NIST Special Publication 800-38D
**DRAFT (April, 2006)**


# Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) for Confidentiality and Authentication

Morris Dworkin

C O M P U T E R  S E C U R I T Y

# Abstract

This Recommendation specifies the Galois/Counter Mode (GCM), an authenticated encryption mode of operation for a symmetric key block cipher.



KEY WORDS: authentication; block cipher; cryptography; information security; integrity; message authentication code; mode of operation.

# Table of Contents



# Table of Figures

# 1 Purpose

This publication is the fourth Part in a series of Recommendations regarding modes of operation of symmetric key block ciphers.

# 2 Authority

This document has been developed by the National Institute of Standards and Technology (NIST) in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets, but such standards and guidelines shall not apply to national security systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), Securing Agency Information Systems, as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This Recommendation has been prepared for use by federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright. (Attribution would be appreciated by NIST.)

Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other federal official.

Conformance testing for implementations of the mode of operation that is specified in this Part of the Recommendation will be conducted within the framework of the Cryptographic Module Validation Program (CMVP), a joint effort of NIST and the Communications Security Establishment of the Government of Canada. An implementation of a mode of operation must adhere to the requirements in this Recommendation in order to be validated under the CMVP. The requirements of this Recommendation are indicated by the word "shall."

# 3 Introduction

This Recommendation specifies an authenticated encryption algorithm called Galois/Counter Mode (GCM) constructed from an approved symmetric key block cipher with a block size of 128 bits, such as the Advanced Encryption Standard (AES) algorithm that is specified in Federal Information Processing Standard (FIPS) Pub. 197 [2]. GCM provides assurance of confidentiality of data using a variation of the Counter mode of operation for encryption. GCM provides assurance of authenticity of the confidential data using a universal hash function that is defined over a binary Galois (i.e., finite) field. GCM can also provide authentication assurance for additional data that is not encrypted. This assurance is stronger than that provided by a (non-cryptographic) checksum or error detecting code, in that GCM can detect both 1) accidental

modifications of the data and 2) intentional, unauthorized modifications.

The GCM authenticated encryption and authenticated decryption functions are relatively efficient and can be computed in parallel; consequently, high-throughput implementations are possible in both hardware and software. GCM has several other useful characteristics, including the following:

- The GCM functions are "online" in the sense that the lengths of the confidential data and the additional, non-confidential data are not required in advance; instead, the lengths can be calculated as the data arrives and is processed.
- The GCM functions do not require the inverse direction ("decryption") of the underlying block cipher.
- The authenticity of the protected data can be verified independently from the recovery of the confidential data from its encrypted form.
- If the unique initialization string is predictable and the length of the confidential data is known, then the block cipher invocations within the GCM encryption mechanism can be precomputed.
- If some or all of the additional, non-confidential data is known to be fixed, then the corresponding elements of the GCM authentication mechanism can be precomputed.
- The unique initialization string for the GCM functions can vary in length.

The designers of GCM are McGrew and Viega. They submitted GCM to NIST in Ref. [3], and they discuss in detail its security and performance in Ref. [4].

# 4 Definitions, Abbreviations, and Symbols

## *4.1 Definitions and Abbreviations*

| | |
|---|---|
| AAD | Additional Authenticated Data |
| Additional Authenticated Data | The input data to the authenticated encryption function that is authenticated but not encrypted. |
| AES | Advanced Encryption Standard. |
| Approved | FIPS approved or NIST recommended: an algorithm or technique that is either 1) specified in a FIPS or a NIST Recommendation, or 2) adopted in a FIPS or a NIST Recommendation. |
| Authenticated Decryption | The function of GCM in which the ciphertext is decrypted into the plaintext, and the authenticity of the ciphertext and the AAD is verified. |

| | |
|---|---|
| Authenticated Encryption | The function of GCM in which the plaintext is encrypted into the ciphertext, and an authentication tag is generated on the AAD and the ciphertext. |
| Authentication Tag (Tag) | A cryptographic checksum on data that is designed to reveal both accidental errors and the intentional modification of the data. |
| Authenticity | The property that data originated from its purported source. |
| Bit | A binary digit: 0 or 1. |
| Bit String | A finite, ordered sequence of bits. |
| Block | For a given block cipher, a bit string whose length is the block size of the block cipher. |
| Block Cipher | A parameterized family of permutations on bit strings of a fixed length. |
| Block Size | For a given block cipher and key, the fixed length of the input (or output) bit strings. |
| Byte | A sequence of 8 bits. |
| Byte String | A finite, ordered sequence of bytes. |
| Ciphertext | The encrypted form of the plaintext. |
| Exclusive-OR | The bitwise addition, modulo 2, of two bit strings of equal length. |
| FIPS | Federal Information Processing Standard. |
| Forward Cipher Function | A permutation on blocks that is determined by the choice of a key for a given block cipher. |
| GCM | Galois/Counter Mode |
| Initialization Vector | A nonce that is associated with an invocation of authenticated encryption on a particular plaintext and AAD. |
| Inverse Cipher Function | The function that is the inverse of the forward cipher function for a given key. |
| IV | Initialization Vector |

| | |
|---|---|
| Key | The parameter of the block cipher that determines the selection of the forward cipher function from the family of permutations. |
| Least Significant Bit(s) | The right-most bit(s) of a bit string. |
| Mode of Operation (Mode) | An algorithm for the cryptographic transformation of data that features a block cipher. |
| Most Significant Bit(s) | The left-most bit(s) of a bit string. |
| NIST | National Institute of Standards and Technology. |
| Nonce | A value that is used only once within a specified context. |
| Permutation | An invertible function. |
| Plaintext | The input data to the authenticated encryption function that is both authenticated and encrypted. |
| XOR | Exclusive-OR. |

## *4.2 Symbols*

### *4.2.1 Variables*

| | |
|---|---|
| $C$ | The ciphertext. |
| $H$ | The hash subkey. |
| $IV$ | The initialization vector. |
| $K$ | The block cipher key. |
| $P$ | The plaintext. |
| $R$ | The constant for the multiplication algorithm. |
| $T$ | The authentication tag. |

### *4.2.2 Operations and Functions*

| | |
|---|---|
| $0^s$ | The bit string that consists of s '0' bits. |
| $\text{CIPH}_K(X)$ | The output of the forward cipher function of the block cipher under the key $K$ applied to the block $X$. |

| | |
|---|---|
| $\mathrm{GCTR}_K(ICB, X)$ | The output of the GCTR function for a given block cipher with key $K$ applied to the bit string $X$ with an initial counter block $ICB$. |
| $\mathrm{GHASH}_H(X)$ | The output of the GHASH function under the parameter $H$ applied to the bit strings $X$. |
| $\mathrm{inc}(X)$ | The output of the GCM incrementing function applied to the block $X$. |
| $\mathrm{int}(X)$ | The integer for which the bit string $X$ is a binary representation. |
| $\mathrm{len}(X)$ | The bit length of the bit string $X$. |
| $\mathrm{LSB}_s(X)$ | The bit string consisting of the $s$ right-most bits of the bit string $X$. |
| $\mathrm{MSB}_s(X)$ | The bit string consisting of the $s$ left-most bits of the bit string $X$. |
| $\lceil x \rceil$ | The least integer that is not less than the real number $x$. |
| $[x]_s$ | The binary representation of the non-negative integer $x$ as a string of $s$ bits, where $x<2^s$. |
| $X >> 1$ | The bit string that results from discarding the rightmost bit of the bit string $X$ and appending a '0' bit on the left. |
| $X \| Y$ | The concatenation of two bit strings $X$ and $Y$. |
| $X \oplus Y$ | The bitwise exclusive-OR of two bit strings $X$ and $Y$ of the same length. |
| $X \cdot Y$ | The product of two blocks, $X$ and $Y$, regarded as elements of a certain binary Galois field. |

# 5 Elements of GCM

The elements of GCM and the associated notation and requirements are introduced in the three sections below. The underlying block cipher and key are discussed in Sec. 5.1. The data elements of the authenticated encryption and authenticated decryption functions of GCM are discussed in Sec. 5.2. The cryptographic primitives for confidentiality and authentication within these two functions are discussed in Sec. 5.3.

## *5.1 Block Cipher*

The operations of GCM depend on the choice of an underlying symmetric key block cipher and thus can be considered a mode of operation (mode, for short) of the block cipher. The GCM key is the block cipher key (the key, for short).

For any given key, the underlying block cipher of the mode consists of two functions that are inverses of each other. The choice of the block cipher includes the designation of one of the two functions of the block cipher as the forward cipher function, as in the specification of the AES algorithm in Ref. [2]. GCM does not employ the inverse cipher function.

The forward cipher function is a permutation on bit strings of a fixed length; the strings are called blocks. The length of a block is called the block size. The key is denoted $K$, and the resulting forward cipher function of the block cipher is denoted $\mathrm{CIPH}_K$.

The underlying block cipher shall be approved, and the block size shall be 128 bits. The key shall be generated uniformly at random, or close to uniformly at random, i.e., so that each possible key is (nearly) equally likely to be generated. The key shall be secret and shall be used exclusively for GCM with the chosen block cipher. To fulfill the requirements on the key, the key should be established among the parties to the information within an approved key management structure; the details of the establishment and management of keys are outside the scope of this Recommendation.

### *5.2 Input and Output Data*

The two functions that comprise GCM are called authenticated encryption and authenticated decryption. The requirements and notation for the input and output data of these functions are discussed in Secs. 5.2.1 and 5.2.2.

#### *5.2.1 Authenticated Encryption*

Given the selection of an approved block cipher and key, there are three input strings to the authenticated encryption operation:

- a plaintext, denoted $P$;
- additional authenticated data (AAD), denoted $A$; and
- an initialization vector (IV), denoted $IV$.

The bit lengths of these inputs shall meet the following requirements:

- $\mathrm{len}(P) \le 2^{39} - 256$;
- $\mathrm{len}(A) \le 2^{64}$;
- $1 \le \mathrm{len}(IV) \le 2^{64}$.

The plaintext and AAD are the two categories of data that GCM protects. GCM protects the authenticity of both the plaintext and the AAD (and the IV, for that matter); GCM also protects the confidentiality of the plaintext, while the AAD is left in the clear. For example, within a network protocol, the AAD might include addresses, ports, sequence numbers, protocol version numbers, and other fields that indicate how the plaintext should be treated.

The IV is a nonce that is associated with the data to be protected: for any two distinct invocations of the authenticated encryption function, across all implementations for the given block cipher

and key, IVs shall have no more than a negligible chance of repeating. Thus, the IV essentially determines an invocation of authenticated encryption. For a given key, the lengths of the IVs may vary; for optimal efficiency, the recommended default length is 96 bits.

The following two bit strings comprise the output data of the authenticated encryption function:

- A ciphertext, denoted *C*, whose bit length is the same as that of the plaintext.
- An authentication tag, or tag, for short, denoted *T*.

The bit length of the authentication tag is an important security parameter that is denoted *t*. A fixed value for *t* between 96 and 128, inclusive, shall be associated with the key. If the length of the IV is not the default recommendation of 96, then *t* shall be 128.

Although GCM is defined on bit strings, the bit lengths of the plaintext/ciphertext, the AAD, the IV, and the tag shall all be multiples of 8, so that these values are byte strings. Other restrictions may be imposed on the lengths of inputs and outputs, as discussed in Sec. 6 below.

#### *5.2.2 Authenticated Decryption*

Given the selection of an approved block cipher and key and an associated tag length, the inputs to the authenticated decryption function are values for *IV*, *A*, *C*, and *T*, as described in Sec. 5.2.1 above. The output is one of the following:

- the plaintext *P* that corresponds to the ciphertext *C*, or
- an indication that the inputs are not authentic, denoted *FAIL*.

There always exists exactly one plaintext *P* such that the authenticated encryption function invoked on *P* with the given *A* and *IV* would produce the given *C* as the ciphertext output. If the same invocation would produce the given *T* as the tag output, then *P* will be the output of the authenticated decryption function. Otherwise, *FAIL* will be the output of the authenticated decryption function.

The authentication assurance that can be inferred from each result is discussed in Appendix A.

### *5.3 Primitives for Confidentiality and Authentication*

The mechanism for the confidentiality of the plaintext within GCM is a variation of the Counter mode [5], with a particular incrementing function, denoted inc, for generating the necessary sequence of counter blocks. The first counter block for the plaintext encryption is generated by incrementing a block that is generated from the IV.

The authentication mechanism within GCM is based on a hash function, called GHASH[1], that features multiplication by a fixed parameter, called the hash subkey, within a binary Galois field.

[1] The designers of GCM define GHASH slightly differently. The simpler version of GHASH in this Recommendation does not obtain all of the properties that are shown for the designers' definition in Ref. [4].

The hash subkey, denoted $H$, is generated by applying the block cipher to the "zero" block. The resulting instance of this hash function, denoted $\text{GHASH}_H$, is used to compress an encoding of the AAD and the ciphertext into a single block, which is then encrypted to produce the authentication tag. This instance of the hash function serves an additional role in GCM, namely, to generate the initial block when the length of the IV is not 96 bits.

GHASH is a keyed hash function but not, on its own, a cryptographic hash function. This Recommendation only approves GHASH for use within the context of GCM.

The intermediate values in the execution of the GCM functions shall be secret. In particular, this requirement precludes the system in which GCM is implemented from using the hash subkey publicly for some other purpose, for example, as an unpredictable value or as an integrity check value on the key.

# 6 Conformance

Implementations may restrict the acceptable lengths of the plaintext, the AAD, the IV, and the tag, consistent with the requirements in Sec. 5.2.1. In particular, a conforming implementation may enforce maximums lengths for the plaintext, the AAD, and the IV that are smaller than the maximum lengths in Sec. 5.2.1. Similarly, an implementation may restrict the possible tag lengths to as few as one of the values permitted in Sec. 5.2.1. In both cases, the length restrictions may affect interoperability with other implementations.

This Recommendation also recognizes two other categories of restrictions:

- Default IV: The length of the IV is restricted to exactly 96 bits.

- GMAC: The plaintext is restricted to the empty string, so that the algorithm in effect generates a stand-alone authentication tag on the AAD with the IV. In this case, the AAD is not the empty string.

Consequently, there are four types of implementations of GCM that may claim conformance with this Recommendation: 1) GCM, 2) GCM with the default IV, 3) GMAC, and 4) GMAC with the default IV.

In the specifications in Sec. 8 below, a value for $P$, $A$, $IV$, or $T$ that fulfills the applicable length restriction is called valid.

For every algorithm that is specified in this Recommendation, a conforming implementation may replace the given set of steps with any mathematically equivalent sets of steps. In other words, different procedures that produce the correct output for any valid input are permitted.

# 7 Mathematical Components of GCM

This section presents the mathematical components that appear in the specifications of authenticated encryption and authenticated decryption in Sec. 8 below. Examples of the basic

operations and functions on strings are given in Sec. 7.1. The incrementing function is defined in Sec. 7.2. An algorithm for "multiplying" blocks is defined in Sec. 7.3. The GHASH function that is constructed from this multiplication is defined in Sec 7.4. The GCTR function is defined in Sec. 7.5.

### *7.1 Examples of Basic Operations and Functions on Strings*

Given a real number $x$, the ceiling function, denoted $\lceil x \rceil$, is the least integer that is not less than $x$. For example, $\lceil 2.1 \rceil = 3$, and $\lceil 4 \rceil = 4$.

Given a positive integer $s$, $0^s$ denotes the string that consists of $s$ '0' bits. For example, $0^8 =$ 00000000.

The concatenation operation on bit strings is denoted ||; for example, 001 || 10111 = 00110111.

Given bit strings of equal length, the exclusive-OR (XOR) operation, denoted $\oplus$, specifies the addition, modulo 2, of the bits in each bit position, i.e., without carries. For example, 10011 $\oplus$ 10101 = 00110.

Given a bit string $X$, the functions $\mathrm{LSB}_s(X)$ and $\mathrm{MSB}_s(X)$ return the $s$ least significant (i.e., right-most) bits and the $s$ most significant (i.e., left-most) bits, respectively, of $X$. For example, $\mathrm{LSB}_3(111011010) = 010$, and $\mathrm{MSB}_4(111011010) = 1110$.

Given a bit string $X$, the bit length of $X$ is denoted len($X$). For example, len(00010)=5.

Given a bit string $X$, the (single) right-shift function, denoted $X >> 1$, is $\mathrm{MSB}_{\mathrm{len}(X)}(0 \,||\, X)$. For example, 0110111 >> 1 = 0011011.

Given a positive integer $s$ and a non-negative integer $x$ that is less than $2^s$, the integer-to-string function, denoted $[x]_s$, is the binary representation of $x$ as a string of bit length $s$ with the least significant bit on the right. For example, for the (base 10) integer 39, the binary representation (base 2) is 100111, so $[39]_8 = 00100111$.

Given a (non-empty) bit string $X$, the string-to-integer function, denoted int($X$), is the integer $x$ such that $[x]_{\mathrm{len}(X)} = X$. In other words, int($X$) is the non-negative integer less than $2^{\mathrm{len}(X)}$ whose binary representation is $X$. For example, int(00011010) = 26.

### *7.2 The Incrementing Function*

For a given block size, the Counter mode encryption within GCM generates a sequence of blocks from an initial block using a particular incrementing function, denoted inc. This incrementing function is defined on a block $X$ as follows:

$$\mathrm{inc}(X) = \mathrm{MSB}_{96}(X) \,||\, \left[\left(\mathrm{int}\left(\mathrm{LSB}_{32}(X)\right) + 1\right) \bmod 2^{32}\right]_{32}.$$

In other words, the incrementing function increments the right-most 32 bits of the block, regarded as the binary representation of an integer, i.e., modulo $2^{32}$; the left-most 96 bits remain

unchanged.

### *7.3 The Multiplication Operation on Blocks*

Let $R$ be the bit string $111000010^{120}$. Given two blocks $X$, $Y$, Algorithm 1 below computes a "product" block, denoted $X \cdot Y$:

Algorithm 1: $X \cdot Y$

*Input*:
blocks $X$, $Y$.

*Output*:
block $X \cdot Y$.

*Steps*:
1. Let $x_0 x_1 ... x_{127}$ denote the sequence of bits in $X$.
2. Let $Z_0 = 0^{128}$ and $V_0 = Y$.
3. For $i = 0$ to 127, calculate blocks $Z_{i+1}$ and $V_{i+1}$ as follows:

$$Z_{i+1} = \begin{cases} Z_i & \text{if } x_i = 0; \\ Z_i \oplus V_i & \text{if } x_i = 1. \end{cases}$$

$$V_{i+1} = \begin{cases} V_i >> 1 & \text{if LSB}(V_i) = 0; \\ (V_i >> 1) \oplus R & \text{if LSB}(V_i) = 1. \end{cases}$$

4. Return $Z_{128}$.

This algorithm corresponds to the multiplication operation for the binary Galois (finite) field of $2^{128}$ elements. The fixed block, $R$, determines a representation of this field as the modular multiplication of binary polynomials of degree less than 128, under the "little endian" convention for interpreting strings as polynomials. For example, if $u$ denotes the indeterminate, then the block $x_0 x_1 ... x_{127}$ corresponds to the polynomial $x_0 + x_1 u + x_2 u^2 + ... + x_{127} u^{127}$. The XOR operation is used to add coefficients of "like" terms during the multiplication. The reduction modulus is the polynomial that corresponds to $R \,||\, 1$. Ref. [3] discusses this field in detail.

### *7.4 The GHASH Function*

Algorithm 2 below specifies the GHASH function that will be called within the algorithms for the GCM authenticated encryption and authenticated decryption functions:

Algorithm 2: $\text{GHASH}_H(X)$

*Prerequisites*:

block $H$, the hash subkey.

*Input*:

bit string $X$ such that len($X$) = 128$m$ for some integer $m$.

*Output*:
block $Y_m$.

*Steps*:
1. Let $X_1, X_2, \ldots, X_{m-1}, X_m$ denote the unique sequence of blocks such that $X = X_1 \| X_2 \| \ldots \| X_{m-1} \| X_m$.
2. Let $Y_0$ be the "zero block," $0^{128}$.
3. For $i = 1, \ldots, m$, let $Y_i = (Y_{i-1} \oplus X_i) \cdot H$.
4. Return $Y_m$.

In effect, the GHASH function calculates $X_1 \cdot H^m \oplus X_2 \cdot H^{m-1} \oplus \ldots \oplus X_{m-1} \cdot H^2 \oplus X_m \cdot H$. Ref. [3] describes methods for optimizing implementations of GHASH in both hardware and software.

The GHASH function is illustrated in Figure 1 below.

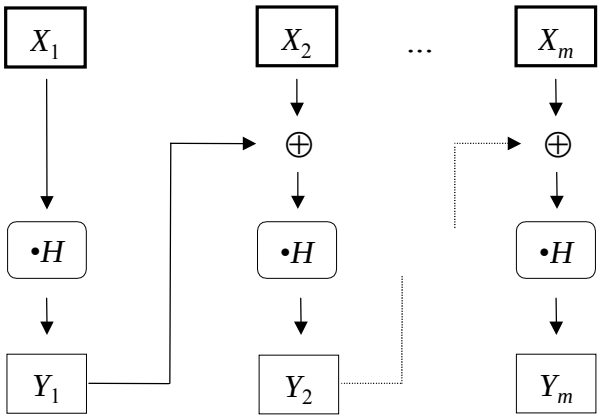


Figure 1: GHASH$_H$ ($X_1 \| X_2 \| \ldots \| X_m$) = $Y_m$.

### *7.5 The GCTR Function*

Algorithm 3 below specifies the GCTR function that will be called within the algorithms for the GCM authenticated encryption and authenticated decryption functions:

Algorithm 3: GCTR$_K$ (*ICB*, *X*)

*Prerequisites*:
approved block cipher CIPH with a 128-bit block size;
key $K$.

*Input*:
bit string $X$, of arbitrary length;
initial counter block *ICB*.

*Output*:

bit string $Y$ of bit length len($X$).

*Steps*:

1. Let $n = \lceil \text{len}(X)/128 \rceil$.
2. Let $X_1, X_2, \ldots, X_{n-1}, X_n^*$ denote the unique sequence of bit strings such that
   $X = X_1 \| X_2 \| \ldots \| X_{n-1} \| X_n^*$ ;
   $X_1, X_2, \ldots, X_{n-1}$ are complete blocks.[2]
3. Let $CB_1 = ICB$.
4. For $i = 2$ to $n$, let $CB_i = \text{inc}(CB_{i-1})$.
5. For $i = 1$ to $n-1$, let $Y_i = X_i \oplus \text{CIPH}_K(CB_i)$.
6. Let $Y_n^* = X_n^* \oplus \text{MSB}_{\text{len}(X_n^*)}(\text{CIPH}_K(CB_n))$.
7. Let $Y = Y_1 \| Y_2 \| \ldots \| Y_n^*$.
8. Return $Y$.

In Steps 1 and 2, the input string of arbitary length is partitioned into a sequence of blocks to the greatest extent possible, so that only the rightmost string in the sequence may be a "partial" block. In Steps 3 and 4, the incrementing function is iterated on the initial counter block input to generate a sequence of counter blocks; the input block is the first block of the sequence. In Steps 5 and 6, the block cipher is applied to the counter blocks and the results are XORed with the corresponding blocks (or partial block) of the partition of the input string. In Step 7, the sequence of results is concatenated to form the output.

Figure 2 below illustrates the GCTR function.

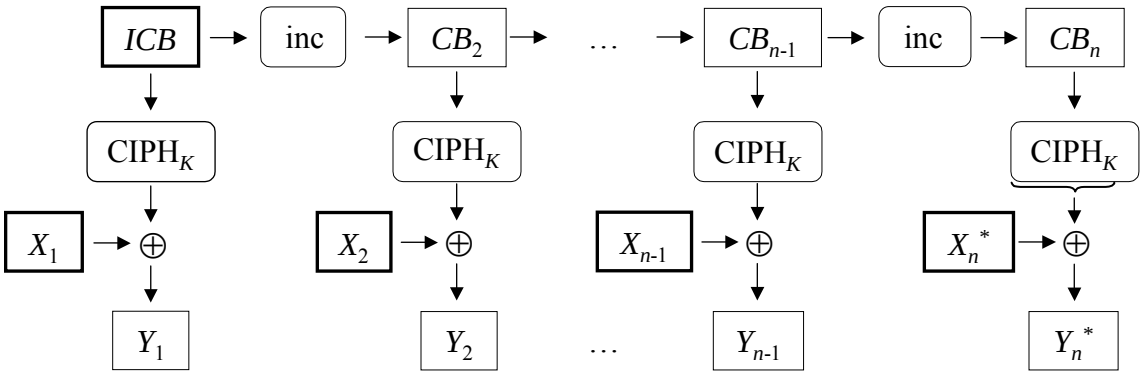


Figure 2: GCTR$_K$ ($ICB$, $X_1 \| X_2 \| \ldots \| X_n^*$) = $Y_1 \| Y_2 \| \ldots \| Y_n^*$.

## 8 GCM Specification

Algorithms for the authenticated encryption and authenticated decryption functions of GCM are specified in Secs. 8.1 and 8.2 below. The specifications include the inputs, the outputs, the steps

[2] Consequently, $X_n^*$ is a either a complete block or a nonempty partial block, and if $1 \le \text{len}(X) \le 128$, then $X = X_1^*$.

of the algorithm, a summary, and a diagram. The inputs that are typically fixed across many invocations of GCM are called the prerequisites. The prerequisites and the other inputs shall meet the requirements in Sec. 5. The suggested notation given in the title of each algorithm includes the key but not the choice of the underlying block cipher.

### *8.1 Authenticated Encryption*

Algorithm 4 below specifies the authenticated encryption function:

Algorithm 4: GCM-AE$_K$ ($IV$, $P$, $A$)

*Prerequisites*:
approved block cipher CIPH with a 128-bit block size;
key $K$;
definition of valid input-output lengths (see Sec. 6);
valid tag length $t$.

*Input*:
valid initialization vector $IV$;
valid plaintext $P$;
valid additional authenticated data $A$.

*Output*:
ciphertext $C$;
authentication tag $T$.

*Steps*:
1. Let $H = \text{CIPH}_K\left(0^{128}\right)$.
2. Define a block, $J_0$, as follows:
   a. If $\text{len}(IV) = 96$, then
      $$J_0 = IV \,||\, 0^{31}1.$$
   b. If $\text{len}(IV) \neq 96$, then
      $$J_0 = \text{GHASH}_H(IV \,||\, 0^s) \qquad \text{where } s = 128 \cdot \lceil \text{len}(IV)/128 \rceil - \text{len}(IV).$$
3. Let $C = \text{GCTR}_K\left(\text{inc}\left(J_0\right), P\right)$.
4. Let $u = 128 \cdot \lceil \text{len}(C)/128 \rceil - \text{len}(C)$ and let $v = 128 \cdot \lceil \text{len}(A)/128 \rceil - \text{len}(A)$.
5. Define a block, $S$, as follows:
   $$S = \text{GHASH}_H\left(A \,||\, 0^v \,||\, C \,||\, 0^u \,||\, \left[\text{len}(A)\right]_{64} \,||\, \left[\text{len}(C)\right]_{64}\right)$$
6. Let $T = \text{MSB}_t\left(\text{GCTR}_K\left(J_0, S\right)\right)$.
7. Return ($C$, $T$).

In Step 1, the hash subkey for the GHASH function is generated by applying the block cipher to the "zero" block. In Step 2, a counter block is generated from the IV. In particular, when the length of the IV is 96 bits, then the padding string $0^{31}1$ is appended to the IV to form the counter block. Otherwise, the IV is padded with the minimum number of '0' bits, possibly none, so that

the length of the resulting string is a multiple of the block size, and the GHASH function is applied to this string to form the counter block. In Step 3, the incrementing function is applied to the counter block that was generated in Step 2 to produce the initial counter block for an invocation of the GCTR function on the plaintext. The output of this invocation of the GCTR function is the ciphertext.

In Steps 4 and 5, the AAD and the ciphertext are each appended with the minimum number of '0' bits, possibly none, so that the bit lengths of the resulting strings are multiples of the block size. The concatenation of these strings is appended with 64-bit representations of the lengths of the AAD and the ciphertext, and the GHASH function is applied to the result to produce a single output block. In Step 6, this output block is encrypted using the GCTR fucntion with the counter block that was generated in Step 2, and the result is truncated to the specified tag length to form the authentication tag. The ciphertext and the tag are returned as the output in Step 7.

The authenticated encryption function is illustrated in Figure 3 below. The determination of $J_0$ from *IV* (Step 2) is not depicted.

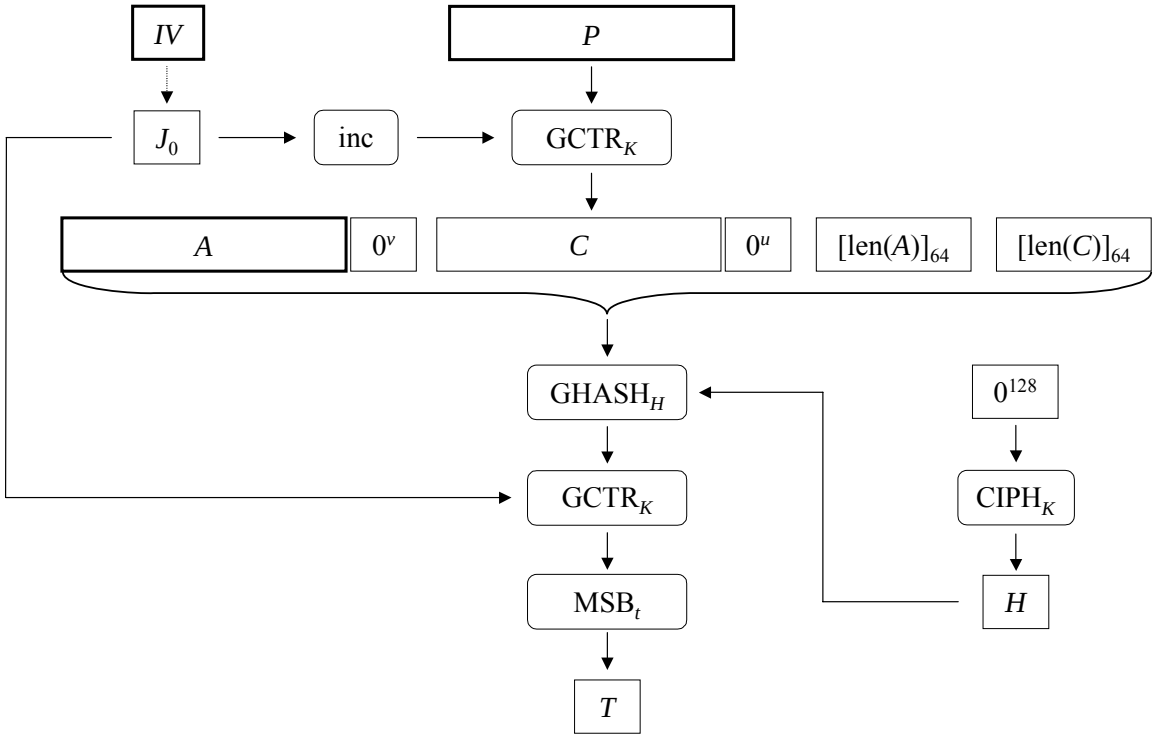


Figure 3 : GCM-AE$_K$ (*IV*, *P*, *A*) = (*C*, *T*).

### *8.2 Authenticated Decryption*

Algorithm 5 below specifies the authenticated decryption function:

Algorithm 5: GCM-AD$_K$ (*IV*, *C*, *A*, *T*)

*Prerequisites*:
approved block cipher CIPH with a 128-bit block size;
key *K*;
definition of valid input-output lengths (see Sec. 6);
valid tag length *t*.

*Input*:
initialization vector *IV*;
ciphertext *C*;
additional authenticated data *A*;
authentication tag *T*.

*Output*:
plaintext *P* or indication of inauthenticity *FAIL*.

*Steps*:

1. If the bit length of *IV*, *C*, or *A* is not valid, or if len(*T*) is not *t*, then return FAIL.
2. Let $H = \mathrm{CIPH}_K\left(0^{128}\right)$.
3. Define a block, $J_0$, as follows:
   c. If $\mathrm{len}(IV) = 96$, then
      $$J_0 = IV \,||\, 0^{31}1.$$
   d. If $\mathrm{len}(IV) \neq 96$, then
      $$J_0 = \mathrm{GHASH}_H(IV \,||\, 0^s) \qquad \text{where } s = 128 \cdot \lceil \mathrm{len}(IV)/128 \rceil - \mathrm{len}(IV).$$
4. Let $P = \mathrm{GCTR}_K\left(\mathrm{inc}(J_0), C\right)$.
5. Let $u = 128 \cdot \lceil \mathrm{len}(C)/128 \rceil - \mathrm{len}(C)$ and let $v = 128 \cdot \lceil \mathrm{len}(A)/128 \rceil - \mathrm{len}(A)$.
6. Define a block, *S*, as follows:
   $$S = \mathrm{GHASH}_H\left(A \,||\, 0^v \,||\, C \,||\, 0^u \,||\, \left[\mathrm{len}(A)\right]_{64} \,||\, \left[\mathrm{len}(C)\right]_{64}\right)$$
7. Let $T' = \mathrm{MSB}_t\left(\mathrm{GCTR}_K\left(J_0, S\right)\right)$.
8. If $T = T'$, then return *P*; else return *FAIL*.

In Step 1, the validity of the lengths of the IV, ciphertext, AAD, and authentication tag is verified. In Step 2, the hash subkey for the GHASH function is generated by applying the block cipher to the "zero" block. In Step 3, a counter block is generated from the IV. In particular, when the length of the IV is 96 bits, then the padding string $0^{31}1$ is appended to the IV to form the counter block. Otherwise, the IV is padded with the minimum number of '0' bits, possibly none, so that the length is a multiple of the block size, and the GHASH function is applied to this string to form the counter block. In Step 4, the incrementing function is applied to the counter block that was generated in Step 3 to produce the initial counter block for an invocation of the GCTR function on the ciphertext. The output of this invocation of the GCTR function is the plaintext that corresponds to the ciphertext for the given IV.

In Steps 5 and 6, the AAD and the ciphertext are each appended with the minimum number of '0' bits, possibly none, so that the bit lengths of the resulting strings are multiples of the block size. The concatenation of these strings is appended with 64-bit representations of the lengths of

the AAD and the ciphertext, and the GHASH function is applied to the result to produce a single output block. In Step 7, this output block is encrypted using the GCTR function with the counter block that was generated in Step 3, and the result is truncated to the specified tag length to form the authentication tag. In Step 8, the result of Step 7 is compared with the authentication tag that was received as an input: if they are identical, then the plaintext is returned; otherwise, *FAIL* is returned.

As noted in Sec. 6, equivalent sets of steps that produce the correct output are permitted. In particular, the verification of the tag may precede the computation of the plaintext.

The authenticated decryption function is illustrated in Figure 4 below. The determination of $J_0$ from *IV* (Step 3) is not depicted.

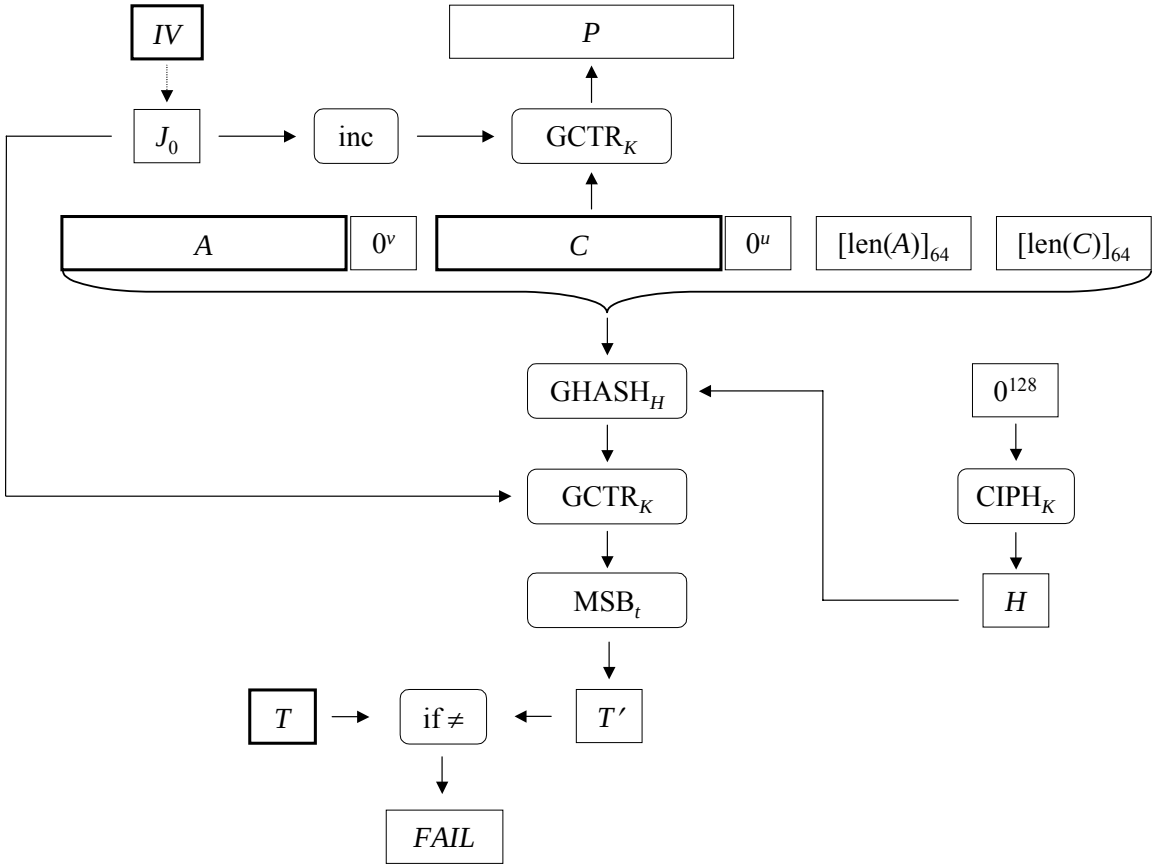


Figure 4: GCM-AD$_K$ (*IV*, *C*, *A*, *T*) = *P* or *FAIL*.

## Appendix A: Authentication Assurance

The creation of an authentication tag during authenticated encryption provides the mechanism whereby assurance of the authenticity of the plaintext and AAD (and IV) can be obtained upon the execution of authenticated decryption. The nature of this assurance depends on the output of authenticated decryption:

- If the output is the plaintext, then the design of the mode provides strong, but not absolute, assurance that the purported source of the data created the tag, i.e., that the plaintext and the AAD (and the IV and the tag) are authentic. Consequently, the mode also provides strong assurance that this information was not subsequently altered, either intentionally or unintentionally.

- If the output is *FAIL*, then it is certain that not all of the given inputs (i.e., the ciphertext, the AAD, the IV, and the tag) are authentic.

In the first case, the assurance is not absolute because forgeries are possible, in principle. In other words, an attacker, i.e., a party without access to the key or to the authenticated encryption function, may be able to produce the correct tag for some triple of ciphertext, AAD, and IV.

As with any tag-based authentication mechanism, if the attacker chooses a $t$-bit tag at random, it will be correct for given data with probability $1/2^t$. With GCM, however, an attacker can choose tags that increase this probability, proportional to the total length of the ciphertext and AAD. In particular, if $n$ denotes the total number of blocks in the encoding (denoted $S$ in Secs. 8.1 and 8.2 above) of the ciphertext and AAD, then a targeted forgery attack is known that would be expected to succeed with probability $n/2^t$. Ref. [1] describes the attack, and Ref [4] gives a detailed treatment of the security properties of GCM.

Of course, an attacker may attempt to systematically guess many different tags for a given input to authenticated decryption, and thereby increase the probability that one (or more) of them, eventually, will be accepted as valid. For this reason, the system or protocol that implements GCM should monitor and, if necessary, limit the number of unsuccessful verification attempts for each key.

Moreover, as with most block cipher modes of operation, the security assurance of GCM degrades as more data is processed with a single key. Therefore, the total number of blocks of plaintext and AAD that are protected by invocations of the authenticated encryption function during the lifetime of the key should be limited. A reasonable limit for most applications would be $2^{64}$.

## Appendix B: Protection Against Replay of Messages

As described in Appendix A, the successful verification of the tag within the authenticated decryption function gives assurance of the authenticity of the data; however, the party that presents the input data to the authenticated decryption function may not be the *original* source of the plaintext and AAD. In other words, GCM does not inherently prevent an attacker from intercepting the output of an invocation of authenticated encryption and "replaying" it for authenticated decryption at a later time, for example, in an attempt to impersonate a party that has access to the key. In some protocols an attacker may even be able to use data that the verifier itself generated earlier in the protocol.

The controlling system or protocol may protect against such an event by monitoring for any duplication of the IVs that are presented for authenticated decryption. Alternatively, certain identifying information can be incorporated into the AAD. Examples of such information include a sequential message number, or a timestamp. Upon successful verification of authenticity, this information may provide a means for the detection of replayed messages, out-of-sequence messages, or missing messages.

## Appendix C: Bibliography


[1] Ferguson, N., Authentication Weaknesses in GCM, Natl. Inst. Stand. Technol. [Web page], http://csrc.nist.gov/CryptoToolkit/modes/comments/, May 20, 2005.

[2] FIPS Publication 197, The Advanced Encryption Standard (AES), U.S. DoC/NIST, November 26, 2001.

[3] D. McGrew, J. Viega, The Galois/Counter Mode of Operation (GCM), Natl. Inst. Stand. Technol. [Web page], http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/, May 31, 2005.

[4] D. McGrew and J. Viega. The Security and Performance of the Galois/Counter Mode (GCM) of Operation. Proceedings of INDOCRYPT '04, Springer-Verlag, 2004. Full paper available from the IACR Cryptology ePrint Archive: Report 2004/193, [Web page], http://eprint.iacr.org/2004/193/, October 7, 2004.

[5] NIST Special Publication 800-38A, 2001 ED, Version 1, Recommendation for Block Cipher Modes of Operation—Methods and Techniques, December 2001, Natl. Inst. Stand. Technol. [Web page], http://www.csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.