Cycling Attacks on GCM, GHASH and Other Polynomial MACs and Hashes

Markku-Juhani O. Saarinen

REVERE SECURITY
4500 Westgrove Drive, Suite 335, Addison, TX 75001, USA.
mjos@reveresecurity.com

Abstract. The Galois/Counter Mode (GCM) of operation has been standardized by NIST to provide single-pass authenticated encryption. The GHASH authentication component of GCM belongs to a class of Wegman-Carter polynomial hashes that operate in the field $\mathrm{GF}(2^{128})$. We present message forgery attacks that are made possible by its extremely smooth-order multiplicative group which splits into 512 subgroups. GCM uses the same block cipher key K to both encrypt data and to derive the generator H of the authentication polynomial for GHASH. In present literature, only the trivial weak key $H = 0$ has been considered. We show that GHASH has much wider classes of weak keys in its 512 multiplicative subgroups, analyze some of their properties, and give experimental results on AES-GCM weak key search. Our attacks can be used not only to bypass message authentication with garbage but also to target specific plaintext bits if a polynomial MAC is used in conjunction with a stream cipher. These attacks can also be applied with varying efficiency to other polynomial hashes and MACs, depending on their field properties. Our findings show that especially the use of short polynomial-evaluation MACs should be avoided if the underlying field has a smooth multiplicative order.

Keywords: Cryptanalysis, Galois/Counter Mode, AES-GCM, Cycling Attacks, Weak Keys.

1 Introduction

Authenticated encryption modes and algorithms provide confidentiality and integrity protection in a single processing step. This results in performance and cost advantages as data paths can be shared.

The Galois/Counter Mode (GCM) has been standardized by NIST [1] to be used in conjunction with a 128-bit block cipher for providing authenticated encryption functionality. When paired with the AES [2] algorithm, the resulting AES-GCM combination has been used as a replacement to dedicated hash-based HMAC [3] in popular cryptographic protocols such as SSH [4], IPSec [5] and TLS [6].

In AES-GCM, data is encrypted using the Counter Mode (CTR). A single AES key K is used to both encrypt data and to derive authentication secrets. The component that is used by GCM to produce a message authentication code is called GHASH. GCM also supports Additional Authenticated Data (AAD) which is authenticated using GHASH but transmitted as plaintext.

The GHASH algorithm belongs to a widely studied class of Wegman-Carter [7, 8] polynomial MACs. These were originally proposed in context of polynomial evaluation independently by three authors [9–11]. A good overview of the genealogy and evolution is by Bernstein [12, 13]. The security bounds known for these algorithms indicate that a n-bit tag will give $2^{-\frac{n}{2}}$ security against forgery [12, 14].

In this paper we give further evidence that this is not only the security lower bound but an upper bound as well. It can be argued that universal hashes sacrifice communication bandwidth for convenience as traditional hash-based MACs are designed to reach the information theoretic 2^{-n} bound against message forgery and are therefore technically somewhat inferior, especially for short MACs. The security against cycling attacks depends very sharply on the properties of the underlying field.

This paper is structured as follows. We give a description of GHASH in Section 2, followed by a key observation regarding collisions derived from cycles in Section 3. Section 4 contains an analysis of cycle lengths and group orders. In Section 5 we discuss the probability of successful forgery. Section 6 briefly considers targeted attacks against underlying protocols. Section 7 contains a test and experimental results related to cycle lengths. We discuss the security of other polynomial mac constructions in Section 8 and conclude in Section 9.

2 Description of GHASH

Let X be a concatenation of unencrypted authenticated data, CTR-encrypted ciphertext, and padding. This data is split into m 128-bit blocks X_i:

$$X = X_1 \mid\mid X_2 \mid\mid \cdots \mid\mid X_m.$$

AES is used to derive the root authentication key $H = E_K(0)$. The same AES key K is also used as the data encryption key. In the present work we assume that H is unknown to the attacker as the scheme would be otherwise trivially breakable.

GHASH is based on operations in the finite field $\mathrm{GF}(2^{128})$. Horner's rule is used in this field to evaluate the polynomial Y.

$$Y_m = \sum_{i=1}^{m} X_i \times H^{m-i+1}. \tag{1}$$

The authentication tag is $T = Y_m + E_K(IV \mid\mid 0^{31} \mid\mid 1)$, assuming that a 96-bit Initialization Vector (IV) is used. The IV value must never be reused as that would lead to the "forbidden attack" discussed by Joux in [15].

3 Collisions from Weak Keys

It has been observed that if $E_K(0) = H = 0$, the polynomial Y evaluates to zero and the security of GHASH breaks down. In fact, some sources assume that this pathological case is the only weak key [16]. AES keys K that produce this fixed point are not known.[1] However, It is easy to see why such keys should exist for AES, especially when the size of K is more than 128 bits.

Our main observation is that sometimes the powers of H will repeat in a relatively short cycle. A trivial example occurs when H is equal to the identity element 1, which will lead to all powers being equal. Due to the commutativity of addition in Equation 1, a GHASH collision can be achieved by swapping any two ciphertext blocks X_i and X_j. This amounts to message forgery.

More generally, if we know that $H^{m-i+1} = H^{m-j+1}$ with $i \neq j$, we may simply swap ciphertext blocks X_i and X_j and the resulting authentication tag stays unmodified which amounts to message forgery. This can be easily observed from Equation 1. Elementary group theory tells us that the powers of H will repeat in cycles which are determined by $n = \mathrm{ord}(H)$, the multiplicative order of H. Hence we may produce collisions by swapping X_i and X_{i+nm} for arbitrary i and m.

4 Cycle Lengths and Group Orders

From Lagrange's theorem in group theory we know that all subgroups divide the group of order $2^{128} - 1$. Numbers of this type factor into Fermat numbers

$$2^{2^n} - 1 = \prod_{i=1}^{n} 2^{2^{i-1}} + 1. \tag{2}$$

We can easily obtain the full factorization of $2^{128} - 1$:

$$3 * 5 * 17 * 257 * 641 * 65537 * 274177 * 6700417 * 67280421310721. \tag{3}$$

As this is a "smooth number", we can see that there are classes of H and therefore K values that produce cycles of length $n = 1, 3, 5, 15, 17, 51, \ldots$; any one of the $2^9 = 512$ subset products of the primes in Equation 3 is a valid group order.[2]

[1] Some block ciphers such as GOST allow such fixed-point keys to be very easily found.

[2] The term *smooth number* comes from factorization theory and indicates that a number factors into a large number of small primes.

4.1 Illustrating Multiplicative Subgroup Cycles

We will give a few trivial examples. Due to the way finite field arithmetic is defined in the GCM standard [1], the identity element with $\mathrm{ord}(H) = 1$ is:

```
H = 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Apparently this was considered as the "first bit" by those who originally implemented GCM. Otherwise standard polynomial arithmetic is used with the field representation defined by the reducing polynomial $x^{128} + x^7 + x^2 + x + 1$.

The following two elements will produce a cycle of length $\mathrm{ord}(H) = 3$ (the cycle obviously goes through the identity as well):

```
H = 10 D0 4D 25 F9 35 56 E6 9F 58 CE 2F 8D 03 5A 94
H = 90 D0 4D 25 F9 35 56 E6 9F 58 CE 2F 8D 03 5A 94
```

These four elements have $\mathrm{ord}(H) = 5$:

```
H = 46 36 BD BD 1C 76 43 D3 4E E4 BB 1B F9 CA 08 4F
H = 92 17 8D 40 26 DA 1D CA 42 96 77 87 30 EB 9A 9E
H = 82 C7 C0 65 DF EF 4B 2C DD CE B9 A8 BD E8 C0 0A
H = D6 E6 F0 98 E5 43 15 35 D1 BC 75 34 74 C9 52 DB
```

We do not know which actual AES keys produce these H values, nor do we recommend testing against these particular values as the probability of hitting them is exceedingly small.

Note that a cycle of length such as $15 = 3 * 5$ also contains the beforementioned component groups of order 1, 3 and 5, in addition to the 6 unique elements that can act as a generator of the cycle of order 15. This is entirely analogous to arithmetic in the addition group of integers modulo 15; 0 will generate a "cycle" of one element when repeatedly added to itself, 5 and 10 will generate a cycles of order 3, the four elements { 3, 6, 9, 12 } cycles of order 5 and the rest of the numbers will have order 15.

5 Message Forgery

We know that the field $\mathrm{GF}(2^{128})$ offers a generous serving of $2^9 = 512$ different multiplicative subgroups. Figure 1 shows that these are quite evenly distributed in the range due to the nearly log-uniform progression of the factors.

In our attack the adversary does not know H but will simply attempt a blind forgery by swapping two (or more) message blocks in transit as discussed in Section 3.

It is easy to show that it is sufficient that the group order divides the distance between swapped elements. Since each subgroup of size n has exactly n elements, we arrive at the following observation:

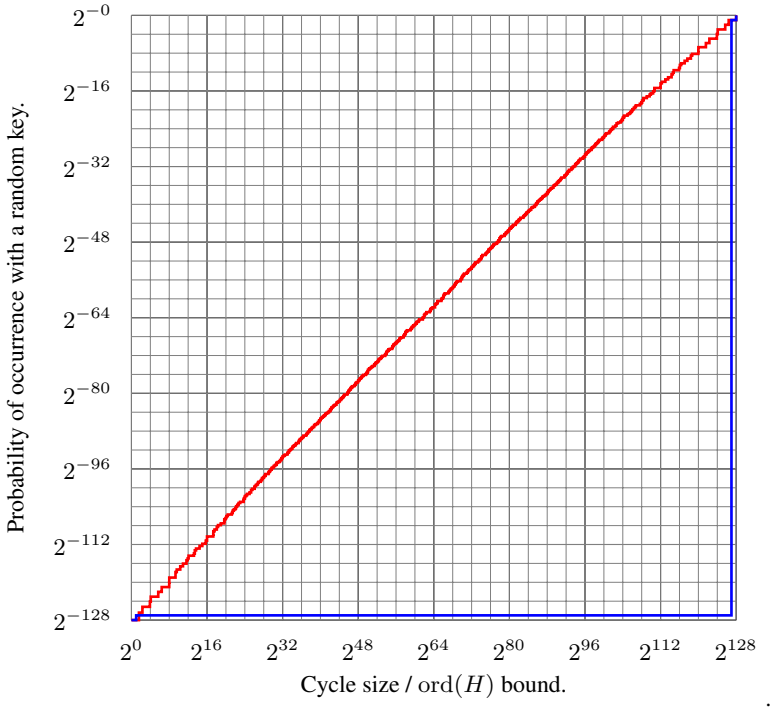


Fig. 1. GCM / GHASH: probability of hitting a multiplicative subgroup (cycle) of given (or smaller) size with a random authentication generator H in $\mathrm{GF}(2^{128})$. For comparison we also graph the security for $\mathrm{GF}(2^{127})$, which is entirely contained in the lower and right borders of the graph due to the fact that its multiplicative group order $2^{127} - 1$ is a prime.

Theorem 1. *Let n be a number satisfying $\gcd(2^{128} - 1, n) = n$. Blindly swapping blocks X_i and X_j, where $i \equiv j \pmod{n}$ will result in a successful forgery with probability of at least $\frac{n+1}{2^{128}}$ for some random H.*

Proof. The distance congruence implies that the distance between X_i and X_j is a multiple of n. The $\gcd(2^{128} - 1, n) = n$ condition implies that n is one of the $2^9 = 512$ possible multiplicative subgroup sizes in $\mathrm{GF}(2^{128})$. If indeed $\mathrm{ord}(H) \mid n$ then $H^i = H^j$ and the forgery is successful due to commutativity of equation 1. We observe that the cycles are unique; there are n members in a subgroup of size n and the set of n elements is unique to each subgroup size. Hence the probability of hitting one of these cycle elements is $\frac{n}{2^{128}}$. In addition there is the pathological case $H = 0$ which completes the proof. □

If the $\gcd$ condition given in Theorem 1 does not hold, we have no reason to expect that the forgery is successful with a probability higher than $\frac{1}{2^{128}}$.

Assuming that an oracle has indicated a successful message forgery, any number of consecutive forgeries can be produced with probability 1 if the key remains unchanged (IV may change).

6 Targeted Multiple Bit Forgeries

Our attacks enable elaborate message forgeries against authenticated encryption hybrids such as GCM due to the fact that the CTR encryption mode behaves like a stream cipher; flipping a ciphertext bit will result the corresponding plaintext bit to be flipped. This is especially true for slightweight protocols that combine a short binary polynomial MAC with a stream cipher.

If $\mathrm{ord}(H)|(i-j)$ the authentication tag will remain valid as long as the equation

$$X_i \times H^{m-i+1} + X_j \times H^{m-j+1} = c \tag{4}$$

holds for some (unknown) constant c related to the authentication tag. If we write $H^{m-i+1} = H^{m-j+1} = H_c$, this can be simplified to

$$X_i + X_j = c \times H_c^{-1}. \tag{5}$$

We see that the authentication tag will be valid if the sum of ciphertext blocks on the left side of Equation 5 remains constant. One may therefore flip *individual bits* in block X_i if the corresponding bit in X_j is also flipped. Any number of such modifications can be done to a message without affecting the probability of success (assuming that the same distance is used) indicated by Theorem 1.

7 Testing for AES-GCM Weak Keys

We know that finding weak H values is easy, so a natural question arises on how to determine weak AES keys K that produce these weak H roots.

To determine group order, we use a simple algorithm which is related to the Silver-Pohlig-Hellman algorithm for discrete logarithms [17]. Our algorithm is based on the following elementary observation:

Theorem 2. *Let p be one of the prime divisors given in Equation 3. If and only if p divides $\mathrm{ord}(H)$ we have*

$$H^{\frac{2^{128}-1}{p}} \neq 1. \tag{6}$$

Proof. Let g be a generator of the full multiplicative group; $\mathrm{ord}(g) = 2^{128} - 1$. Then each element $H \neq 0$ can be expressed as a power $H = g^h$ for some h,

$0 \leq h < 2^{128} - 1$. Raising an element to power q, where $q \mid 2^{128} - 1$, sets the index modulo q to zero: $(g^h)^q = g^{qh}$. Since $\frac{2^{128}-1}{p}$ is divisible with all prime divisors q_i of the group order except p, we see that the condition of Equation 6 only holds if $h \neq 0 \pmod{p}$, which is equivalent to the condition $p \mid \mathrm{ord}(H)$. □

By performing the exponentiation test of Theorem 2 for each one of the nine prime divisors of $2^{128} - 1$ in Equation 3, we may completely determine the multiplicative order of H.

7.1 An Efficient Algorithm for Subgroup Size

Raising a finite field element to a Fermat $F_n = 2^{2^n} + 1$ power can be done efficiently. It is well known that squaring operation is "linear" in $\mathrm{GF}(2^n)$ [18]. For $\mathrm{GF}(2^{128})$, a unique 128×128 bit matrix $\mathbf{M}_0$ exists that satisfies

$$X^2 = \mathbf{M}_0 X \tag{7}$$

for all X. In the following $\mathbf{M}_0 X$ denotes a matrix multiplication where X is interpreted as a vector of 128 bits and $X \times X = X^2$ is a multiplication where X is interpreted as a (polynomial) member of $\mathrm{GF}(2^{128})$.

By squaring $\mathbf{M}_0$, we obtain $\mathbf{M}_1 = \mathbf{M}_0^2$ which satisfies $X^4 = \mathbf{M}_1 X$ for all X. By repeating this process we can rapidly compute $\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_6$ that satisfy

$$X^{2^{2^i}} = \mathbf{M}_i X. \tag{8}$$

Once the matrices (table lookups) $\mathbf{M}_i$ have been initialized, raising the authentication key H to a Fermat number power can be achieved with:

$$H^{F_n} = \mathbf{M}_n H \times H. \tag{9}$$

Therefore this operation can be made with a table lookup (multiplication with $\mathbf{M}_n$) and a single Galois Field multiplication. The matrices need to be computed only once as they are independent from particular H.

Since $2^{128} - 1 = \prod_{i=0}^{6} F_i$, checking whether the group order is of H is divisible with Fermat number F_i involves raising H to all Fermat powers F_j *except* F_i. For example, to check whether or not group order is divisible with $F_3 = 257$, we may see if this equation holds:

$$\mathbf{M}_6(\mathbf{M}_5(\mathbf{M}_4(\mathbf{M}_2(\mathbf{M}_1(\mathbf{M}_0 H \times H) \times H) \times H) \times H) \times H) \times H = 1. \tag{10}$$

The Fermat numbers F_5 and F_6 are not primes (unlike F_0, F_1, F_2, F_3 and F_4 which are indeed the only known Fermat primes). Here the technique involves

first powering H to all Fermat powers except $F_5 = 641 * 6700417$ or $F_6 = 274177 * 67280421310721$. Then then we use a conventional square-multiply exponentiation method to individually check these two subfactors.

In practice the matrix $\mathbf{M}_i$ multiplication is implemented as byte-based table lookups with seven $16 \times 256 \times 128$ - bit tables. The initialization of these tables is very fast as $\mathbf{M}_{i+1}$ can be developed from $\mathbf{M}_i$ with a loop of $16 * 256$ table lookups. Significant speedups are achieved by reusing partial results.

7.2 Experimental Results

Using the techniques outlined in the previous subsection, we have developed a reasonably efficient cycle determination code specifically for GCM's $\mathrm{GF}(2^{128})$, together with an AES-128 key setup and encryption function for deriving H values from K values.

Our implementation is currently able to fully determine the order of 25000 AES keys per second on a Linux laptop that has a single 1.7 gHz AMD V140 processor.

Over couple of days we tested 2^{32} AES-128 keys and found progressively smaller subgroups:

$n \approx 2^{126.4}$	$K =$ 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02
$n \approx 2^{125.6}$	$K =$ 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03
...	
$n \approx 2^{96.52}$	$K =$ 00 00 00 00 00 00 00 00 00 00 00 00 24 3E 8B 40
$n \approx 2^{96.00}$	$K =$ 00 00 00 00 00 00 00 00 00 00 00 00 37 48 CF CE
$n \approx 2^{93.93}$	$K =$ 00 00 00 00 00 00 00 00 00 00 00 00 42 87 3C C8
$n \approx 2^{93.41}$	$K =$ 00 00 00 00 00 00 00 00 00 00 00 00 EC 69 7A A8

As indicated by Figure 1, a significantly smaller group than $2^{128-32} = 2^{96}$ was found with 2^{32} effort, due to the large number of multiplicative subgroup sizes available in $\mathrm{GF}(2^{128})$.

There is clearly room for improvement. The search is fully parallelizable, and hence a massively parallel FPGA or GPU-based search could be performed to find subgroups of magnitude $n \approx 2^{64}$ or less.

8 Other Polynomial-Evaluation MACs

The security of Polynomial-evaluation MACs against attacks of this type can be determined from the factorization of the group size in straightforward fashion. Trivial changes can introduce radical differences.

One may consider this difference by comparing the binary field $GF(2^{127})$ and the prime field $\mathrm{GF}(2^{127}-1)$. Here the binary field is perfectly secure due to the fact that $2^{127}-1$ is indeed a prime (if the message is processed in 127-bit blocks). However, the latter prime field has a multiplicative order $2^{127}-2$ which factors spectacularly into 15 pieces and is exceptionally weak against a cycling attack! We note that the HASH127 MAC is based on the latter [19]. This is illustrated in Figure 1.

If a prime field is to be used, we recommend Sophie Germain primes where $q=(p-1)/2$ is also a prime. Such a field has well-understood cycle properties which may be easily determined using the Legendre symbol from elementary number theory. A practical alternative to GCM would use a Sophie Germain prime such as $GF(2^{128}+12451)$, which is slightly larger than the 2^{128} to deter trivial collisions.

It is clear that risks rise quadratically when GCM is used with a 64-bit block cipher as suggested in Appendix A of [20]. There is a substantial risk of hitting a bad long-term key and therefore we recommend against using the 64-bit GCM.

9 Conclusions and Future Work

We have shown that the GHASH algorithm has other weak key classes besides the trivial $H=0$ case considered in current literature [16]. This is a result of the multiplicative group of $\mathrm{GF}(2^{128})$ having a particularly smooth order.

Our attacks allow specific plaintext bits to be targeted by modifying ciphertext bits, which can have a devastating effect when a short polynomial MAC over a binary field is combined with a stream cipher in a (lightweight) communication protocol. The probability of randomly hitting an exploitable weak key with a AES-GCM cryptographic protocol such as SSH [4], IPSec [5] or TLS [6] is very small.

However, malicious players may exploit subtle weaknesses in cryptographic protocols in surprising ways. One feature of cycle attacks is that an attacker may first test for short cycles and then force a re-keying event if the test fails; once a long-term key with a short cycle is found, she may exploit it any number of times.

We have also described a straightforward method of detecting GHASH weak keys. We performed an exhaustive experiment that found many AES-128 keys that produce H with order below $n \approx 2^{96}$.

We suggest that binary fields $\mathrm{GF}(2^n)$ with prime 2^n-1 or Sophie Germain prime fields are used in constructions of this type as this minimizes the total number of weak keys. This was illustrated with the surprising observation that

$\mathrm{GF}(2^{127})$ is perfectly secure against this type of attack while GCM's $\mathrm{GF}(2^{128})$ is not.

One interesting future research direction and open question is the feasibility of mapping the weak H values to K symmetric keys with various block ciphers other than AES.

References

1. NIST: Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC. NIST Special Publication 800-38D (2007)
2. NIST: The advanced encryption standard (AES). FIPS Publication 197 (2001)
3. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In: CRYPTO '96. Volume 1109 of LNCS., Springer (1996) 1 – 55
4. Igoe, K., Solinas, J.: AES Galois counter mode for the secure shell transport layer protocol. IETF Request for Comments 5647 (2009)
5. Law, L., Solinas, J.: Suite B cryptographic suites for IPsec. IETF Request for Comments 4869 (2007)
6. Salter, M., Rescorla, E., Housley, R.: Suite B profile for transport layer security (TLS). IETF Request for Comments 5430 (2009)
7. Wegman, M.N., Carter, J.L.: New classes and applications of hash functions. In: 20th annual symposium on foundations of computer science, IEEE Computer Society, New York (1979)
8. Wegman, M.N., Carter, J.L.: New hash functions and their use in authentication and set equality. Journal of Computer and System Sciences **22** (1981) 265–279
9. den Boer, B.: A simple and key-economical unconditional authentication scheme. Journal of Computer Security **2** (1993) 65–71
10. Taylor, R.: An integrity check value algorithm for stream ciphers. In: CRYPTO '93. Volume 773 of LNCS., Springer (1994) 40–48
11. Bierbrauer, J., Johansson, T., Kabatianskii, G., Smeets, B.: On families of hash functions via geometric codes and concatenation. In: CRYPTO '93. Volume 773 of LNCS., Springer (1994) 331–342
12. Bernstein, D.J.: Stronger security bounds for Wegman-Carter-Shoup authenticators. In: EUROCRYPT 2005. Volume 3494 of LNCS., Springer (2005) 164–180
13. Bernstein, D.J.: The Poly1305-AES message-authentication code. In: FSE 2005. Volume 3557 of LNCS., Springer (2005) 32–49
14. Sarkar, P.: A trade-off between collision probability and key size in universal hashing using polynomials. Designs, Codes and Cryptography **58**(3) (2011) 271–278
15. Joux, A.: Authentication failures in NIST version of GCM. NIST Comment (2006)
16. Handschuh, H., Preneel, B.: Key-recovery attacks on universal hash function based MAC algorithms. In: CRYPTO 2008. Volume 5157 of LNCS., Springer (2008) 144–161
17. Pohlig, S., Hellman, M.: An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. IEEE Transactions on Information Theory **24**(1) (1978) 106–110
18. Ferguson, N.: Authentication weaknesses in GCM. NIST Comment. (May 2005)
19. Bernstein, D.J.: Floating-point arithmetic and message authentication. `http://cr.yp.to/papers.html#hash127` (1999)
20. McGrew, D.A., Viega, J.: The Galois/counter mode of operation (GCM). Submission to NIST. (2005)

Document version 20111118062200.