Peertechz

ENGINEERING GROUP

TRENDS IN Computer Science and Information Technology
OPEN ACCESS JOURNAL

ISSN: 2641-3086 DOI: https://dx.doi.org/10.17352/tcsit

Review Article

Redefining GCM's resistance to cryptanalysis with offset mechanisms

Muhammad Faisal Nawaz[1] and Yasir Nawaz[2]*

[1]University of Lahore, Pakistan

[2]Shanghai Jiao Tong University, China

Received: 19 February, 2024
Accepted: 29 March, 2024
Published: 30 March, 2024

***Corresponding author**: Yasir Nawaz, Shanghai Jiao Tong University, China, E-mail: my_nawaz@alumni.sjtu.edu.cn, my_nawaz@sjtu.edu.cn

ORCiD: https://orcid.org/0009-0001-3093-4475

Keywords: Authenticated encryption; Galois/Counter Mode (GCM); Differential cryptanalysis offset mechanism; Cryptographic security

https://www.engineegroup.us

Check for updates

Abstract

The research paper proposes an enhancement to the Galois/Counter Mode (GCM) of authenticated encryption by introducing an "offset" mechanism. This modification aims to improve privacy and resist differential cryptanalysis without significantly impacting the mode's efficiency and simplicity. The improved GCM maintains its original features, such as minimal block cipher invocations, the use of a single cryptographic key, and efficient offset computation. It provides a detailed analysis of the operational framework, including the integration and calculation of offsets in encryption and decryption processes. By complicating the predictability of differential cryptanalysis through unique offsets, the paper asserts this enhancement significantly increases GCM's security within a concrete security model. The discussion emphasizes the benefits of the offset-enhanced GCM over other modes, highlighting its suitability for high-speed, parallelizable cryptographic applications while bolstering resistance against cryptanalytic attacks.

Introduction

Authenticated encryption overview

Block cipher mode of operation is a scrutinized cryptographic primitive for secure encryption and decryption that ensures privacy, authenticity, and authenticated encryption [1]. Authenticated encryption is a term that simultaneously provides confidentiality and authenticity to the data. Every cryptosystem requires both forms of security, but until relatively recently confidentiality and authenticity have been designed separately. Now, Authenticated encryption is implemented using a block cipher mode of operation structure. Recently, many authenticated encryption modes have been proposed [2,3]. The first authenticated encryption mode was *IAPM* (Integrity-Aware Parallelizable) mode proposed by Jutla's [4]. The *OCB* (*offset Code Book*) mode which refine one of *IAPM* [5,6]. The *OCB2*, and *OCB3* are refine version of *OCB* mode [7,8]. All of these are parallelized authenticated encryption mode suitable for high-speed cryptosystem [9,10]. There are some others motivated work combined with *Counter* mode and *CBC – MAC* is *CCM* mode uses only one key, however, it is not a suitable for high speed authenticated encryption because *CBC – MAC* is not parallelizable [11]. The other mode similar to *CCM* is *EAX* mode, combined *Counter* mode with *OMAC* [2,12]. The *OMAC* is not parallelized, so, *EAX* is not high speed authenticated encryption mode, but it refines some properties of *CCM* mode. Another authentication encryption mode is *CWC* combined with *Counter* mode with *MAC* based on the universal hash function over $GF(2^{127}-1)$, But it's relatively expensive to implement [13,14]. There is some authenticated encryption mode's ability to authenticate with associated which simultaneously assures the confidentiality and authenticity of data. The method is sometimes termed for *AHED* (authenticated encryption with associated data) [15]. The *CCM* mode and *GCM* mode both have facilities for *AHED* and increase usability [16-18]. The Galois/Counter mode is recommended by the National Institute of Standards and Technology (NIST) and most favorite than *CCM* due to parallelizability [19,20].

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051. DOI: https://dx.doi.org/10.17352/tcsit.000079

Galois/Counter Mode (*GCM*) is a block cipher mode of operation designed to meet the need for confidentiality and authenticity of data and use universal hashing over a binary Galois field. It is implemented in many cryptosystems to achieve high speeds with low latency and low cost. Its design is supported by a well-understood theoretical foundation. There is an enthralling need for a mode of operation that can efficiently provide parallel authenticated encryption. The modes of operation must admit pipelined and parallelized construction and have high data rates. The *Counter* mode meets those requirements and has become a mode for high-speed cryptosystems [21,22]. However, the *Counter* mode provide only confidentiality not message authentication. So *GCM* incorporates with *Counter* mode and builds on it by adding a Message Authentication Code (*MAC*) based on universal hashing provide message authentication that can keep up with our cipher [23]. It uses Polynomial hashing in the finite field $GF(2^n)$ [24]. The multiplication in $GF(2^n)$ can be efficiently implemented using XOR and shift operation. Additionally, *GCM* also has useful properties, it can be used as an incremental *MAC* and stand-alone *MAC*. These properties of *GCM* unique among all of the proposed authenticated encryption modes.

The *GCM* associated *Counter* mode, changes the inputs bits of underlying block cipher serially, and it is well known that the successive block of *Counter* has small hamming difference underlying the block cipher, this led to concern that adversary can obtain many plaintext pairs with a known small plaintext difference, which would facilitate the differential cryptanalysis [25]. It is the responsibility of the mode to compensate for the weak block cipher. Our work refines the privacy property of *GCM* by using an extra input value that is an *offset*, such that each *offset* input is unique.

The principal characteristics of *GCM offset* retain the same (*like:* fully parallelizable) only add a small overhead compared to conventional *GCM* mode. Now, the *GCM offset* combines the following features:

Arbitrary - message length: The *GCM* encrypt and authenticate a nonempty any length of string $M \in \{0,1\}^*$ using $\lceil |M|/n \rceil + 1$ block cipher invocations. The message length (|M|) need to be a multiple of *n*.

Minimal requirement on counter: Like another authenticated encryption mode *GCM* require a *nonce* as *counter*. The *counter* value must be non-repeating (each block cipher chooses a new *counter* value for every message block with restriction no *counter* value used twice).

Offset calculation: We need a sequence of *offset* as with [26,27]. The *offset* value generate in a particularly cheap way, and each *offset* value need just a machine cycle.

Single key: The *GCM offset* used a single block cipher *key*. All the block cipher invocations are keyed by this one *key*.

The paramount contribution of this paper is the introduction of an "offset" mechanism to the GCM of authenticated encryption, aimed at significantly enhancing its resistance to differential cryptanalysis without detracting from its efficiency or simplicity. By integrating unique offsets in the encryption and decryption processes, this enhancement complicates the predictability upon which differential cryptanalysis relies, thereby strengthening GCM's security posture within a robust security model. Through comprehensive analysis and discussion, we demonstrate the practical application of this offset-enhanced GCM in modern cryptographic systems, emphasizing its minimal overhead and retained efficiency. This advancement not only fortifies GCM against sophisticated cryptanalytic attacks but also underscores the feasibility of such an approach in high-speed, parallelizable cryptographic operations, marking a significant stride forward in the domain of authenticated encryption.

Preliminaries

Notation

Let there are two integers *a* and *b*, if $a \leq b$, then it means $\{a, a+1, \ldots, b\}$. If $i > 0$ is an integer, then $ntz(i)$ is the trailing 0 - *bits* in the binary representation of *i*. A string {0,1} represent the set of binary numbers and a string {0,1}* denote the set of all strings. The set $\{0,1\}^n$ denote all the strings of length *n*. If there is no element in the string, then it's called the empty string denoted ε. $A||B$ represents the concatenation of set *A* and *B* where $A,B \in \{0,1\}^*$. If $A \neq \varepsilon$ then firstbit (A) represent the first bit of *A*, in such a way lastbit (A) denote the last bit of the *A*. Let *I* and *n* be two integers then o^i and 1^i represent the string of *o's* and *1's* respectively. If $A \in \{0,1\}^*$ then |A| represent the bit length of A while $||A||_n = \max\{1, \lceil |A|/n \rceil\}$ represent the length of *A* in *n - bit* block. Let $A \in \{0,1\}^*$ and $\tau \in [0..|A|]$ then A[*firstbit* τ] denote first τ bit of *A* and A[*lastbit*] denote the last τ bit of *A* respectively. If $A,B \in \{0,1\}^*$ then $A \oplus B$ is the bitwise XOR of *firstbit* (*A*) and *firstbit* (*B*), where |A| = |B|. If $A = a_{n-1} \ldots a_1 a_o \in \{0,1\}^n$ then *stir2num* is the number $\sum_{i=0}^{n-1} 2^i a_i$. If $a \in [0..\ 2^n - 1]$ then $num2str_n(a)$ is *n - bit* string *A* such that $stir2num(A) = a$. $len_n(A) = num2str_n(|A|)$. If $A = a_{n-1}a_{n-2}\ldots a_1 a_o \in \{0,1\}^n$ then $A << 1$ is the *n - bit* string $a_{n-2}\ldots a_1 a_o o$ which is the left shift of *A* by one bit, while $A >> 1$ is the *n - bit* string $0a_{n-1}a_{n-2}\ldots a_1$ which is the right shift of *A* by *1 - bit*. The plaintext message *M* partitioned into $m_1 m_2 \ldots m_n$ and $|m_i| = n$ *for* $1 < I < n$. We partition *C* into $c_1 c_2 \ldots c_n$, where C refers to the ciphertext resulting from the encryption process. The partitioning of *C* into multiple blocks $c_1 c_2 \ldots c_n$ facilitates the processing of the ciphertext in blocks, aligning with the block cipher mode of operation used by GCM. This is crucial for both encrypting the plaintext message in blocks and subsequently or generating or verifying the authentication tag, which ensures data integrity and authenticity.

The field with 2^n points

Lets $GF(2^n)$ represent a field with 2_n point [28]. We interchangeably think of a point *a* in $GF(2^n)$ in any of the following ways:

1. As an abstract point in a field
2. As an *n - bit* string $a_{n-1} \ldots a_1 a_o \in \{0,1\}^n$
3. As a formal polynomial $a(x) = a_n x^{n-1} + a_1 x + a_o$ with binary coefficients.

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

4. As an integer between 0 and $2^n - 1$, where the string a $\in \{0,1\}^n$ corresponding to the number *str2num(a)*.

We write $a(x)$ instead of a if we wish to emphasize that we are thinking of a as a polynomial. We take XOR to add two points in $GF(2^n)$, and for the multiplication of two points, we fix an irreducible polynomial $p_n(x)$ having binary coefficients and degree n. For $n = 128$ the indicated polynomial is $p_{128}(x) = x^{128} + x^7 + x^2 + x + 1$. A few other $p_n(x)$ values are $x^{64} + x^4 + x^3 + x + 1$ and $x^{96} + x^{10} + x^9 + x^6 + 1$ and $x^{160} + x^5 + x^3 + x^2 + 1$ and $x^{192} + x^7 + x^2 + x + 1$ and $x^{224} + x^9 + x^8 + x^3 + 1$ and $x^{256} + x^{10} + x^5 + x^2 + 1$. To multiply a,b $GF(2^n)$ represent a and b as polynomial $a(x) = a_{n-1}x_{n-1} + \cdots a_1x + a_o + a_1x + a_o$ and $b(x) = b_{n-1}x_{n-1} + \cdot\cdot b_1x + b_o + b_1x + b_o$, form product $c(x)$ over $GF(2)$. When dividing $c(x)$ by $p_n(x)$ it takes a reminder. The multiplication of x and a $\in \{0,1\}^n$ computationally simple. When $n = 128$, in this case multiplying a = $a_{n-1} + \cdots + a_1 + a_o$ by x give $a_{n-1}x^n + a_{n-2}x^{n-1} + \cdot\cdot + a_1x^2 + a_ox$. If *first bit* of a is o, then $a.x << 1$. When *first bit* of a is 1, then add x^{128} to $<< 1$. Since $p_{128}(x) = x^{128} + x^7 + x^2 + x + 1 = o$, in such a way $x^{128} + x^7 + x^2 + x + 1$. So, add x^{128} means to XOR by 0^{120} 10000111. So, when $n = 128$,

$$a.x = \begin{cases} a \ll 1, \ if \ \ firstbit\left(a\right) = 0 \\ (a \ll 1) \oplus 0^{120}10000111, \ if \ \ firstbit\left(a\right) = 1 \end{cases} \quad (1)$$

On the other hand, in the case of divide a $\in \{0,1\}^n$ by x, if the *last bit* of a is o, then $a.x^{-1}$ is a $>>1$. In such a way, if the *last bit* of a is 1, then XOR to a $>>1$ the value x^{-1}. Since $x^{128} + x^7 + x^2 + x + 1$ we know that $x^{-1} = x^{127} + x^6 + x + 1 = 10^{120}$ 1000011. So, when $n = 128$,

$$a.x^{-1} = \begin{cases} a \gg 1, \ if \ lastbit\left(a\right) = 0 \\ (a \gg 1) \oplus 10^{120}1000011, \ if \ lastbit\left(a\right) = 1 \end{cases} \quad (2)$$

If $L \in \{0,1\}^n$ and $i \geq -1$, then $L(i) = L.x^i$. so, we can compute from L the values $L(-1), L(o), L(1), ..., L(\mu)$, where μ is a small number.

Gray codes

Gray code is a sequence of $\gamma^l = (\gamma_0^l\gamma_1^l \cdots \gamma_{2^l-1}^l)$ of $\{0,1\}^l$, where $l \geq 1$ and successive points just one bit differ. When n is a fixed number *GCM* use canonical gray code $\gamma = \gamma^l$ from $\gamma^1 = (0\ 1)$. So, for $l > o$,

$$\gamma^{l+1} = (0\gamma_0^l\ 0\gamma_1^l \ldots 0\gamma_{2^l-2}^l\ 0\gamma_{2^l-1}^l\ 1\gamma_{2^l-1}^l\ 1\gamma_{2^l-2}^l \ldots 1\gamma_1^l\ 1\gamma_0^l) \quad (3)$$

Thus, γ is a gray code, for computing successive points,

$$1 \leq i \leq 2^n - 1, \ \ \gamma_i = \gamma_{i-1} \oplus (0^{n-1}1 \ll ntz(i))$$

Let $L \in \{0,1\}^n$ and $\gamma_1.L, \gamma_2.L, \gamma_3.L, ..., \gamma_m.L$ are considered the problem of successive forming strings. Thus, $\gamma_1.L = 1.L = L$. Since $\gamma_{i-1} = \gamma_{n-1} \oplus (on\text{-}1\ 1 << ntz(i))$ we know that,

$$\gamma_i.\mathrm{L} = \gamma_{i-1} \oplus \left(0^{n-1}1 \ll ntz\left(i\right)\right).\mathrm{L} \quad (4)$$

$$\gamma_i.\mathrm{L} = (\gamma_{i-1}.\mathrm{L}) \oplus (0^{n-1}1 \ll ntz(i)) \quad (5)$$

$$\gamma_i.\mathrm{L} = (\gamma_{i-1}.\mathrm{L}) \oplus (\mathrm{L}.\mathrm{x}^{ntz(i)}) \quad (6)$$

$$\gamma_i.\mathrm{L} = (\gamma_{i-1}.\mathrm{L}) \oplus \mathrm{L}(ntz\left(i\right)) \quad (7)$$

The ith word can be obtained by xoring $L(ntz(i)$ with previous words. The ith word would be obtained in the same way for $I \geq 2$ *e.i* $\gamma_1.\mathrm{L} \oplus \mathrm{R}, \gamma_2.\mathrm{L} \oplus \mathrm{R}$ The first word in the sequence is L ⊕ R instead of L.

GCM offset

This section describes the complete definition of GCM with additional input offset for 128 – bit block ciphers. Generally, GCM encryption have the following inputs, each of which is a bit string:

- A plaintext M, partitioned into $m_1m_2...m_n$ and length of each message block exact multiple of a block cipher.
- Authenticated data, which is denoted as AD. This data just authenticates but does not encrypt.
- Secret key K, whose length is multiple of a block cipher.
- The Counter value, all that is expected of the *Counter* is that it be as a nonce. it is not required to be random or unpredictable.
- The offset (z_i) for each block cipher, such that each z_i is unique.

Each different value of Counter produces a different set of z_i. Thus each offset XOR with the corresponding counter value produces an unpredictable value (comparable, to nonrandom and predictable nonce-related counter value) for the underlying block cipher. The calculation of z_i is summarized in the following equations.

$$L_0 = L = E_k\left(0^n\right) \ where \ 0^n \ is \ consist \ of \ n \ zero \ bits. \quad (7)$$

$$R = E_k\left(ctr + i \oplus L\right) \quad (8)$$

$$L_i = 2.L_{i-1}\ 1 \leq i \leq m \quad (9)$$

$$Z\left(1\right) = R \oplus L \quad (10)$$

$$z_i = z_{i-1} \oplus L(ntz(i))\ 1 \leq i \leq m \quad (11)$$

Offset calculation

Initialization of L_0: The document describes that the initial value L_0 is derived by encrypting a block of n zero bits using the block cipher encryption function E_k under the secret key K. Mathematically, it's represented as $L_0 = L = E_k(o_n)$ where o_n denotes a string of n zero bits.

Calculation of R: The value R is computed as $R = E_k(ctr + i) \oplus$

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

L, where *ctr* is the counter value used in the encryption process, and i is an incrementing value for each block to ensure that R is unique for every block of data being processed.

Sequential Calculation of L_i: For $i \geq 1$, each L_i is computed by doubling the previous L_{i-1} in the finite field $GF(2^n)$. This operation can be efficiently implemented using shift and conditional XOR operations to account for the field's polynomial representation.

Generation of Z_i: The first offset value Z_i is simply $R \oplus L$, combining the previously computed R and L values. Subsequent Z_i values for $i > 1$ are derived by XORing Z_{i-1} with L shifted by the number of trailing zeros in i (noted as $ntz(i)$). This is represented as $Z_i = Z_{i-1} \oplus L.(ntz(i))$

Usage in encryption and decryption

Encryption: During the encryption process, the offset values Z_i are XORed with the counter values before they are encrypted with the block cipher under the key k. This step generates a unique keystream for each block, which is then XORed with the plaintext blocks to produce the ciphertext. Specifically, if Y_i represents the encrypted counter (plus offset) blocks, then $C = M \oplus (Y_1 || Y_2 || Y_3 || \cdots)$, where M is the plaintext message.

Decryption: For decryption, the same process is mirrored. The offsets Z_i are recalculated in the same manner as during encryption and used to generate the keystream by XORing with the counter values and encrypting the result under k. The ciphertext is then XORed with this keystream to recover the plaintext.

The operator "." refers to multiplication over the finite field $GF(2^n)$, *GCM* use $G(2^{128})$ defined in section 2. The operator $ntz(i)$ is the number of trailing o - *bits* in the binary representation of i such as L_i can be computed with $ntz(i)$ defined in section 2. Equivalently, $ntz(i)$ is the largest integer z such that 2^z divides i. However, authenticated encryption takes these inputs and resulting a ciphertext whose length exactly that of the plaintext and a tag T whose length also be the same. The length of the tag T denoted as t. The authenticated encryption of *GCM* with extra input Z_i shown in Figure 1. The authenticated decryption operation has an extra input than authenticated encryption that is tag T and output, either the plaintext or *fail*. The symbol *fail* indicates that the inputs are not authentic. The authenticated data *AD* is used to protect the message that needs to be authenticated and does not need to be encrypted. When *GCM* used for secure network protocol, the *AD* includes protocol version numbers, ports, sequence numbers, addresses, and other fields that indicate how communication should be handled, processed, or forwarded. When the length of M is zero, *GCM* acts as a *MAC* on the *AD*. The mode of operation that uses *GCM* as a stand-alone *MAC* is denoted as *GMAC*. The strength of the authentication is determined by the length of tag t, and t must be fixed for any fixed value of the key. The length of t must be at least *64 bit*, whenever possible *128 bit* should be used because this length provides the best security.

The plaintext consists of a sequence of n - *bit* strings (m_1, m_2, ..., m_{n-1},m_n) that is called a data block, and the bit length of each data block is*128 bit*. Although the bit length of the last data block may not be equal to n *bit*, so we denoted the *bit* length of the last block by u, where $1 \leq u \leq 128$. Similarly, the corresponding ciphertext block is denoted as c_1, c_2, ..., c_{n-1}, c_n, where the *bit* length of the last block is u. The authenticated data block *AD* denoted as AD_1, AD_2,..., AD_{n-1}, AD_n, where the *bit* length of AD_n may not be a complete block, so the length of the last block we denote as v, where$1 \leq v \leq 128$. In the following given equation, we can define the authentication encryption operation.

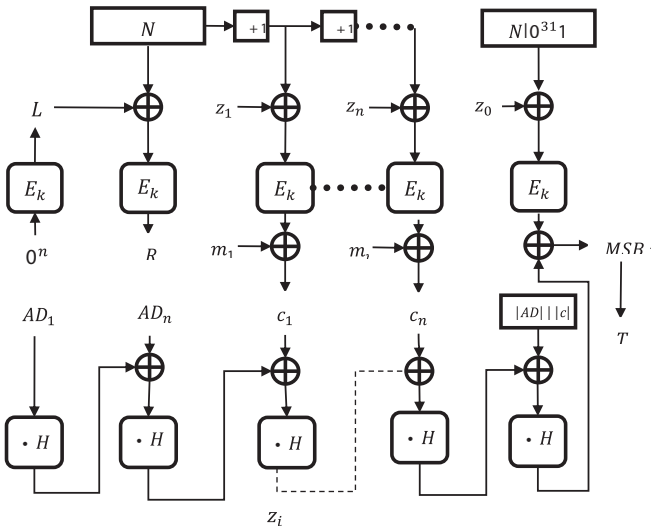

Figure 1: GCM authenticated encryption.

algorithm $GCM_k^{ctr,A}(P)$

algorithm $GCM_k^{ctr,A}(P)$

if $|N| = 96$ **then** $ctr \leftarrow N \backslash 0^{31}1$

else $ctr \leftarrow GHASH_K\left(N0^i |N|_{128}\right)$ for minimal $i \geq 0$

$m \leftarrow |M| / 128;$

for $i \leftarrow 0$ *to* m **do** $Y_i = E_K(ctr + i) \oplus z_i$

$C \leftarrow M \oplus (Y_1 \backslash Y_2 \backslash Y_3 \backslash \ldots)$

$X \leftarrow AD \backslash 0^i \backslash C \backslash 0^j \backslash |AD|_{64} \backslash |C|_{64}$ *for minimal* $i, j \geq 0$

$Tag \leftarrow Y_1 \oplus GHASH_H(X)$

$T \leftarrow MSB_\tau(Tag)$

return C||T

algorithm $GHASH_H(X)$

$X_1 \ldots X_n \leftarrow X$ *where* $|X_i| = 128$

$Y \leftarrow 0^{128}$; *for* $i \leftarrow 1$ *to* m *do* $Y \leftarrow (Y \oplus X_i) \cdot H$

Return y

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

The successive value of the counter (*ctr*) generated by using *incr* () function, which treat 32 *lsb* (least significant bit) and increment with *modulo* 2^{32}. The authentication decryption process of *GCM* similar to the authentication encryption process, rather than the hash and encrypt step, both are reversed in authentication decryption.

Generally, additional authenticated data (*AD*) and blocks of plaintext ($m_1 m_2 \ldots m_{n)}$ is shown. Here E_k denotes the block cipher encryption using the *key K*, • denotes multiplication in $GF(2^{128})$ by the *hash key H*, and + 1(*incr*) denotes the counter increment function.

Security proof

The block cipher is a function $E{:}K \times \{0,1\}^n \rightarrow \{0,1\}^n$ and if it is assumed to be a secure pseudorandom permutation (*PRP*) [29], then *GCM* stand a secure authentication encryption, where *k* is a finite set $EK(\cdot) = EK(,\cdot)$ is a permutation on $\{0,1\}^n$. This requirement is met when *E* cannot be distinguished from a random permutation (*R*) by an adversary that can choose its inputs and view its outputs. The block cipher *E* with a fixed key and *permutation oracle*, both have the same interface. Let *A* be adversary given access to *permutation oracle* to determine whether it is *R* or *E* with random selected key. the probability in each case is 1/2. The adversary asks sequence of queries to the oracle and wants to guess whether the response is. Let *D* be the event that it guesses the *E*, in such a way, D^R denote the guess of *R*. Then finally we can conclude advantage function of adversary *A*.

$$Adv = P[D\backslash E] - P[D\backslash R] \tag{12}$$

The notation *P*[*X*] denotes the probability of event *X*. The $P[X|Y] = P[X \cap Y] / P[Y]$ denotes the probability of event *x* given event *Y* equals the probability of event *Y* and event *X* divided by the probability of event *Y*. We make an assumption that advantage $Adv > o$, thus the range of *Adv* between *o and 1*. The *AEAD* of *GCM* follows the following security model [15]. It has the following input bit strings: *M, counter, AD, and* Z_i and return *C and T. authenticated decryption oracle* models take *counter, AD,* z_i, *C,T* an input and return *M* or special symbol *fail*.

According to the definition of privacy (confidentiality), we use the indistinguishability of ciphertext from random under a *CPA attack* and indistinguishability of plaintext from random under a *CCA attack*, this definition equivalent to [30]. *GCM* encryption is secure under these assumptions when adversary presented with these oracles cannot tell if they contain *GCM* with a randomly selected key (E_{GCM}) or if *C* and *T* are a random function of inputs E^R_{GCM}. The probability in each of these cases is 1/2. Generally, the value of *H* for both (computing the authentication *tag* and hashing) provides the adversary a potential attack vector against privacy. So, we need to give access to the adversary to the *authenticated decryption oracle*. *GCM* takes *E* as a pseudorandom function *PRF*. For security of *PRF*, consider, we give access to the *function oracle*, and guess whether it contains *PRF* or a true random function (oracle have the same interface) [31,32]. The advantage of *PRF distinguisher* are following.

$$Adv_{PRF} = P\left[D|E_{PRF}\right] - P\left[D|E^R_{PRF}\right] \tag{13}$$

Where, E_{PRF} and E^R_{PRF} denoted corresponding to *PRF* and *random function*. Advantage against both *PRF* and *PRP* are similar, because having similar properties.

Lemma 1

The advantage Adv_{PRF} *of* an adversary in distinguishing *a n bit PRP E* from a random function is bounded by $Adv_{PRF} \leq Adv_E + q(q-1)2^{-n-1}$ where Adv_E is the adversary's advantage in distinguishing *E* from a

ramdom permutation, and a value *q is the number of queries* to the *function oracle*.

Theorem 1

If there is an adversary that can distinguish *GCM* encryption from a random function with advantage Adv_{GCM}, when the output of that function is limited to *q* queries to the authenticated encryption and decryption oracles, where the total number of plaintext bits processed is l_p and where *len(C) + len(Adv)* ≤ *l and len(ctr)* ≤ len_{ctr} for each query, then that adversary can distinguish *E* from a random permutation with advantage Adv_E, where

$$Adv_E \geq Adv_{GCM} - \left(\frac{l_P}{n} + 2q\right)^2$$
$$2^{-n-1} - q\left(\frac{l_P}{n} + 2q\right)\frac{l_{ctr}}{n} + 12^{1-n} + \frac{l}{n} + 12^{-t}) \tag{14}$$

The formulation for the adversary's ability to distinguish *E* (the block cipher encryption function) from a random permutation with an advantage Adv_E is derived from a theoretical framework for evaluating the security of cryptographic algorithms. Specifically, in the context of the Galois/Counter Mode (GCM) and its improved versions discussed in the document, the advantage Adv_E quantifies the effectiveness of an adversary in distinguishing the encryption function *E* used within GCM from a perfectly random permutation. This measure is crucial in cryptographic security to assess how well the encryption scheme withstands attempts at cryptanalysis.

The formulation for Adv_E is based on several factors, including:

The Number of Queries (*q*): This represents the number of times the adversary is allowed to interact with the encryption oracle (or the block cipher being analyzed) and observe its outputs. A larger number of queries might increase the adversary's chances of distinguishing *E* from a random permutation, up to a certain limit.

The Total Number of Plaintext Bits Processed (*lp*): This is the cumulative length of all plaintext messages that

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

the adversary encrypts using the oracle. It factors into the adversary's advantage because processing a large volume of data might reveal patterns or weaknesses in the encryption scheme.

Constraints on the Counter Values and the Size of the Authentication Tag (*t*): Constraints on the length of the counter and the size of the authentication tag also influence the adversary's advantage. For instance, a shorter authentication tag might be easier to forge or guess, potentially increasing Adv_E.

Security Bounds of the Underlying Block Cipher: The inherent security of the block cipher itself, against both known and unknown attacks, plays a critical role. The stronger the block cipher, the lower the adversary's advantage in distinguishing it from a random permutation.

The specific formulation of Adv_E provided in the document considers these and potentially other factors, such as the parallelizability of the GCM mode and its resistance to specific types of cryptanalytic attacks (e.g., differential cryptanalysis). The goal of such a formulation is to establish a concrete security proof or bound that quantifies the level of security offered by the encryption scheme against an adversary capable of conducting chosen plaintext attacks (CPA) or ciphertext attacks (CCA).

GCM encryption security also depends on the authentication *tag size* but it's relatively weak. In the bound on Adv_E contain 2^{-t} not dominate that value *as long as t* is greater than about $n - \lg(q\frac{l}{n} + \frac{l_{ctr}}{n})$.

In the presence of *CPA attack* for *MAC* security we use the standard model and give access to the adversary to *tag generation oracle* and *tag verification oracle*. The adversary sends messages to the *tag generation oracle tag generation oracle* and construct any message pairs and send these to the *tag verification oracle*. After making the *q* queries to both of *oracles*, the probability of the adversary getting *verification oracle* to accept a message pair other than generated by the *tag generation oracle*. This is the forgery advantage of GCM (F_{GCM}).

Theorem 2

Adversary with F_{GCM} against *GCM* has Adv_E against *pseudorandom permutation E* used in *GCM* are.

$$F_{GCM} - \left(\frac{l_P}{n} + 2q\right)^2 \cdot 2^{-n-1} - q\left(\left(\frac{l_P}{n} + 2q + 1\right)\frac{l_{ctr}}{n} + 12^{1-n} + \frac{l}{n} + 12^{-t}\right) \quad (15)$$

Result and discussion

In this section, we discuss the characteristics of *GCM offset* with respect to another authenticated encryption mode. Our proposal extends by feature to the NIST recommended *GCM* mode. *Counter* mode is the obvious choice for the foundation of most of *authenticated encryption* mode since it is the one simple, efficient and well-known privacy-only mode that is fully parallelizable. It comes with a security proof that guarantees no attacks up to the birthday bound and has been proven secure against *CPA attack* up to $2^{n/2}$ encrypted data blocks. We refine the *Counter* mode with small additional overhead which is known as the *offset*. The *offset* is an unpredictable input underlying the block cipher and it led to achieving higher resistance against differential cryptanalysis and improved the security of *Counter* mode without breaking its important advantages. So that we can improve the security of *Counter* mode related authenticated encryption mode like *XCBC, CCM, EAX, and cwc.* In this paper, we refine the *GCM* with unpredictable *offset* input. Thus, the desirable characteristics of *offset* associated *GCM* are the following:

- Assuming underlying block cipher is a good *PRP* and authenticated *tag length t* equal to the block length *n* then *GCM offset* provable secure up to birthday bound.
- When encrypting the plaintext and getting the corresponding ciphertext, then we have the same length of the plaintext and authentication tag *t*.
- We used nonce (each value used at most once in a given session, having the property of counter); it is not required to be random or unpredictable.
- The *offset* is unpredictable input underlying the block cipher, which is XOR with corresponding counter value, each input value of *offset* used at most once in a given session.
- *GCM offset* use the forward direction of the block cipher. This saves chip area compared to AEAD constructions. So, for *AHED descryption* require the block cipher backward direction.
- *GCM offset* is fully parallelizable, enabling hardware throughput. Which is not limited by benefiting software embodiments and block cipher latency.
- The authenticity of the data after decryption can be verified from the recovery of the confidential data. The invalid data cannot be processed without counter mode decrypting them.
- The confidential portion of *GCM* is a counter mode with extra input that is *offset* is a simple and efficient for hardware to construct *GF*(2^{128}) *multiplier.* Overall, in hardware, *GCM* is unrivaled by any *authenticated encryption* scheme.
- As well as it can also be efficient for software. *GCM* is online and no one needs to know the message length in advance of processing it. However, need to know the *AD* and its length before processing the message. This makes the *GCM* suitable for networking applications and

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

incremental *API* (Application Programmers Interface) where, *M*, *C*, or *AD* provided incrementally in chunks.

The characteristics of fully parallelizable authenticated encryption modes of operation are summarized in Table 1. The remaining serial modes are described in Table 2. The modes that come with security proof are based on the assumption that the underlying block cipher is secure. The confidentiality and authenticity of each mode proved together with the fact that no attacker can get a significant advantage to distinguish between a random stream and ciphertext. There are some provably secure modes, and some are not proven both characteristics mentioned in Tables 1,2.

Some characteristics definitions are the following:

- **Patent:** Provably secure authentication encryption modes are patent (*i.e GCM*, *OCB*, *XCBC*, *IAPM*), and some modes trying to e patent aware.
- **Provably Secure:** If the underlying block cipher is a secure *PRP* and modes come with the proof of security and give message privacy and authenticity then modes are known as provably secure.
- **Parallelizability:** For a high-speed environment we use parallelize mode, where encryption/decryption can be done parallel (Table 1). In the case of parallelizable authenticated encryption modes, both (encryption and authentication) are parallelizable denoted as *A* + *E*.
- **Associated data authentication:** The unencrypted data that is used for protection of ciphertext, where authenticated data is denoted as *AD*. The *AD* typically used is to encode header information in a networking context.
- **Ciphertext Expansion:** Many modes of operation expand message up to *authentication tag length*, so for the short message this property is important where can overcome a length of the original message.
- **Online message processing:** this is an important property for memory memory-restricted environment, where the possibility to encrypt or decrypt a message without obtaining all messages, *GCM* have this property.
- **Endian dependency:** the modes of operation that use the integer *multiplication/ addition* are endian dependent. All the discussed (in this section) modes are endian dependent other than *OCB* mode.
- **Incremental *MAC*:** In the application data set frequently changes and must be authenticated remote database or recalculating an authenticator for all data cannot be efficient.

Table 1: Properties of fully parallelizable modes.

Feature	IAPM	OCB	CWC	CS	XCBC	GCM
Parallelizability	E+A	E+A	E+A	E+A	E+A	E+A
Patent	Yes	Yes	No	No	Yes	Yes
Provably Secure	Yes	Yes	Yes	No	Yes	Yes
Ciphertext Expansion	*0 … n + n*	τ	τ	*0 … n + n*	*0 … n(L+1) *n*	τ
Keying Material	2 keys	1 keys	1 keys	1 keys	2 keys	1 keys
Online	Yes	Yes	Yes	Yes	Yes	Yes
Endian Dependent	Yes	No	Yes	Yes	Yes	Yes
Incremental MAC	No	No	No	No	Yes	Yes
Error Propagation	No	No	No	No	Yes	No
Two-pass	No	No	Yes	No	No	Yes
Authenticator length	*n*	*0 … n*	*0 … n*	*n*	*(L+1) *n*	*0 … n*
Only encrypt engine	No	No	Yes	No	No	Yes
Associated Data	No	No	Yes	No	No	Yes

Table 2: Properties of serial modes.

Feature	CCM	EAX	PCFB	XCBC
Parallelizability	E only	E only	No	No
Patent	No	No	N/A	No
Provably Secure	Yes	Yes	No	Yes
Ciphertext Expansion	16k, k ∈ {0 … 8}	τ	$\left(\frac{128}{j}\right)*n$	*0 … n + n*
Keying Material	*1 keys*	*1 keys*	*1 keys*	*2 keys*
Online	No	Yes	No	Yes
Endian Dependent	Yes	Yes	Yes	Yes
Incremental MAC	No	No	No	No
Error Propagation	No	No	Yes	Yes
Two-pass	Yes	Yes	No	No
Authenticator length	16k, k ∈ {0, …, 8}	*0 … n*	$\left(\frac{128}{j}\right)$	*n*
Only encrypt engine	Yes	Yes	Yes	No
Associated Data	Yes	Yes	No	No

Differential Distribution Table (DDT)

The Differential Distribution Table (DDT) is a crucial tool in differential cryptanalysis, as it maps the difference between two inputs to the difference between the corresponding outputs for each possible input pair. DDT is used to identify differential characteristics with high probabilities that can be exploited in attacks. The probability of a differential characteristic, which can be derived from the DDT, is a measure of how likely it is that a specific input difference will lead to a specific output difference after going through the cipher.

The implementation of an offset in the improved GCM aims to make the prediction of output differences harder by introducing an additional layer of unpredictability into the encryption process. By XORing each block cipher input with a unique offset, the improved scheme aims to disrupt the predictability that differential cryptanalysis relies on. This unpredictability complicates the construction of a DDT with high probability differential characteristics that are useful for an attacker.

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

Results of attacks and complexity

In terms of the results of attacks against the improved GCM, including the number of rounds of the block cipher attacked and the complexity of these attacks, such specifics would typically result from extensive cryptanalytic research. The document mentions improvements to privacy and a theoretical resistance to differential cryptanalysis but does not provide detailed results of attacks, such as specific numbers of rounds that can be securely encrypted or the exact complexity of potential attacks against the improved mode.

In general, the resistance of a cryptographic algorithm or mode of operation to differential cryptanalysis (or any other form of cryptanalysis) is evaluated based on:

The Number of Rounds: More rounds generally increase security against differential cryptanalysis, as they make it more difficult to find useful differential paths that cover the entire cipher.

Data Complexity: This refers to the amount of plaintext-ciphertext pairs an attacker needs to analyze to successfully exploit a differential characteristic. The introduction of offsets aims to increase the data complexity required for a successful attack.

Attack Complexity: This encompasses both the computational resources and the data required for an attack to be feasible. Ideally, the complexity should be close to or exceed brute-force search complexity, making the attack impractical.

For detailed cryptanalytic results, including specific vulnerabilities and the resistance of the improved GCM to differential cryptanalysis, one would look to specialized cryptographic literature and research that conducts a thorough analysis of the scheme, including practical and theoretical attacks. Without explicit results in the provided document, it's recommended to consult further cryptographic analysis and peer-reviewed research for a comprehensive understanding of the improved GCM's resistance to differential cryptanalysis and other cryptographic attacks.

Furthermore, the improved GCM "offset" mechanism is designed to enhance the mode's privacy and resistance to differential cryptanalysis. This modification is pivotal in the realm of authenticated encryption, where the quest for both robust security and high performance is incessant. To appreciate the value brought by the improved GCM, it's essential to compare it with other authenticated encryption modes such as CBC-MAC (Cipher Block Chaining Message Authentication Code) and CCM (Counter with Cipher Block Chaining-Message Authentication Code), focusing on their security features and performance metrics.

Starting with the core of its enhancement, the improved GCM integrates an offset into the encryption process, which is a strategic move to complicate the predictability that differential cryptanalysis exploits. This means that for each block encrypted, a unique offset is applied, significantly obstructing the ability of an attacker to use differential techniques to infer key information or plaintext. This addition does not notably impact the operational efficiency of GCM, which is renowned for its parallel processing capabilities. The ability to process multiple encryption and authentication operations in parallel is a crucial determinant of performance in high-speed network environments, making the improved GCM exceptionally well-suited for applications requiring rapid data processing without compromising security.

On the other hand, CBC-MAC, an older mode of authenticated encryption, employs a sequential block cipher operation to provide message integrity and authenticity. While CBC-MAC is fundamentally secure under certain conditions, its security model is contingent upon the proper management of keys and initialization vectors. Specifically, if a key is reused across different sessions or improperly managed, the security of CBC-MAC can be compromised, making it susceptible to forgery attacks. Furthermore, the inherent sequential processing of CBC-MAC limits its throughput and efficiency, especially in comparison to modes like GCM that excel in parallel processing.

CCM mode, another contender in the realm of authenticated encryption, combines the Counter mode of encryption with CBC-MAC for authentication. This dual approach necessitates a unique nonce for each message to ensure security, introducing complexities in nonce management that can be problematic in systems where nonce reuse might occur. Additionally, CCM operates in two passes over the data—one for authentication and one for encryption—which inherently doubles the processing requirement for any given message. This two-pass process significantly affects performance, particularly in systems where latency and throughput are critical factors.

Comparatively, the improved GCM, with its offset mechanism, not only enhances security by thwarting differential cryptanalysis but also maintains high performance through its parallelizable architecture. This unique blend of security and efficiency is not as pronounced in CBC-MAC and CCM. The sequential nature of CBC-MAC's operation and CCM's two-pass requirement for encryption and authentication translate into inherent performance bottlenecks. These limitations become increasingly significant in the context of high-speed data transmission and processing, where delays, even milliseconds in length, can be detrimental.

In terms of security, the improved GCM's offset mechanism offers a tangible advantage by increasing the complexity for attackers attempting to leverage cryptanalysis techniques. Unlike CBC-MAC, where security can be undermined by key management issues, or CCM, which requires stringent nonce management to avoid security pitfalls, the improved GCM provides a robust security model that is less susceptible to such operational hazards. This makes the improved GCM a more resilient choice for environments where the integrity and confidentiality of data are paramount, and where the operational context might not always guarantee perfect key or nonce management.

From a performance standpoint, the improved GCM's ability to leverage parallel processing stands in stark contrast

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

to the inherently sequential CBC-MAC and the two-pass CCM mode. This architectural advantage enables the improved GCM to achieve higher throughput rates and lower latency, making it exceptionally well-suited for high-performance computing environments, real-time applications, and large-scale data processing scenarios. Furthermore, the minimal overhead introduced by the offset mechanism ensures that the improved GCM maintains its performance advantages without incurring significant computational costs.

In conclusion, while CBC-MAC and CCM have played pivotal roles in the development of authenticated encryption, the advent of the improved GCM with its offset mechanism signifies a leap forward in both security and performance. The enhanced resistance to differential cryptanalysis, combined with the ability to execute encryption and authentication operations in parallel, positions the improved GCM as a superior choice for modern cryptographic needs. Its design not only addresses the inherent weaknesses observed in CBC-MAC and CCM but also sets a new standard for efficiency, making it a compelling option for securing data in an increasingly interconnected and high-speed digital world.

Conclusion

In conclusion, this research paper introduces a significant enhancement to the GCM mode of authenticated encryption through the incorporation of an "offset" mechanism, aimed at augmenting privacy and bolstering resistance against differential cryptanalysis. The modified GCM mode retains its original advantages, such as high efficiency, simplicity, and the use of a single cryptographic key, while the introduction of unique offsets complicates the predictability that underpins differential cryptanalysis. This innovation ensures that the improved GCM stands as a formidable option for applications requiring authenticated encryption, especially in scenarios where high-speed, parallelizable cryptographic operations are paramount.

The detailed analysis and discussions presented in the paper highlight the practicality of the offset-enhanced GCM in contemporary cryptographic applications. By maintaining the mode's original features and adding minimal overhead, the paper convincingly argues for the enhanced mode's suitability in securing high-speed networks and systems against sophisticated cryptanalytic attacks, without compromising on efficiency or security.

Moreover, the paper's exploration into the operational framework, including the meticulous integration and computation of offsets in both encryption and decryption processes, underscores the thoughtful approach taken to improve GCM. The security proofs and theoretical discussions further solidify the enhanced GCM's stance as a robust, secure, and efficient mode of operation that can significantly contribute to the field of cryptography.

Future research could potentially explore the practical implications of this enhancement in real-world applications, examining its performance and security in diverse scenarios. Additionally, the adaptability of the offset mechanism in other cryptographic modes and its potential to enhance the security of existing protocols could offer exciting avenues for further exploration. Overall, this paper not only contributes to the cryptographic community by presenting a more secure and efficient version of GCM but also sets the stage for future advancements in the field of authenticated encryption.

References

1. Rogaway P. Evaluation of some blockcipher modes of operation. Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan. 2011.
2. Bellare M, Rogaway P, Wagner D. A conventional authenticated-encryption mode. Manuscript. 2003.
3. Švenda P. Basic comparison of Modes for Authenticated-Encryption (IAPM, XCBC, OCB, CCM, EAX, CWC, GCM, PCFB, CS).
4. Jutla CS. Parallelizable encryption mode with almost free message integrity. Contribution to NIST. 2000.
5. Rogaway PM. Bellare, and J. Black, OCB: A block-cipher mode of operation for efficient authenticated encryption. ACM Transactions on Information and System Security (TISSEC). 2003; 6(3): 365-403.
6. Krovetz T, Rogaway P. The OCB authenticated-encryption algorithm. 2014.
7. Rogaway P. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. in International Conference on the Theory and Application of Cryptology and Information Security. 2004. Springer.
8. Krovetz T, Rogaway P. The software performance of authenticated-encryption modes. in International Workshop on Fast Software Encryption. 2011. Springer.
9. Black J, Rogaway P. A block-cipher mode of operation for parallelizable message authentication. In International Conference on the Theory and Applications of Cryptographic Techniques. 2002. Springer.
10. Iwata T. New blockcipher modes of operation with beyond the birthday bound security. In International Workshop on Fast Software Encryption. 2006. Springer.
11. Dworkin M. Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. 2004. National Institute of Standards and Technology.
12. Iwata T, Kurosawa K. Omac: One-key cbc mac. In International Workshop on Fast Software Encryption. 2003. Springer.
13. Wegman MN, Carter JL. New hash functions and their use in authentication and set equality. Journal of computer and system sciences. 1981; 22(3): 265-279.
14. Kohno T, Viega J, Whiting D. The CWC authenticated encryption (associated data) mode. ePrint Archives. 2003.
15. Rogaway P. Authenticated-encryption with associated-data. in Proceedings of the 9th ACM conference on Computer and communications security. 2002. ACM.
16. Szalachowski P, Ksiezopolski B, Kotulski Z. CMAC, CCM and GCM/GMAC: Advanced modes of operation of symmetric block ciphers in wireless sensor networks. Information Processing Letters. 2010; 110(7): 247-251.
17. Housley R. Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS). 2007.
18. Hiller J. Improving functionality, efficiency, and trustworthiness of secure communication on an internet diversified by mobile devices and the internet of things. 2023, Dissertation, RWTH Aachen University, 2022.

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079

19. McGrew D, Viega J. The Galois/counter mode of operation (GCM). Submission to NIST Modes of Operation Process. 2004; 20.

20. Miao X. Bit-Sliced Implementation of SM4 and New Performance Records. 2023.

21. Lipmaa H, Rogaway P, Wagner D. CTR-mode encryption. In First NIST Workshop on Modes of Operation. 2000. Citeseer.

22. McGrew DA. Counter mode security: Analysis and recommendations. Cisco Systems. 2002; 2(4).

23. Gueron S, Jha A, Nandi M. COMET: COunter Mode Encryption with authentication Tag. 2019.

24. Saarinen MJO. Cycling attacks on GCM, GHASH and other polynomial MACs and hashes. In International Workshop on Fast Software Encryption. 2012. Springer.

25. Lipmaa H, Wagner D, Rogaway P. Comments to NIST concerning AES modes of operation: CTR-mode encryption. 2000.

26. Gligor VD, Donescu P. Fast encryption and authentication: XCBC encryption and XECB authentication modes. in International Workshop on Fast Software Encryption. 2001. Springer.

27. Jutla CS. Encryption modes with almost free message integrity. In International Conference on the Theory and Applications of Cryptographic Techniques. 2001. Springer.

28. Benvenuto CJ. Galois field in cryptography. University of Washington. 2012.

29. Aljohani M. Performance Analysis of Cryptographic Pseudorandom Number Generators. IEEE Access. 2019; 7: 39794-39805.

30. Bellare M. A concrete security treatment of symmetric encryption. In Proceedings 38th Annual Symposium on Foundations of Computer Science. 1997. IEEE.

31. Bellare M, Kilian J, Rogaway P. The security of the cipher block chaining message authentication code. Journal of Computer and System Sciences. 2000; 61(3): 362-399.

32. Goldreich O, Goldwasser S, Micali S. How to construct random functions. Journal of the ACM (JACM). 1986; 33(4): 792-807.

Citation: Nawaz MF, Nawaz Y (2024) Redefining GCM's resistance to cryptanalysis with offset mechanisms. Trends Comput Sci Inf Technol 9(1): 042-051.
DOI: https://dx.doi.org/10.17352/tcsit.000079