db6d	0xc16db6eedb6db6db6db6db1b6db6db6d
0x3fb6db836db6db6db6db6d5249249249	0x3fb6db036db6db6db6db6d5249249249
0x000001d800000000000001c000000000	0x000000d800000000000001c000000000
0x7f6db56edb6db6db6db6d8e492492492	0x7f6db76edb6db6db6db6d8e492492492
0x81b6dc036db6db6db6db6aadb6db6db6	0x81b6d8036db6db6db6db6aadb6db6db6
0xbe000ed80000000000000e3ffffffff	0xbe0006d80000000000000e3ffffffff
0xc16dab6edb6db6db6db6c71b6db6db6d	0xc16dbb6edb6db6db6db6c71b6db6db6d
0x3fb6e0036db6db6db6db555249249249	0x3fb6c0036db6db6db6db555249249249
0x000076d800000000000071c000000000	0x000036d800000000000071c000000000
0x7f6d5b6edb6db6db6db638e492492492	0x7f6ddb6edb6db6db6db638e492492492
0x81b700036db6db6db6daaaadb6db6db6	0x81b600036db6db6db6daaaadb6db6db6
0xbe03b6d80000000000038e3ffffffff	0xbe01b6d80000000000038e3ffffffff
0xc16adb6edb6db6db6db1c71b6db6db6d	

Table 4. List of solutions of (5)

0xbe076db80000000000071c7ffffffff	0xc16c000edb6db6db6db5555b6db6db6d
0xc16e000edb6db6db6db5555b6db6db6d	0xfedbb6d5b6db6db6db6c71c924924924
0xfedab6d5b6db6db6db6c71c924924924	0x00006db8000000000000e38000000000
0x0000edb8000000000000e38000000000	0x7f6d800edb6db6db6db6aaa492492492
0x7f6dc00edb6db6db6db6aaa492492492	0x40db76d5b6db6db6db6d8e36db6db6db
0x40db56d5b6db6db6db6d8e36db6db6db	0xbe000db80000000000001c7ffffffff
0xbe001db80000000000001c7ffffffff	0xc16db00edb6db6db6db6d55b6db6db6d
0xc16db80edb6db6db6db6d55b6db6db6d	0xfedb6ed5b6db6db6db6db1c924924924
0xfedb6ad5b6db6db6db6db1c924924924	0x000001b8000000000000038000000000
0x000003b8000000000000038000000000	0x7f6db60edb6db6db6db6daa492492492
0x7f6db70edb6db6db6db6daa492492492	0x40db6dd5b6db6db6db6db636db6db6db
0x40db6d55b6db6db6db6db636db6db6db	0xbe0000380000000000000007ffffffff
0xbe0000780000000000000007ffffffff	0xc16db6cedb6db6db6db6db5b6db6db6d
0xc16db6eedb6db6db6db6db5b6db6db6d	0xfedb6db5b6db6db6db6db6c924924924
0xfedb6da5b6db6db6db6db6c924924924	0x000000080000000000000000000000000

$j \neq 12$. The examples include the following values:

$$(U_1, U_2, V) = \begin{cases} ((\mathtt{0x0})^{15} \,\|\, \mathtt{0x2} \,\|\, (\mathtt{0x0})^{16}, (\mathtt{0x0})^{15} \,\|\, \mathtt{0x6} \,\|\, (\mathtt{0x0})^{16}, (\mathtt{0x0})^{30} \,\|\, \mathtt{0x70}) \\ ((\mathtt{0x0})^{17} \,\|\, \mathtt{0x2} \,\|\, (\mathtt{0x0})^{14}, (\mathtt{0x0})^{17} \,\|\, \mathtt{0x6} \,\|\, (\mathtt{0x0})^{14}, (\mathtt{0x0})^{30} \,\|\, \mathtt{0x70}) \\ ((\mathtt{0x0})^{17} \,\|\, \mathtt{0x4} \,\|\, (\mathtt{0x0})^{14}, (\mathtt{0x0})^{17} \,\|\, \mathtt{0xc} \,\|\, (\mathtt{0x0})^{14}, (\mathtt{0x0})^{30} \,\|\, \mathtt{0x48}) \end{cases}$$

These values are equivalent to $(0^{60} \,\|\, 0010 \,\|\, 0^{64}, 0^{60} \,\|\, 0110 \,\|\, 0^{64}, 0^{120} \,\|\, 01110000)$, $(0^{68} \,\|\, 0010 \,\|\, 0^{56}, 0^{68} \,\|\, 0110 \,\|\, 0^{56}, 0^{120} \,\|\, 01110000)$, and $(0^{68} \,\|\, 0100 \,\|\, 0^{56}, 0^{68} \,\|\, 1100 \,\|\, 0^{56}, 0^{120} \,\|\, 01001000)$ in binary. N_1 and N_2 are the most significant $\mathsf{int}(V)$ bits of U_1 and U_2, respectively.