Authentication Key Recovery in Galois/Counter Mode (GCM)

John Mattsson

Ericsson Research, Stockholm, Sweden
{firstname.lastname}@ericsson.com

Abstract. GCM is used in a vast amount of security protocols and is quickly becoming the de facto mode of operation for block ciphers. In this paper we suggest several novel improvements to Fergusons's authentication key recovery method and show that for many truncated tag sizes, the security levels are far below, not only the current NIST requirement of 112-bit security, but also the old NIST requirement of 80-bit security. We therefore strongly recommend NIST to make a revision of SP 800-38D.

Keywords: Secret-key Cryptography, Message Authentication Codes, Block Ciphers, Cryptanalysis, Galois/Counter Mode, GCM, Authentication Key Recovery, AES-GCM, Suite B

1 Introduction

GCM [3] is quickly becoming the de facto mode of operation for block ciphers. GCM is included in the NSA Suite B set of cryptographic algorithms [12], and AES-GCM is the benchmark algorithm for the AEAD competition CAESAR [13]. Together with GMAC, GCM is used in a vast amount of security protocols:

- Many protocols such as IPsec [14], TLS [15], SSH [16], JOSE [17], 802.1AE (MACsec) [18], 802.11ad (WiGig) [19], 802.11ac (Wi-Fi) [20], P1619.1 (data storage) [21], and Fibre Channel [22] only allow 128-bit tags.
- The exceptions are IPsec [23] that allows 64, 96, and 128 bit tags, CMS [24] that allows 96, 104, 112, 120, and 128 bit tags, NFC-SEC [25][26] that only allows 96 bit tags, SRTP [27] that allows 96 and 128 bit tags, and SRTP [28] that allows 64 and 128 bit tags.

GCM is also used in several cryptography APIs. W3C Web Cryptography API [29] and Oracle Java SE [30] allow 32, 64, 96, 104, 112, 120, and 128 bit tags. PKCS #11 [31] allows tags of any length between 0 and 128 bits, and for Microsoft Cryptography API [32] we could not find any information on allowed tag lengths.

The popularity is very well deserved, GCM has exceptional performance and proven security, it is on-line and fully parallelizable, and it is efficient in both hardware and software, especially on new processors with dedicated AES-GCM instructions. Weaknesses of the GCM decryption function were described by

Ferguson [5], which showed that the forgery probability is not 2^{-t}, and that feedback on successful forgeries allows an attacker to recover the authentication key H. As a note, the fact that the substitution probability decreases as message length increases was already known [33]. The complexity of Ferguson's authentication key recovery method for the NIST approved tag lengths have not previously been analyzed.

In Sect. 2.1 we suggest an improvement to Fergusons's method for message forgery and authentication key recovery method [5]. We then use this improved method to calculate the probabilities for multiple message forgeries. In Sect. 2.2 we use these probabilities to calculate the complexity for authentication key recovery using Ferguson's method. In Sect. 2.3 we suggest several novel improvements to Fergusons's method that significantly reduces the security levels for short tags. We show that independently of the encryption key size, the security levels (i.e. the effective key lengths) are 62–67 bits for 32-bit tags, 70–75 bits for 64-bit tags, and t bits for t-bit tags where t = 96, 104, 112, 120, or 128. For many tag sizes, the security levels are far below, not only the current NIST requirement of 112-bit security, but also the old NIST requirement of 80-bit security. The results are applicable to both GCM and GMAC.

We strongly recommend NIST to make a revision of [3] so that the security levels of all allowed options are clearly stated, that options offering less than 112-bit security are removed from the standard, or, if they are allowed, explain why a security level below 112 bits is acceptable.

We do however fully recommend GCM for usage with 128-bit tags, especially with AES-128. In fact we believe that with its excellent performance and proven security, it should be the first choice for everybody wanting an AEAD algorithm.

2 Galois/Counter Mode (GCM)

Galois/Counter Mode (GCM) is an Authenticated Encryption with Associated Data (AEAD) mode of operation for block ciphers with a block size of 128 bits. It was designed by McGrew and Viega [1][2] and is standardized in NIST SP 800-38D [3] and ISO/IEC 19772:2009 [4]. This analysis is based on [3]. GCM combines the well-known counter mode of encryption with the Galois mode of authentication, which is based on universal hashing. The Galois mode of authentication makes use of the function $\mathrm{GHASH}_H(A, C)$, which uses multiplications in $\mathrm{GF}(2^{128})$ that can easily be parallelized. The 128–bit authentication tag is defined as

$$\mathrm{Tag} = E_K(N) \oplus \mathrm{GHASH}_H(A, C) \ , \tag{1}$$

where K is the encryption key, N is the nonce, $H = E_K(0^{128})$ is the authentication key (the encryption of 128 zero bits), A is the associated data (to be authenticated but not encrypted), and C is the ciphertext. The output of the authenticated decryption function is either the plaintext P or the special error code $FAIL$.

SP 800-38D specifies that the 128-bit authentication tag may be truncated to 96, 104, 112, or 120 bits. For tags lengths of at least 96 bits, the maximum combined length of A and C is $L = 2^{57}$ blocks and the maximum number of invocations q of the authenticated decryption function is unlimited. For certain applications the tag may be truncated to 32 or 64 bits, and for these tag lengths, L and q are more restricted. Galois Message Authentication Code (GMAC) is an authentication-only variant of GCM. It can be seen as a special case of GCM where the ciphertext C is the empty string. We refer to [3] for the full specification of GCM and GMAC. Explicit weaknesses of the GCM functions have been discussed by Ferguson [5], Joux [6], Handschuh and Preneel [34], and Saarinen [7]. An extensive evaluation of GCM was done by Rogaway [8].

In Appendix B of SP 800-38D [3], NIST summarizes some particulars of the GCM authentication function that were pointed out by Ferguson [5]:

- For t-bit tags, the forgery probability is not the ideal 2^{-t} but instead $2^l \cdot 2^{-t}$ where 2^l is the length in blocks of the largest message (A and C) processed by the authenticated encryption function.
- Each successful forgery enables the adversary to recover parts of the authentication key H and increases the probability of subsequent forgeries.

Our conclusions from the first bullet is:

- The tag and message lengths must be chosen so that the forgery probability $L \cdot 2^{-t}$ is acceptable. But as the complexity of performing a single forgery is still the expected 2^t, we do not find this overly problematic.

The second bullet is harder to analyze and more problematic. NIST draws the conclusions that for short tags:

- There should not be feedback of whether a forgery attempt is successful or unsuccessful.
- The maximum combined length L of A and C shall be heavily restricted.
- The maximum number of invocations q of the authenticated decryption function shall be restricted.

The requirements are listed in Table 1. Unfortunately, NIST does not give any motivations for the specific restrictions, or for that matter the security levels they were assumed to give.

Table 1. NIST requirements on the usage of GCM with short tags

t	32						64					
L	2^1	2^2	2^3	2^4	2^5	2^6	2^{11}	2^{13}	2^{15}	2^{17}	2^{19}	2^{21}
q	2^{22}	2^{20}	2^{18}	2^{15}	2^{13}	2^{11}	2^{32}	2^{29}	2^{26}	2^{23}	2^{20}	2^{17}

2.1 Effective Tag Length

Fergusson [5] demonstrates through a concrete attack that due to the linear behavior of the GCM authentication function, the forgery probability for t-bit tags is not 2^{-t}. If the length of the largest ciphertext C processed by the authenticated encryption function is 2^l blocks, the forgery probability is 2^{l-t}. For subsequent forgeries, the effective tag length is smaller. We refer to [5] for the full specification of how this forgery is done. As pointed out by McGrew and Viega in [9], this does not break the security guarantees of GCM; it proves that the bounds in [2] are tight. Our view is that a better and more natural measure of forgery resistance is complexity. For an ideal MAC, the data complexity to perform a single forgery is $2^0 \cdot 2^t = 2^t$. For GCM, the data complexity to perform a single forgery is $2^l \cdot 2^{t-l} = 2^t$.

Reading [5], it is not trivial to understand or calculate the effective tag lengths for re-forgeries. In this section we extend Fergusson's method to use associated data in addition to ciphertext. We then suggest an improvement to Ferguson's method, derive a formula for the effective tag lengths, and apply this formula to the NIST approved tag and message lengths. These effective tag lengths are then used in Sect. 2.2 to evaluate the data complexity of Ferguson's method for authentication key recovery.

Extension to use associated data. The attacker tries to change the associated data A and the ciphertext C without changing the tag. The attacker does not change the number of blocks in A and C. Let A' be the modified associated data, let C' be the modified ciphertext, and define B and B' as

$$B = A \parallel 0^{128-v} \parallel C \parallel 0^{128-u} \parallel \text{len}(A) \parallel \text{len}(C) \ , \tag{2}$$

$$B' = A' \parallel 0^{128-v} \parallel C' \parallel 0^{128-u} \parallel \text{len}(A) \parallel \text{len}(C) \ , \tag{3}$$

where v is the bit length of the final block of A and u is the bit length of the final block of C. Let B_i be block i of B, where we number the blocks in the same order as Ferguson, i.e. $B_0 = \text{len}(A) \parallel \text{len}(C)$. We can now define the error polynomial E as in [5]

$$E = \sum_i D_i \cdot H^{2^i} \ , \tag{4}$$

where $D_i = E_{2^i}$ and $E_i = (B_i - B_i')$.

Effective Tag Length. Let t_n be the effective tag length after n successful forgeries (with feedback). Following the procedures in [5] and assuming that:

- The byte length of A or C is not a multiple of 16. This implies that the attacker can modify the length encoding in D_0.
- The combined length of A and C is at least $2^l - 1$ blocks.

With these assumptions, the attacker has $128l$ free variables and can in the first forgery force $e_0 = l$ bits of the error polynomial E to zero. The effective tag length is therefore $t_0 = t - l$. In subsequent forgeries the attacker knows more bits of the authentication key H and can force even more bit of the error polynomial E to zero. Feedback of successful forgery of a message with effective tag length t_n allows recovery of t_n additional bits of the authentication key H. After n successful forgeries, the attacker knows h_n bits of H and can force e_n bits of the error polynomial E to zero where

$$h_n = \sum_{j=1}^{n} t_n \quad \text{and} \quad e_n = \left\lfloor \frac{128l}{128 - h_n} \right\rfloor \ . \tag{5}$$

Following [5], the effective tag length is $t_n = t - e_n$ until the attacker knows all 128 bits of H ($h_n \geq 128$) or when the attacker can force more then t bits of the error polynomial to zero ($e_n \geq t$), in which case the effective tag size is zero.

Our improvement. We notice that when $128 - h_n . < t - e_n$, the effective tag length can be reduced by doing exhaustive search on the $128 - h_n$ unknown bits of H instead of doing exhaustive search on the $t - e_n$ bits of the tag that could not be forced to zero. With this improvement, the effective tag size is

$$t_n = \max\big(0,\ \min\big(t - e_n,\ 128 - h_n\big)\big) \ . \tag{6}$$

This improvement significantly reduces some of the effective tag lengths, but has negligible effect on the authentication key recovery complexities in Table 3 and 4. The result of applying the improved formula (6) to the NIST approved tag and maximum message lengths, as well as the maximum message lengths of 2^{12} and 2^{28} blocks imposed by IPv4 and IPv6 is shown in Table 2.

Table 2. Effective tag lengths for the NIST approved tag and message lengths

t	32						64						96			104			112			120			128		
L	2^5	2^6	2^7	2^8	2^9	2^{10}	2^{15}	2^{17}	2^{19}	2^{21}	2^{23}	2^{25}	2^{16}	2^{32}	2^{61}	2^{16}	2^{32}	2^{61}	2^{16}	2^{32}	2^{61}	2^{16}	2^{32}	2^{61}	2^{16}	2^{32}	2^{61}
t_0	31	30	29	28	27	26	53	51	49	47	45	43	84	68	39	92	76	47	100	84	55	108	92	63	116	100	71
t_1	31	30	29	27	26	25	46	43	40	38	35	33	44	37	15	36	36	14	28	31	13	20	21	8	0	0	0
t_2	31	29	27	25	24	23	16	16	15	14	14	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_3	29	26	24	22	20	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_4	6	13	12	13	12	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_5	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

While the values t_0 might look short, the complexity of performing a single forgery is still the expected 2^t. If a tag length of $t = 128$ is used with an encryption

key of length 128 bits, performing a single forgery is as hard as recovering the encryption key, hardly a weakness.

The effective tag lengths in Table 2 are calculated with the greedy algorithm used by Ferguson. Using the suggestions we propose in Sect. 2.3, it is possible to decrease the effective tag length of later forgeries by increasing the effective tag length of earlier forgeries.

2.2 Complexity of Ferguson's Authentication Key Recovery Method

The discussions [9][10] after Ferguson's paper [5] focused mostly on multiple forgeries and authentication key recovery after nonce collisions in the encryption function, i.e. the forbidden attack later discussed by Joux [6]. We think the most important aspect of Ferguson's paper is the full recovery of the authentication key H after successful forgeries to the decryption function. While we agree with McGrew and Viega that the expected complexity to perform multiple forgeries is unclear, the expected complexity against key recovery is very clear. The complexity of performing full key recovery is expected to be 2^k where k is the stated security level. Unless stated otherwise, k is expected to be equal to the key length. In e.g. HMAC-SHA-256 the complexity for key recovery is believed to be 2^{256}, unless the authentication key is derived from a smaller key. In GCM, the authentication key is always 128 bits, which means that the security level against authentication key recovery is never more than 128 bits, even if block ciphers with larger key sizes like AES-192 or AES-256 are used. Other AEAD schemes like CCM and OCB gives a security level equal to the encryption key size. This shortcoming is not mentioned in [2][3][8].

An important detail mentioned in [6] but not in [3][5] is that as the authentication tag depends on $E_K(N)$, authentication key recovery in GCM does not mean that the attacker can independently create new messages. Knowledge of the authentication key H enables an attacker to modify a valid message by freely choosing A and C, but not N. Assuming known-plaintext, an attacker can freely chose A and P, where P is the plaintext. Still, we would expect a security level of no less than the encryption key length against authentication key recovery attacks.

Complexity without query restrictions. Assuming a maximum combined length of $L = 2^l$ blocks, the effective tag length is $t_0 = t - l$, and the data complexity (measured in blocks) of performing the first forgery is $2^l \cdot 2^{t-l} = 2^t$. As the complexity of Ferguson's authentication key recovery method is dominated by the complexity of the first forgery, this is also the data complexity c for full authentication key recovery

$$c \approx 2^t \ . \tag{7}$$

Hence, without restrictions on q and irrespective of encryption key length, the security level of GCM against full authentication key recovery is only equal to the tag length t. This shortcoming is not mentioned in [3][8].

Complexity with query restrictions. The complexity of Fergusson's key recovery method with restrictions q and L has not previously been analyzed. In this section we derive the complexities for the NIST approved tag and maximum message lengths. Limiting the maximum number of invocations q of the decryption function so that $2^{t_0} \gg q \gg 2^{t_1}$ does not increase the complexity of authentication key recovery. The data complexity is $q \cdot 2^l$ and the probability that the attacker succeeds with a forgery in q attempts is $p_1 \approx q \cdot 2^{-t_0}$, resulting in the same total complexity of $q \cdot 2^l \,/\, p_1 = 2^l \,/\, 2^{l-t} = 2^t$.

Restricting q so that $2^{t_1} \gg q$ does however increase the complexity of Ferguson's method. Let $\phi_i = 2^{-t_i}$. The probability that the first successful forgery will occur on forgery attempt f is approximately $\phi_0(1-\phi_0)^{f-1}$ and the probability of a second forgery is approximately $\phi_1(q-f)$. The probability p_2 that an attacker succeeds with two forgeries in q attempts is therefore:

$$p_2 \approx \sum_{f=1}^{q} \phi_0(1-\phi_0)^{f-1} \cdot \phi_1(q-f) = \frac{\phi_0\phi_1}{2} q^2 + \mathcal{O}\left(\frac{\phi_0^2\phi_1}{6} q^3\right) . \qquad (8)$$

We used SageMath to calculate the Taylor series and then collected the leading terms for the domain $\phi_0, \phi_1 \ll q^{-1}$. McGrew and Viega prove the same formula in [10]. With the above approximation for p_2 we can approximate p_3, and with p_3 we can approximate p_4, etc.

$$p_3 \approx \sum_{f=1}^{q} \phi_0(1-\phi_0)^{f-1} \cdot \frac{\phi_1\phi_2}{2}(q-f)^2 = \frac{\phi_0\phi_1\phi_2}{6} q^3 + \mathcal{O}\left(\frac{\phi_0^2\phi_1\phi_2}{24} q^4\right) ,$$

$$p_4 \approx \sum_{f=1}^{q} \phi_0(1-\phi_0)^{f-1} \cdot \frac{\phi_1\phi_2\phi_3}{6}(q-f)^3 = \frac{q^4}{4!}\prod_{j=0}^{3}\phi_j + \mathcal{O}\left(\frac{\phi_0 q^5}{5!}\prod_{j=0}^{3}\phi_j\right) , \qquad (9)$$

$$p_5 \approx \sum_{f=1}^{q} \phi_0(1-\phi_0)^{f-1} \cdot \frac{\phi_1\phi_2\phi_3\phi_4}{24}(q-f)^4 = \frac{q^5}{5!}\prod_{j=0}^{4}\phi_j + \mathcal{O}\left(\frac{\phi_0 q^6}{6!}\prod_{j=0}^{4}\phi_j\right) .$$

This lead us to the hypothesis that

$$p_n \approx \frac{q^n}{n!}\prod_{j=0}^{n-1}\phi_j + \mathcal{O}\left(\frac{\phi_0 q^{n+1}}{(n+1)!}\prod_{j=0}^{n-1}\phi_j\right) . \qquad (10)$$

This is something that we do not prove, but by dividing q into n intervals, it is easy to prove that $p_n \geq \frac{q^n}{n^n}\prod_{j=0}^{n-1}\phi_j$. Let $l = \log_2 L$, we can now calculate the complexity c of authentication key recovery with Fergusons method as

$$c \approx q \cdot 2^l / p_n . \qquad (11)$$

Complexity for the NIST tag and message lengths. Table 3 shows the complexities achieved by applying (11) to the NIST approved tag and maximum message lengths. The grey coloring shows the t_n that was used in the calculation. In a few cases the domain assumption does not hold as $2^{t_n} \approx q$. In these cases we have chosen n to overestimate rather than underestimate the complexity. Note that the complexities for authentication key recovery are independent of the encryption key length.

Table 3. Data complexity with Ferguson's method for full authentication key recovery.

t	32						64						96			104			112			120			128		
L	2^1	2^2	2^3	2^4	2^5	2^6	2^{11}	2^{13}	2^{15}	2^{17}	2^{19}	2^{21}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}
q	2^{22}	2^{20}	2^{18}	2^{15}	2^{13}	2^{11}	2^{32}	2^{29}	2^{26}	2^{23}	2^{20}	2^{17}	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
t_0	31	30	29	28	27	26	53	51	49	47	45	43	84	68	39	92	76	47	100	84	55	108	92	63	116	100	71
t_1	31	30	29	27	26	25	46	43	40	38	35	33	44	37	15	36	36	14	28	31	13	20	21	8	0	0	0
t_2	31	29	27	25	24	23	16	16	15	14	14	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_3	29	26	24	22	20	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_4	6	13	12	13	12	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	$2^{61.6}$	$2^{61.6}$	$2^{62.6}$	$2^{65.6}$	$2^{68.9}$	$2^{71.9}$	$2^{79.0}$	$2^{79.0}$	$2^{79.0}$	$2^{80.0}$	$2^{80.0}$	$2^{81.0}$	$2^{96.0}$	$2^{96.0}$	$2^{96.0}$	2^{104}	2^{104}	2^{104}	2^{112}	2^{112}	2^{112}	2^{120}	2^{120}	2^{120}	2^{128}	2^{128}	2^{128}

Our analysis shows that with Ferguson's method the security levels for 32-bit tags are below the old NIST requirement of 80-bit security (that was in place in 2007 when [3] was published), while 64-bit tags are just on the border. Only 112, 120, and 128 bit tags fulfill the current NIST requirement of 112-bit security. Unfortunately, NIST does not give any motivations for the exact restrictions they put on 32 and 64 bit tags, or for that matter the security levels they were assumed to give.

2.3 Our Improved Method for Authentication Key Recovery

In this section we propose several novel improvements to Ferguson's method for authentication key recovery. These improvements significantly reduce the security levels for short tags.

- The attacker may choose to modify a message with a message length that is smaller than the maximum message length L.
- After each successful forgery, the attacker may choose to modify a different message.
- The attacker may choose to modify messages with different lengths 2^{l_0}, 2^{l_1}, 2^{l_2}, …

Let the length of the first message be 2^{l_0} and let $l = \max(l_1, l_2, \dots)$. The probability that the attacker does not achieve a single successful forgery in q attempts is

$(1-\phi_0)^q$ in which case the attacker sends $q2^{l_0}$ blocks of data. The probability that the first successful forgery will occur on forgery attempt f is approximately $\phi_0(1-\phi_0)^{f-1}$ in which case the attacker sends at most $f2^{l_0}+(q-f)2^l$ blocks of data. The average number of blocks w sent by the attacker is therefore bounded by:

$$w \le (1-\phi_0)^q \cdot q2^{l_0} + \sum_{f=1}^{q} \phi_0(1-\phi_0)^{f-1} \cdot \left(q2^l - f(2^l - 2^{l_0})\right)$$

$$= q2^{l_0} + \frac{1}{2}\phi_0 q^2(2^l - 2^{l_0}) + \mathcal{O}(\phi_0^2 q^3 2^{l_0}) \quad . \tag{12}$$

We used SageMath to calculate the Taylor series and then collected the leading terms for the domain $\phi_0 \ll q^{-1}$. Using this improved method, the data complexity c of authentication key recovery is

$$c \approx q \cdot 2^{l_0}/p_n \quad . \tag{13}$$

Example. Let $l_0 = 0$ and $l = \log_2 L$. For 64-bit tags, the effective tag lengths are $t_0 = 64$, $t_1 = 64 - 2l$, and the complexity is $c \approx q \cdot 2^{l_0}/p_n = 2^{t_0+t_1+1}/q = 2^{129}/L^2 q$. Applying this to the column ($L = 2^{21}$, $q = 2^{17}$), the already low complexity is reduced from $2^{81.0}$ to $2^{70.0}$. It seems infeasible to increase the security level to 112 bits, as this would either restrict the message length too much or make deployments vulnerable to denial-of-service attacks.

Table 4. Data complexity with our improved method for full authentication key recovery.

t	32						64						96			104			112			120			128		
L	2^1	2^2	2^3	2^4	2^5	2^6	2^{11}	2^{13}	2^{15}	2^{17}	2^{19}	2^{21}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}	2^{12}	2^{28}	2^{57}
q	2^{22}	2^{20}	2^{18}	2^{15}	2^{13}	2^{11}	2^{32}	2^{29}	2^{26}	2^{23}	2^{20}	2^{17}	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
t_0	32	32	32	32	32	32	64	64	64	64	64	64	96	96	96	104	104	104	112	112	112	120	120	120	128	128	128
t_1	31	30	28	27	26	24	42	38	34	30	26	22	32	0	0	24	0	0	16	0	0	0	0	0	0	0	0
t_2	31	29	27	25	23	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_3	29	26	23	21	19	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
t_4	5	9	11	10	10	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	$2^{61.6}$	$2^{61.6}$	$2^{60.6}$	$2^{64.6}$	$2^{65.9}$	$2^{66.9}$	$2^{75.0}$	$2^{74.0}$	$2^{73.0}$	$2^{72.0}$	$2^{71.0}$	$2^{70.0}$	$2^{96.0}$	$2^{96.0}$	$2^{96.0}$	2^{104}	2^{104}	2^{104}	2^{112}	2^{112}	2^{112}	2^{120}	2^{120}	2^{120}	2^{128}	2^{128}	2^{128}

Table 4 shows the complexities achieved by applying our improved method (13) with $l_0 = 0$ and $l = \log_2 L$ to the NIST approved tag and maximum message lengths. This significantly reduces the data complexities of authentication key recovery for short tags. With our improved method, the security levels are 62–67 bits for 32-bit tags and 70–75 bits for 64-bit tags; this is below the old NIST requirement of 80-bit security and far below the current NIST requirement of 112-bit security.

3 Conclusions

The security levels of GCM and GMAC against authentication key recovery are for many tag sizes far below, not only the current NIST requirement of 112-bit security, but also the old NIST requirement of 80-bit security. With our improved authentication key recovery method, the security levels are 62–67 bits for the NIST approved usage of 32-bit tags, 70–75 bits for the NIST approved usage of 64-bit tags, and t bits for the NIST approved usage of t-bit tags where $t = 96$, 104, 112, 120, or 128. These security levels are independent of the encryption key size. It seems infeasible to increase the low security levels to 112 bits, as this would either restrict the message length too much or make deployments vulnerable to denial-of-service attacks.

One might argue that it is acceptable to allow a lower security level against authentication key recovery than encryption key recovery, especially if authentication key recovery requires online access to the hopefully short-lived GCM instances. With this arguing, 96-bit tags could be acceptable, even if they only offer 96 bits of security against authentication key recovery. We do not take a stance on this, but note that the current NIST requirements in NIST SP 800-57 Part 3 [11] states that the authentication key strength shall be equal or grater than 112 bits and that less than 112 bits of security shall not be used.

Appendix C of [3] first states that implementations should not provide feedback on the integrity of individual packets and then nevertheless heavily restricts the number of invocations of the decryption function. Our view is that feedback on the integrity of individual packets is almost always possible and we recommend NIST to forbid the use of GCM and GMAC with short tags.

We strongly recommend NIST to make a revision of [3] so that the security levels of all allowed options are clearly stated, that options offering less than 112-bit security are forbidden, or, if they are allowed, explain why a security level below 112 bits is acceptable.

We do however fully recommend GCM for usage with 128-bit tags, especially with AES-128. In fact we believe that with its excellent performance and proven security, it should be the first choice for everybody wanting an AEAD algorithm. We note that the design choices causing the security problems with truncated tags are also responsible for the excellent performance of GCM.

References

1. McGrew, Viega, "The Galois/Counter Mode of Operation (GCM)", May 2005, http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/gcm/gcm-revised-spec.pdf
2. McGrew, Viega, "The Security and Performance of the Galois/Counter Mode of Operation", October 2004, http://eprint.iacr.org/2004/193.pdf

3. NIST SP 800-38D, “Recommendations for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC”, November 2007, http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf
4. ISO/IEC 9772:2009, “Information technology -- Security techniques -- Authenticated encryption”, July 2008, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=46345
5. Ferguson, “Authentication weaknesses in GCM”, May 2005, http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC-GCM/Ferguson2.pdf
6. Joux, “Authentication Failures in NIST version of GCM”, 2006, http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/800-38_Series-Drafts/GCM/Joux_comments.pdf
7. Saarinen, “GCM, GHASH and Weak Keys”, 2011, http://www.iacr.org/archive/fse2012/75490220/75490220.pdf
8. CRYPTREC TR No. 2012, “Evaluation of Some Blockcipher Modes of Operation”, February 2011, http://www.cryptrec.go.jp/estimation/techrep_id2012_2.pdf
9. McGrew, Viega, “GCM Update”, May 2005, http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC-GCM/gcm-update.pdf
10. McGrew, Fluhrer, “Multiple forgery attacks against Message Authentication Codes”, May 2005, http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC-GCM/multi-forge-01.pdf
11. NIST SP 800-57 Part 3-Rev.1 “Recommendation for Key Management: Part 3 - Application-Specific Key Management Guidance”, January 2015, http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57Pt3r1.pdf
12. NSA, “Suite B Cryptography”, https://www.nsa.gov/ia/programs/suiteb_cryptography/
13. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness, http://competitions.cr.yp.to/caesar.html
14. IETF RFC 4543, “The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH”, May 2006, https://tools.ietf.org/html/rfc4543
15. IETF RFC 5288, “AES Galois Counter Mode (GCM) Cipher Suites for TLS”, August 2008, https://tools.ietf.org/html/rfc5288
16. IETF RFC 5647, “AES Galois Counter Mode for the Secure Shell Transport Layer Protocol”, August 2009, https://tools.ietf.org/html/rfc5647
17. Jones, “JSON Web Algorithms (JWA)” (IETF work in progress), January 2015, https://tools.ietf.org/html/draft-ietf-jose-json-web-algorithms-40
18. IEEE 802.1AE-2006, “Media Access Control (MAC) Security”, August 2006, http://standards.ieee.org/getieee802/download/802.1AE-2006.pdf
19. IEEE 802.11ad-2012, “Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band”, October 2012, http://standards.ieee.org/getieee802/download/802.11ad-2012.pdf

20. IEEE 802.11ac-2013, “Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz”, December 2013, http://standards.ieee.org/getieee802/download/802.11ac-2013.pdf

21. IEEE 1619.1-2007, “IEEE Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices”, May 2008

22. ANSI INCITS 496-2012, “Information technology - Fibre Channel Security Protocol 2 (FC-SP-2)”

23. IETF RFC 4106, “The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)”, June 2005, https://tools.ietf.org/html/rfc4106

24. IETF RFC 5084, “Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS)”, November 2007, https://tools.ietf.org/html/rfc5084

25. ECMA-409, “NFC-SEC-02: NFC-SEC Cryptography Standard using ECDH-256 and AES-GCM”, December 2014, http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-409.pdf

26. ECMA-411, “NFC-SEC-04: NFC-SEC Entity Authentication and Key Agreement using Symmetric Cryptography”, December 2014, http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-411.pdf

27. Kim, Lee, Kim, Park, Kwon, “The ARIA Algorithm and Its Use with the Secure Real-time Transport Protocol (SRTP)” (IETF work in progress), September 2014, https://tools.ietf.org/html/draft-ietf-avtcore-aria-srtp-07

28. McGrew, Igoe, “AES-GCM Authenticated Encryption in Secure RTP (SRTP)” (IETF work in progress), April 2015, https://tools.ietf.org/html/draft-ietf-avtcore-srtp-aes-gcm-15

29. W3C, “Web Cryptography API”, December 2014, http://www.w3.org/TR/WebCryptoAPI/

30. Oracle, “Java Platform, Standard Edition 8 API Specification”, https://docs.oracle.com/javase/8/docs/api/index.html

31. OASIS, “PKCS #11 Cryptographic Token Interface Current Mechanisms Specification Version 2.40” September 2014, http://docs.oasis-open.org/pkcs11/pkcs11-curr/v2.40/cs01/pkcs11-curr-v2.40-cs01.pdf

32. Microsoft, “Cryptography API: Next Generation”, https://msdn.microsoft.com/en-us/library/windows/desktop/aa376210

33. Kabatianskii, Smeets, Johansson, “On the Cardinality of Systematic Authentication Codes Via Error-Correcting Codes”, IEEE Transactions on Information Theory, Vol. 42, No 2, March 1996

34. Handschuh, Preneel, “Key-recovery attacks on universal hash function based MAC algorithms” CRYPTO 2008, http://www.cosic.esat.kuleuven.be/publications/article-1150.pdf