

Received February 10, 2020, accepted February 24, 2020, date of publication March 3, 2020, date of current version March 13, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2977953

# Patterned Cipher Block for Low-Latency Secure Communication

SEOUNGHWAN OH, SEONGJOON PARK, AND HWANGNAM KIM 

Electrical Engineering Department, Korea University, Seoul 02841, South Korea

Corresponding author: Hwangnam Kim (hkim@korea.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIT) under Grant 2020RIA2C1012389, and in part by the Human Resources Program in Energy Technology of the Korea Institute of Energy Technology Evaluation and Planning (KETEP) Granted Financial Resource from the Ministry of Trade, Industry and Energy, South Korea, under Grant 20174030201820.


**ABSTRACT** Despite the increasing importance of network security, increasing Internet of Things (IoT) uptake and traffic tends to apply tighter resource constraints for cryptography. To cope with the constraints, security systems must choose between time cost and security. Cyber-attack model evolution and quantum computing technologies have severely limited current cryptography uptake and imposed too much overhead to operate effectively on lightweight communication environments. Therefore, we propose a new operation mode using multiple symmetric key ciphers alternately in a regularized order. The proposed design exploits lightweight cryptography methods, reducing encryption/decryption overhead compared to a single heavy cryptography approach, as well as avoiding exhaustive key extraction attack. Since sequences can change both time cost and security performance widely, the design can be applied to various situations, from the delay-constrained communications to highly secure networks. Our cryptography design incorporates patterned cipher block (PCB) operation, an integrity verification technique to identify if a ciphertext has been forged, handshaking protocol exchanging pattern information and a key using two-round communication, and pattern optimization to maximize the cryptographic performance. We confirmed the proposed operation mode numerically, and verified the outcomes experimentally, confirming that the proposed scheme outperformed current best practice cryptography.

**INDEX TERMS** Cryptography, symmetric key cipher, operation mode, confidentiality, integrity, authentication, protocol.

## I. INTRODUCTION

Cryptography is of particular significance when sender and receiver want to exchange important information without exposing it to a third party. Security and time costs have become critical considerations recently, and cryptography has evolved to balance them across various network environments. However, current cipher algorithms often suffer from excessive delay and system performance impacts because sensor, actuator, and vehicular devices tend to have limited resources and low computation power, making them difficult to incorporate into network environments.

Several lightweight solutions have been previously proposed to address the growing constraints, including the simple exclusive OR (XOR) operation [29], packet header encryption [21], and specific region encryption [1].

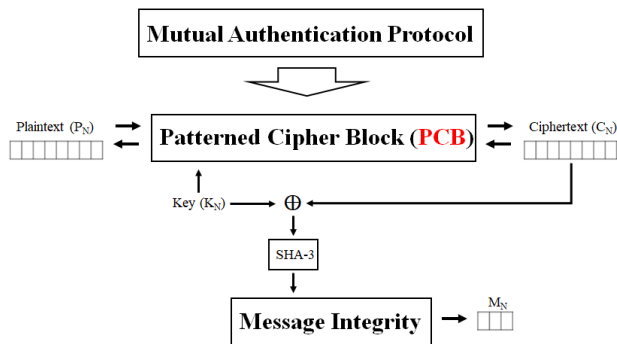
The associate editor coordinating the review of this manuscript and approving it for publication was Luis Javier Garcia Villalba .

However, an attacker can forge or replace ciphertext under stream ciphers, because message integrity is not verified, and error propagation often occurs when synchronization fails. On the other hand, the original data in the packet could be exposed and vulnerable to packet sniffing attacks if only the packet header is encrypted [13]. Although encrypting a specific region with a symmetric key cipher is more secure than stream ciphers, region detection incurs high overheads unless the specific region is fixed by pixel and used only when data importance can be differentiated. Although some lightweight solutions can reduce encryption/decryption delay, they can also reduce security, hence narrowing the potential application range.

Several groups, including Academy Research Institute Agency (ARIA) [18], SEED [20], and Advanced Encryption Standard (AES) [7] have warned that current symmetric key ciphers will no longer be safe when quantum computers are commercialized [4]. According to the National Institute of

**TABLE 1.** Notations for the proposed patterned cipher block (PCB) protocol.

Notation	Type	Purpose
PK, SK	Elliptic Curve Digital Signature Algorithm (ECDSA)	Encryption / Decryption
r	Random value	Mutual authentication through challenge-response
H	SHA-3	$SHA(HAMC_{SK}(r \oplus G^{x \text{ or } y}))$ generation for mutual authentication
M	SHA-3	$SHA(K_i \oplus C_i)$ generation for message integrity (MAC)
x or y	Random value	K(pre-master secret key) generation using Diffie-Hellman key exchange
G	Prime value	
N	Modulo value	
P	Pattern sequence	
PCB	AES-128 / 192 / 256	Operation mode for using symmetric key cipher

**FIGURE 1.** The proposed cryptography scheme.

Standards and Technology (NIST), internal algorithms for most symmetric key ciphers, except for AES, should be modified to have larger key length in Table 1 of [2]. AES also recommends using larger key length, e.g. 256-bit. However, network environments that require real time communication often cannot use larger key sizes due to encryption/decryption delay [31]. Current solutions suffer from increasingly tighter limitations due to time cost savings and security enforcement requirements. Attack models are constantly evolving and network traffic requirements are increasing. Tradeoff between performance and security hinders universal availability of conventional cryptography.

There is no better solution to resolve this network security deadlock. Therefore, we propose a cryptography scheme that can provide high level security while utilizing current symmetric key ciphers and hence reducing time delay. Fig. 1 shows the proposed cryptography design incorporates the patterned cipher block (PCB) that uses multiple alternate symmetric key ciphers per block following a particular sequence or pattern. In PCB operation, sender and receiver should have the same pattern and perform encryption/decryption using the same algorithm for each block. Thus, even if the attacker obtained a set of ciphertexts, the shared key is extremely hard to extract since the attacker does not know which set of blocks are encrypted with the same key. This will improve security and reduce time by enabling relatively fast encryption/decryption methods. The message integrity verification module allows recipients to verify whether message contents may have been improperly changed. Handshaking protocol exchanges pattern and key information with mutual authentication. Current symmetric key ciphers, such as quantum

resistive cryptography (QRC) or post-quantum cryptography (PQC), can be used with PCB containing patterned sequences and message authentication codes (MACs).

This paper incorporates the following contributions.

- A novel PCB mode utilizing alternating multiple symmetric key ciphers in a regularized order sequence. We also verified that PCB can defend existing attack models, minimizing cryptographic encryption/decryption time, and reduce error propagation and loss.
- A message integrity system that verifies message integrity in PCB mode. Message integrity system security is based on the hash function, which can be performed safely and quickly with integrity verification.
- Mutual authentication incorporated into the cryptography design, and proved that the protocol can exchange key and pattern information with two-round communication while simultaneously performing mutual authentication securely through a challenge-response system.
- Modeled PCB mode performance and designed the algorithm to derive the optimal cryptic index ratios. We also proposed a rate adaptation algorithm to flexibly change encryption/decryption delay by regenerating the pattern ratio.

The proposed operation mode can employ any symmetric key cipher, and hence could be applied to any case that exchanges a pre-master key for communication, such as Wireless Fidelity (Wi-Fi) or Transport Layer Security (TLS) systems. Lightweight symmetric key ciphers also reduce encryption/decryption, and the PCB disperses key usage, making it difficult for attackers to retrieve the key. Therefore, the proposed cryptography offers a promising solution for resource constrained delay-aware network security, such as sensor or mobile networks.

The remainder of this paper is organized as follows. Section II introduces current operating modes and Section III discusses related security issues. Section IV details the proposed cryptography design, and we model PCB and address the optimization and rate adaptation algorithms in Section V. Section VI evaluates the proposed cryptography experimentally, and Section VIII summarizes and concludes the paper.

## II. RELATED WORK

Cryptography is divided into symmetric key ciphers or asymmetric key ciphers. Symmetric key ciphers are generally based on structural complexity and perform

encryption/decryption using 128, 192, or 256 bit symmetric key lengths. However, they have the disadvantage that the symmetric key must be securely shared. In contrast, asymmetric key ciphers take advantage of the mathematical difficult discrete logarithm problem and uses has two keys: a public key for encrypting and a private key for decryption. However, asymmetric key ciphers are generally 2048 bit, hence encryption/decryption time is longer than for symmetric key ciphers. Thus, symmetric key ciphers are preferred to reduce network delay.

Since message sizes vary, they are usually divided into fixed-size blocks, as required by the symmetric key cipher. Various operating modes have been devised to accommodate various message sizes using symmetric key ciphers, including electronic codebook (ECB) [15], cipher block chaining (CBC) [26], propagating cipher block chaining (PCBC) [24], cipher feedback (CFB) [27], output feedback (OFB) [9], counter (CTR) [22], and Galois counter mode (GCM) [23].

Since ECB performs encryption/decryption in independent structures, there is no error propagation, and it is faster than other block based operation modes because it does not require padding, using the ciphertext stealing technique instead. However, ECB has a fatal disadvantage that an attacker can re-use ciphertext deterministically, because it does not provide message integrity authentication [17].

Unfortunately, CBC and PCBC require an initial vector (IV) and their structures allow previous ciphertexts to affect the next ciphertext generation in a chain format. Therefore, if a ciphertext bit error occurs during transmission, the error propagates and the original content cannot be obtained. They also require padding, which takes more time to encrypt and decrypt than ECB.

In contrast to ECB, CBC, and PCBC, which encrypt plaintext in block units, CFB, OFB, and CTR perform XOR operation with plaintext per bit using an IV, which is also often used to generate the key stream. Therefore, if the same IV is used, the generated key stream value also becomes the same, and an attacker can easily obtain the plaintext through XOR from the ciphertext. Even if the keystream value is unknown, plaintext can be predicted from the ciphertext through periodicity [5].

The GCM mode adds the Galois HASH (GHASH) authentication component to existing CTR for message integrity authentication, and hence is widely used for network security protocols, e.g. IPsec [19], SSH [16], and TLS [14]. GCM encryption encrypts plaintext using counter mode (CTR), generating an initial counter block by encrypting the synchronized IV with a secret key. Authenticated blocks generated from ciphertexts are multiplied with the initial counter block (GHASH), providing authenticated encryption with associated data (AEAD) functionality from the authenticated blocks. Hence, GCM is the only operation mode (SP 800-38D) that NIST has standardized [10]. However, it requires pre-computation and considerable additional information to use GCM for communication, which can cause several problems [11]. For example, error propagation can

occur if the IV does not match the bit synchronization, it is difficult to perform GCM real time in an IoT device environment because parallel operations are not possible, and it needs pre-computation for encryption [30].

The proposed cryptography design combines advantages across the operating modes. PCB adopts independent structures but uses multiple symmetric key ciphers. Periodically alternating key usage can provide high security, as well as prevent error propagation. Since it uses different keys to generate different MAC values each time for the same plaintext, PCB is verifiable whereas stream ciphers and ECB are not. Thus, the proposed secure protocol includes PCB operation mode, mutual authentication, and message integrity authentication for subsequent sections.

### III. PROBLEM FORMULATION & STATEMENT

Several issues must be addressed to send and receive messages securely in real time over a network. The sender and receiver must be mutually authenticated to prevent disguised attackers, the receiver must verify message integrity on the received ciphertext to determine it was sent by the expected sender, and we should also consider packet loss depending on the network environment. Since encryption/decryption overheads and security requirements may vary depending on the generated pattern, mathematical models for each pattern must be derived and optimized. The following list summarizes the challenges to overcome in designing the operation mode.

- **Message Confidentiality** Message confidentiality means that the information content not disclosed to an unauthorized third party. Thus, sender and receiver must transmit an encrypted message, i.e., ciphertext, using the cryptography to ensure a third party cannot derive the original plaintext from the ciphertext. Although the ciphertext is exposed to the third party, the third party does not know the key, hence the ciphertext is secure, according to *Kerckhoff's* principle [25]. Therefore, ciphertext security depends on what cryptography the application uses. Section IV-B discusses attack models and shows how to assess PCB message confidentiality in detail.
- **Message Integrity** Message integrity prevents an attacker from tampering by allowing the receiver to verify if the message content has been improperly changed during transmission. Although cryptography used by the sender and receiver may remain secure, message authentication technique is required to identify any cases when an attacker maliciously forges or modifies ciphertext. Message integrity security depends on the particular system and algorithm employed. Section IV-C discusses the PCB approach in detail.
- **Mutual Authentication** Mutual authentication identifies the other party's identity, preventing a disguised attacker. This is commonly achieved through challenge-response [8], established such that only the person who owns the unique key can perform it. The particular challenge-response system employed depends on the

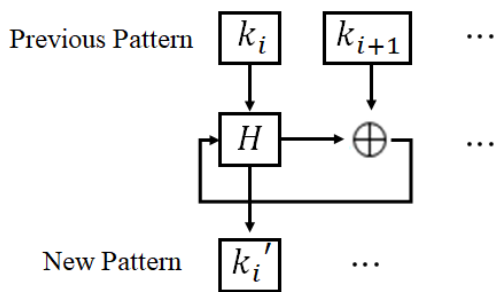


FIGURE 2. Proposed pattern generation structure.

cryptography security and protocol. Section IV-D discusses the proposed PCB mutual authentication system in detail.

- Pattern Optimization** PCB uses various symmetric key ciphers, with varying encryption/decryption overhead. Our cryptography design enhances security using formerly breakable symmetric key ciphers to the quantum-resistant level. We found that no current cryptography could resolve time cost and security issues simultaneously, due to structural the limitations. Following the analysis discussed in Section V-A, Section V-B proposes a genetic search algorithm to find the optimal cryptic index ratios.

IV. CRYPTOGRAPHY OVERVIEW

We propose an overall cryptography providing performance and high-level security simultaneously through patterned operation mode using different key lengths in an independent structure, message integrity to prevent attacker message forgery or replacement, mutual authentication protocol, and pattern optimization depending on the network communication environment.

A. PCB OPERATION MODE

The proposed PCB operation mode combines advantages from current modes, i.e., ECB [9] and GCM [23]. In PCB mode, sender and receiver have the same information or pattern, i.e., a specific sequence of integers that map to symmetric key ciphers. We define the *cryptic index*, a series of integers referring to specific symmetric key ciphers. If  $K_i$  in the previous pattern is the symmetric key to encrypt the  $i$ -th plaintext, then the  $i$ -th new symmetric key  $K'_i$  for the next pattern is generated following the structure shown in Fig. 2 before the previous pattern period ends. Symmetric key information for the pattern is cyclically generated through hash ( $H$ ) and XOR operation (Fig. 2). If the static session key is used for a long time, sender and receiver may be at risk from attackers. However, periodically reproducing the session key following the pattern generation structure in Fig. 2 ensures the key is safe to use for a long time. PCB mode performs encryption/decryption for each message block using the corresponding symmetric key cipher indicated by the cryptic index, following the pattern order. If there are  $n$  available symmetric key ciphers (and their pre-master keys), then pattern length should be larger than  $n$  so that some

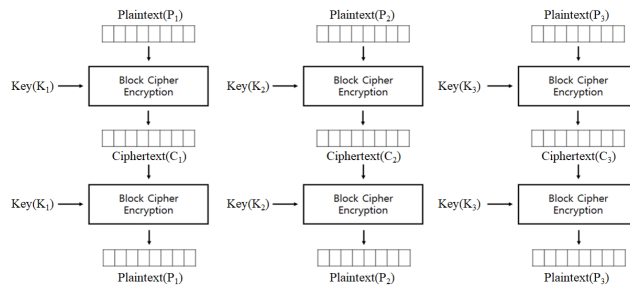


FIGURE 3. Proposed patterned cipher block (PCB) structure.

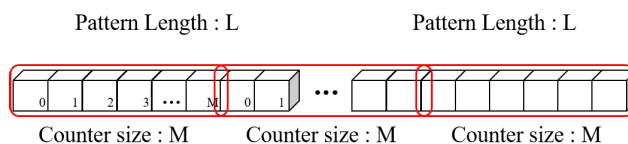


FIGURE 4. Counter using the proposed patterned cipher block (PCB).

or all are used in PCB mode. Fig. 3 shows the proposed PCB operation mode. The cipher keys in Fig. 3 are generated from Fig. 2. Since the plaintext was independently encrypted using different key lengths generated from Fig. 2, the plaintext and ciphertext maintain a one-to-one relationship. A secure symmetric key cipher is established at the session connection and plaintexts are encrypted according to the key length type. PCB is an independent structure with a pattern format that enables parallel encryption/decryption. Thus, PCB does not require an additional padding operation by using ciphertext stealing [6]. Therefore, even if bit error or loss occurs for a ciphertext in the middle of transmission, PCB can transmit in real time without error propagation.

Fig. 3 shows the proposed PCB has a pattern format that can use different key lengths, hence it is statistically unlikely that it would not generate the same ciphertext for the same plaintext, which makes it impossible for an attacker to analyze the pattern for block reuse or existing decryption. Network transmission control (TCP) or user datagram (UDP) protocol characteristics should be considered to apply PCB to the actual network environment. In contrast to TCP, packet loss may occur frequently when UDP is used in the network layer. Therefore, error propagation may occur or a proper ciphertext could be unable to obtain if block (e.g. GCM) or stream ciphers are used. In contrast, since PCB knows the pattern length,  $L$ , it can provide a counter size,  $M$ , for the ciphertext blocks to detect packet loss and skip it, as shown in Fig. 4. Therefore, only the lost block is retransmitted or ignored.

Three symmetric key cipher lengths are commonly used: 128, 196, and 256 bit. PCB mode can assign different cryptic indices for the same cipher while differing key lengths, which can be easily generated in several ways as follows.

- Use a single hash function with a single password (PW) as input, and then cut the resulting value to fit the key length. Using the hash function ensures that even if the attacker finds one key, remaining keys are unknown due to preimage resistance, a hash function property.



- 2) Use three different hash functions with a single PW as input, and then cut the resulting values to fit the key length. All three hash functions provide suitable codes, but having three values available can help avoid collisions.
- 3) (most secure) as for method 2, but generate three different PWs.

The proposed cryptography has the advantage of providing not only high-level security and real-time performance, but also compatibility without additional cost, which makes it easier to apply to other symmetric key ciphers (e.g. Data Encryption Standard (DES), TripleDES, Advanced Encryption Standard (AES), Academy Research Institute Agency (ARIA), SEED). For example, applying PCB to DES, which is not currently used due to vulnerability, provides the following benefits. PCB can slice the variable text data to 64bits, as required by DES. Then the DES data blocks are encrypted by the patterned keys. This quickly encrypts the plaintexts or ciphertexts due to the independent structure, achieving high-level security through the patterned keys. Thus, PCB can slice variable text data to the required data size required for other symmetric key ciphers without additional padding operation due to an independent structure. Furthermore, PCB provides the message integrity authentication.

**B. ATTACK MODEL AND SECURITY PROOF**

We segregated attack models into ciphertext only (COA), known plaintext (KPA), chosen plaintext (CPA), and chosen ciphertext (CCA) attacks, depending on how the sender and receiver exchange messages [12]. Since ECB mode generates the same ciphertext for the same plaintext, an attacker can find repeated ciphertext for a particular plaintext even if only the ciphertext is obtained. Thus, ECB mode is vulnerable to CPA. In contrast, PCB uses independent structures to apply patterned formats, hence CPA is impossible, which can be proven using the LR Encryption Oracle. Let the adversary be  $A^{E_k(LR(\dots,b))}$ , where the plaintext is  $M_i$ , and ciphertext  $C[i]$ .  $C[1]$  ( $=E_k(0^n)||E_k(0^n)$ ) is generated by encrypting  $M_1$  ( $=0^n||0^n$ ), and  $C[2]$  ( $=E_{k'}(0^n)||E_{k'}(1^n)$ ) is generated by encrypting  $M_2$  ( $=0^n||1^n$ ). Thus,  $C[1]$  and  $C[2]$  in the left word are indistinguishable, and hence  $E_k(0^n) \neq E_{k'}(0^n)$ . Thus, IND-CPA attacks are impossible since  $Adv_{SE}^{IND-CPA}(A)$  is just 0. Therefore, Algorithm 1 ensures that PCB can defend against IND-CPA attacks.

**C. MESSAGE INTEGRITY AUTHENTICATION**

No matter how secure a ciphertext encrypted using PCB operation mode, if message integrity is not verified, the receiver cannot recognize if an attacker deleted or replace part of the ciphertext during communication. Figure 5 shows the proposed message integrity system to address ciphertext stealing. The  $(N - 1)$ -th plaintext in plaintext length  $N$ ,  $P_{N-1}$ ,  $E_{K_{N-1}}(P_{N-1})$  is generated by encrypting  $P_{N-1}$ . Then,  $C_{N-1}$  ( $=E_{K_{N-1}}(P_N||Tail)$ ) is generated by encrypting  $(P_N||Tail)$ . Final output value  $M_{N-1}$  ( $=SHA(K_{N-1} \oplus C_{N-1})$ ) is generated by XOR operation on the ciphertext  $C_{N-1}$  with  $K_{N-1}$ .

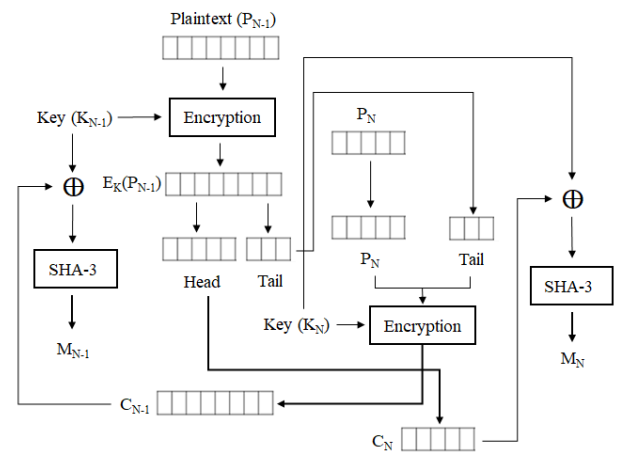
**Algorithm 1** IND-CPA Attack Model

**Input:**  $M_1 \leftarrow 0^n||0^n, M_2 \leftarrow 0^n||1^n$ ; (Plaintext)  
**Output:**  $C[1]C[2] \leftarrow E_k(LR(M_1, M_2, b))$ ; (Ciphertext)

- 1: Let Adversary be  $A^{E_k(LR(\dots,b))}$
- 2: **if**  $C[1] = C[2]$  **then**
- 3:     return 1
- 4: **else**
- 5:     return 0
- 6: **end if**
- 7:

$$E_k(LR(M_l, M_r, b)) = \begin{cases} E_k(M_l) & \text{if } b = 1 \\ E_k(M_r) & \text{if } b = 0 \end{cases}$$

- 8:  $C[i]$  denotes the  $i$ -th block of a string  $C$
- 9: In the left word,  $C[1]C[2] = E_k(0^n)||E_{k'}(0^n)$
- 10: As a result,  $E_k(0^n) \neq E_{k'}(0^n)$
- 11: In the right word,  $C[1]C[2] = E_k(0^n)||E_{k'}(1^n)$
- 12: As a result,  $E_k(0^n) \neq E_{k'}(1^n)$
- 13:  $Adv_{SE}^{IND-CPA}(A) = 0 - 0 = 0$
- 14: Because of different key



**FIGURE 5.** Message Integrity Authentication Structure.

The  $N$ -th operation is performed in the same manner as described above. For example, the  $N$ -th ciphertext,  $C_N$ , is generated by encrypting the  $N$ -th plaintext without padding, and the  $N$ -th MAC value,  $M_N$  ( $=SHA(K_N \oplus C_N)$ ), is generated by XOR operation on the ciphertext  $C_N$  with  $K_N$ . Thus,  $M_N$  ( $=SHA(K_N \oplus C_N)$ ) and  $C_N$  are generated independently. Thus, PCB allows secure and fast communication since it allows encryption/decryption and integrity information verification to be parallel operations. Therefore, the sender sends ciphertexts  $C_i$  and message authentication codes (MAC)  $M_i$  ( $=SHA(K_i \oplus C_i)$ ) to the receiver to prevent manipulation by an attacker.

Integrity information verification checks the ciphertext using the patterned keys, hence even if an attacker successfully forged or exchanged ciphertext in transit, the receiver can verify the ciphertext was sent from the expected sender using  $M_i$ .

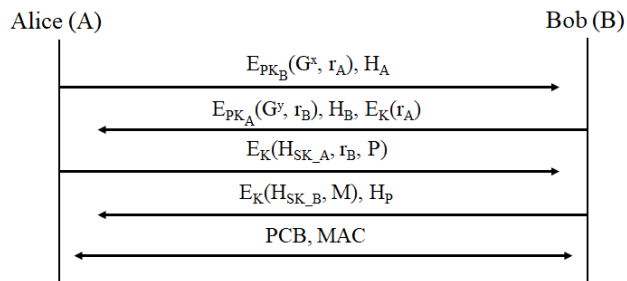


FIGURE 6. Proposed mutual authentication protocol.

- 1) The first preimage resistance defines that, for a given hash value, it is computationally difficult to find the input value that produces the hash value, i.e., given  $h$ , it is infeasible to find  $x$  such that  $H(x) = h$ .
- 2) The second preimage resistance defines that it is computationally infeasible to find an input value different from the current input value without changing the hash value for the current input value, i.e., given  $x$ , it is infeasible to find  $y$  such that  $H(y) = H(x)$ .
- 3) The third collision resistance defines that finding two input values that produce the same hash value is computationally difficult, i.e., it is infeasible to find any  $(x, y)$  such that  $H(y) = H(x)$ .

Therefore, cryptography hash is a one-way function that cannot reproduce an original text from the hash value. We enhanced message integrity security by using SHA-3, e.g. SHA-384, with hash based message authentication (HMAC), which has been proven secure, considering SHA-2 or SHA-256 security.

**D. PROTOCOL**

For the sender and receiver to encrypt sensitive data information and securely transmit the pattern information, it is necessary to securely establish a session through mutual authentication. Fig. 6 shows the proposed mutual authentication protocol, which that satisfies secure mutual authentication by performing a challenge-response. Thus, the proposed protocol can prevent an attacker from disguising themselves as a sender.

Suppose that before establishing a session to communicate securely using the PCB, Alice and Bob try to form a secure channel with some crypto elements corresponding to TABLE. 1.

- Alice encrypts  $G^x$  using Bob’s public key ( $PK_B$ ) to create a pre-master secret key and sends  $r_A$  to confirm that Bob is the correct receiver from Alice’s perspective.
- Alice sends  $H_A$  to Bob.
- Bob uses  $H_A (=SHA(HAMC_{SK_A}(r_A \oplus G^x)))$  to simultaneously verify Alice’s signature and message integrity. The signature is a means of proof that Alice is the person Bob wants to communicate with.
- Bob decrypts the ciphertext received from Alice with his secret key ( $SK_B$ ) to obtain  $G^x$  and  $r_A$ , and generates a pre-master secret ( $K$ ) from  $G^{xy} \pmod N$ .

- Bob encrypts  $G^y$  and  $r_B$  with Alice’s ( $PK_A$ ), encrypts  $r_A$  received from Alice with  $K$ , and sends these to Alice with  $H_B (=SHA(HAMC_{SK_B}(r_B \oplus G^y)))$ .
- Alice receives  $r_A$  (challenge) from the receiver, and acknowledges that the receiver is Bob (response). She then decrypts the ciphertext received from Bob with her  $SK_A$  and obtains  $G^y$  and  $r_B$ .
- Alice generates  $K$  by  $G^{xy} \pmod N$ . This allows Alice and Bob to share  $K$  securely using the Diffie-Hellman key exchange technique.
- Alice decrypts ciphertext  $E_K(r_A)$  received from Bob with the generated  $K$  to confirm that  $r_A$  is the  $r$  sent by her. She then encrypts  $H_{SK_A} (=HAMC_{SK_A}(r_A \oplus G^x)), r_B$ , and  $P$  with the  $K$ .
- Bob decrypts the ciphertext received from Alice to verify  $r_B$ . If  $r_B$  is the  $r$  sent by him, the first challenge-response is completed safely.
- Bob compares  $H_A$  received previously from Alice with  $H'_A$  obtained using Alice’s  $H_{SK_A}$  with SHA-3. Then Bob sends Alice the ciphertext  $H_{SK_B} (=HAMC_{SK_B}(r_B \oplus G^y))$ , general messages  $M, K$ , and  $H_P (=SHA(HAMC_K(P)))$ , indicating that Bob has received  $P$  correctly.
- From the ciphertext received from Bob, Alice uses Bob’s  $H_{SK_B}$  to get plaintext  $H'_B$  using SHA-3 hash is the same as  $H_B$  received previously from Bob. This safely completes the second challenge-response.

Mutual authentication is successful if both processes complete successfully. The proposed protocol is similar TLS Version 1.3 [28] in terms of overhead because it receives the pattern value ( $P$ ) from only three information transmissions. The protocol could also employ the digital signature algorithm (DSA), used in TLS for mutual authentication, but the two-step verification provides increased security without significant increased overhead. Thus, the proposed protocol allows Alice and Bob to not only complete mutual authentication, preventing an attacker from disguising themselves Alice or Bob, but also to share pre-master secret key and pattern information securely.

**V. PROPOSED PATTERNED CIPHER BLOCK ANALYSIS**

This section analyzes the PCB mode in terms of time cost and security. To verify PCB performance, we modelled expected encryption/decryption time cost and relative gain of the key attack cost. Consequently, we propose a parameter adjustment scheme to enhance encryption/decryption performance while increasing channel security. Section V-A discusses PCB dynamics, and Sections V-B and V-C discuss the proposed parameter tuning algorithm design.

**A. PERFORMANCE ANALYSIS**

Assume an  $L$  size PCB pattern, randomly generated from  $N$  cryptic indices. Since encryption/decryption time varies considerably for different methods, we determined expected PCB time cost as Eq. (1).

$$T = \sum_{i=0}^{N-1} r_i t_i \tag{1}$$

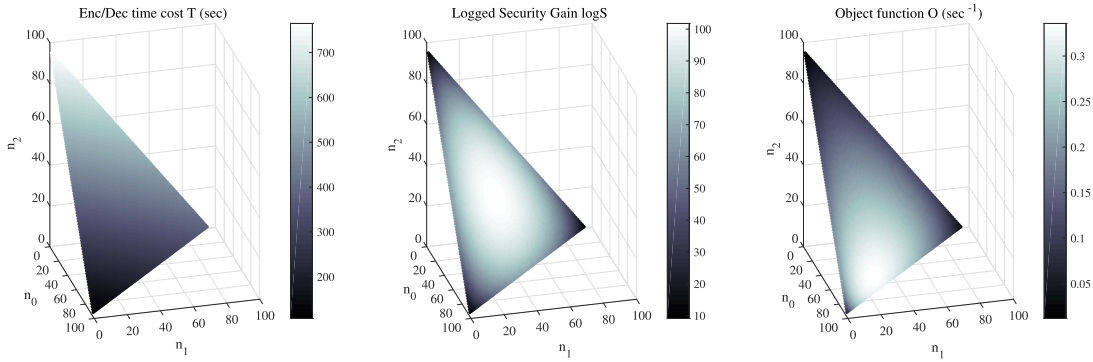


FIGURE 7. Analytic results for  $T$ ,  $\log S$ , and  $O$  when  $N = 3$ .

where  $r_i$  and  $t_i$  are the ratio and expected time cost for the  $i$ -th cryptic index.

Most recent studies based key attack methods on collections of cipher/plaintext ( $C/P$ ) pairs ( $E_k(P)/D_k(C)$ ). However, as discussed above, PCB is resilient to key attack, since sequential collection does not guarantee using the same key. Therefore, we propose a modified attack model, *Selective Key Attack*, where the attacker randomly selects a subset of the  $C/P$  pair collection and tries to extract one of the keys. If the key extraction process is unsuccessful, the attacker re-selects from the subset and tries again. The attacker does not know  $L$ , since the pattern information is mutually passed between the hosts during key exchange. Although the attacker may know the key usage periodicity, they first need to find  $L$ . Almost all key extraction techniques require a large number of  $C/P$  pairs ( $2^{39}$ ) [3], so the attacker must guess  $L$  and then try to find the  $N$  subsets along the guessed length. The expected number of trials to find the correct pattern is Eq. (2)

$$S = A \times \frac{L!}{\prod_{i=0}^{N-1} n_i!} \quad (2)$$

where  $A$  is the size of the search space of  $L$ , and  $n_i = \lfloor Lr_i \rfloor$  is the number of usages of the  $i$ -th cryptic index in a pattern. We use  $S$  for the *security gain*, since the expected key extraction time is calculated by the multiplication of  $S$  and the sum of the key extraction time of each cryptic index.

### B. PATTERN OPTIMIZATION

From Eq. (2),  $S$  is maximized when  $r_0 = r_1 = r_2 = \dots = r_{N-1}$ . However, if some encryption/decryption time costs ( $t_i$ ) are particularly high, maximizing  $S$  could redundantly reduce expected PCB time. Considering time cost and security from Eqs. (1) and (2), respectively, we define the object function,  $O$ , to optimize  $r_i$  as Eq. (3)

$$O(R) = \frac{\ln S}{T} \quad (3)$$

where  $R = \{r_0, r_1, r_2, \dots, r_{N-1}\}$ . The change of  $S$  with respect to  $R$  is significantly dominant compared with the change of  $T$ , due to the exponential. Therefore, to fairly consider both metrics, we take the logarithm of  $S$ . Thus, the optimal ratio along the cryptic indices,  $R^*$ , is Eq. (4).

$$R^* = \arg \max_R O(R) \quad (4)$$

where

$$\sum_{i=0}^{N-1} r_i = 1. \quad (5)$$

---

### Algorithm 2 Pattern Optimization

---

**Input:** Pattern length  $L$

**Output:** A set of the number of key usages  $M$

- 1: Collect  $t_i$  where  $0 \leq t < N$
  - 2: Create  $M_0$
  - 3: Set all  $n_i$  in  $M_0$  to  $\lfloor L/N \rfloor$
  - 4: **if**  $\sum n_i < L$  **then**
  - 5:    $n_0 \leftarrow L - \sum_{i=1}^{N-1} n_i$
  - 6: **end if**
  - 7: Set  $M_0$  to a candidate solution
  - 8: **while** True **do**
  - 9:   Populate  ${}_N P_2$  solutions from candidates (see Fig. 8)
  - 10:   Evaluate solutions including candidates using (4)
  - 11:   Select two solutions  $M'$  and  $M''$  that have the largest  $O$
  - 12:   **if**  $M'$  is one of the candidates **then**
  - 13:     **return**  $M'$
  - 14:   **end if**
  - 15:   Terminate the solutions except  $M'$  and  $M''$
  - 16:   Set  $M'$  and  $M''$  to the candidate solutions
  - 17: **end while**
- 

However, Eq. (4) is challenging to solve mathematically due to its complexity. Therefore, we propose a simplified genetic algorithm (Algorithm 2) to solve the optimization exploiting the discrete property and planarity of the constraint in Eq. (5). Although  $r_i$  is floating point,  $n_i$  should be integer. The approach is genetic search for a solution that contains  $n_i$  where  $0 \leq i < N$ , along a discrete search space with the Eq. (6)

$$\sum_{i=0}^{N-1} n_i = L. \quad (6)$$

This is reasonable, since  $L$  is determined before generating the pattern. Fig. 7 shows analytic results for  $T$ ,  $\log S$ , and  $O$  calculated from Eqs. (1), (2), and (3) when  $N = 3$ , and

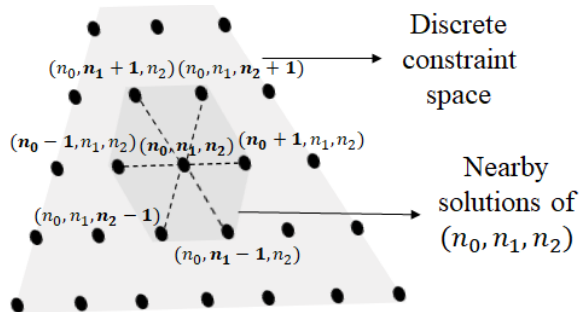


FIGURE 8. Populating candidate solutions when  $N = 3$ .

$L = 97$ .  $S$  and  $O$  have only one extreme point on the constraint plane. Therefore, any point on the plane larger than any nearby values is the maximum value. Algorithm 2 first adds a center point of the constraint space as a candidate solution, and then populates new solutions from existing candidate solutions in Fig. 8. The populated solutions are adjacent to the candidate solution, with  $\sqrt{2}$  distance on the search plane. The number of populated solutions  $=_N P_2$ , since there are 6 populated solutions when  $N = 3$  in Fig. 8. We then calculated Eq. (4) with maximum  $_N P_2 + 2$  solutions for each and remain 2 solutions. If the maximum solution is one of two solutions at the start of the loop, the algorithm breaks the loop and returns the maximum as the optimal solution.

C. RATE ADAPTATION

The proposed optimization algorithm provides the optimal ratio of cryptic indices where time cost and security were equally considered. However, from the hosts' viewpoint, encryption/decryption delay above a certain level, e.g. real-time system deadline, could not be tolerated in some applications. Following the analysis in Section V-A, we propose a PCB based rate adaptation (PRA) method that monitors data exchange overhead and dynamically updates the pattern to sustain the required network performance.

Eqs. (1) and (2) suggest that time cost decreases as  $n_0$  (the fastest cryptic index) increases, and security gain increases when all cryptic indices are equally distributed. Regenerating the pattern while differentiating  $n_i$  can change  $T$  and  $S$  to either reduce encryption/decryption overhead or enhance security. However, modifying  $R$  is not just a trade-off between  $T$  and  $S$ , since their maximized points are not bipolarized. For example, let  $R = \{0.4, 0.4, 0.2\}$ ,  $t_0 = 1$ ,  $t_1 = 2$ , and  $t_2 = 4$ . Then  $S$  and  $T$  both increase if  $R \rightarrow R' = \{0.3, 0.3, 0.4\}$ , whereas if  $R \rightarrow R'' = \{0.4, 0.3, 0.3\}$ ,  $S$  increases as for  $R'$ , but  $T$  increases less. Thus, the algorithm can change the pattern to satisfy both time cost and security requirements for communication.

To solve the constraint, we utilize the genetic approach from Section V-B. Algorithm 3 shows the Proposed PRA mechanism. First, we generate a pattern with optimized ratio from Algorithm 2. Before regenerating the pattern, the algorithm periodically sends a ping message with block size  $L$  and calculates the round trip time (RTT) for the packet. The algorithm defines RTT as elapsed time from encrypting the

Algorithm 3 Patterned Cipher Block (PCB) Based Rate Adaptation (PRA)

```

1:  $M \leftarrow R^*$  from Algorithm 2.
2: Get desired RTT  $T_{RTT}^*$  from the application
3: while True do
4:    $T_{RTT} \leftarrow \text{MeasureRTT}()$ 
5:   if  $T_{RTT} \simeq T_{RTT}^*$  then
6:     Regenerate the pattern without changing  $M$ 
7:     continue
8:   else if  $T_{RTT} < T_{RTT}^*$  then
9:     Populate  $_N P_2$  solutions from the  $M$ 
10:    Evaluate solutions excluding  $M$  using (4)
11:    Select a solution  $M'$  that has lower  $T$  and maximizes  $O$ 
12:     $M \leftarrow M'$ 
13:    Regenerate the pattern
14:   else if  $T_{RTT} > T_{RTT}^*$  then
15:     Populate  $_N P_2$  solutions from the  $M$ 
16:     Evaluate solutions excluding  $M$  using (4)
17:     Select a solution  $M'$  that has lower  $S$  and maximizes  $O$ 
18:      $M \leftarrow M'$ 
19:     Regenerate the pattern
20:   end if
21: end while
    
```

ping to decrypting the ping at the same host. Since RTT represents round trip time in the application layer considering the cryptography, RTT measurement offers a reasonable metric to keep the track of redundant encryption/decryption overhead. If measured RTT is larger than the required RTT, the algorithm chooses a ratio with lower  $T$  and largest  $O$  from the populated solutions. On the other hand, if the measured RTT is less than the required RTT, it chooses ratio with highest  $S$  and largest  $O$  from the populated solutions. If the measured RTT (approximately) equals the desired value, the algorithm does not differentiate the ratio and regenerates the pattern. Pattern re-assignment overhead is relatively low, since pattern-sharing completes after a single communication round and changing the pattern does not require computational resource, such as hashing or key generation, only notification of the sequence change. PRA is based on the unique availability of the proposed cryptography that computational performance or security can be dynamically adjusted by parameter modification. The algorithm could be potentially improved in several ways.

- **Network performance measurement.** Although RTT is an explicit solution to determine network status, this approach could limit the algorithm's utility. Other metrics, e.g. link state, could also be a reasonable solution for other layers.
- **Fast adaptation.** The proposed algorithm modifies pattern ratios in a stepwise manner. If the measured RTT is significantly different from the desired RTT, more significant changes could reduce the adaptation overhead.



However, radically modifying the ratio could cause fluctuating network performance. Thus, an environment-free robust solution needs to be thoroughly researched to accelerate rate adaptation.

We verified that assigning the cryptic index determines the time and security performance for the proposed PCB. From the derived performance model, we proposed optimization and rate adaptation algorithms that search for the best parameter for the given problem. To the best of our knowledge, the proposed optimization and rate adaptation algorithms are the first attempt to flexibly reform cryptography parameter to fulfil network requirements. The rate adaptation metric and fast adaptation method will be addressed in future study to reinforce the proposed cryptography design.

**VI. EVALUATION**

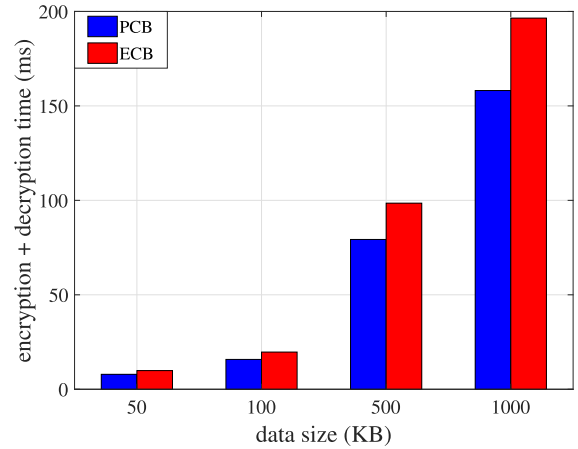
We have proved the proposed cryptography’s security (Section IV-B) and derived a mathematical expression for the enhancement (Section V-A). Therefore, this section focuses on encryption/decryption delay. To measure empirical time costs, we implemented ECB, and GCM in python. Also, we developed PCB in python in the same environment. The following experiments were performed using a PC with Intel(R) Core(TM) i5-6600 CPU @ 3.30GHz (quad core), 16GB RAM, and GeForce GTX 1060 3GB. Our implementation included the proposed PCB operation mode, handshaking protocol, and message integrity authentication system addressed in Sections IV-A, IV-D, and IV-C. We deployed the `pyaes` package to run the practical symmetric key cipher, using three cryptic indices (128, 192, and 256 bit key lengths) for the AES algorithm. We deployed two hosts connected at the same Wi-Fi Access Point (Wi-Fi AP), so the hosts send packets by two-hop communication.

**A. ENCRYPTION/DECRYPTION DELAY EVALUATION**

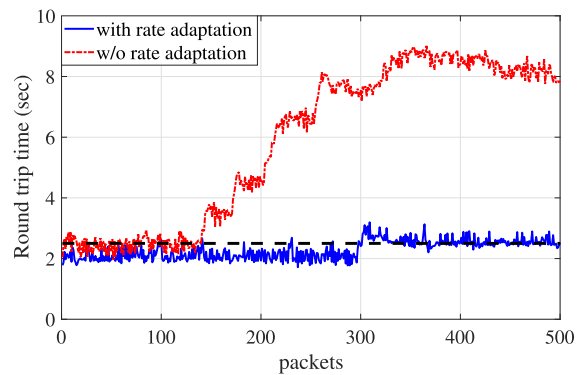
We implemented a simple video streaming application to measure practical encryption/decryption delay. After the application establishes the connection, one host sends an image data frame encrypted in PCB operation mode at 30 fps. The other host receives and decrypts the encrypted data and displays the image. The application measures encryption/decryption delay until video streaming ends. The pattern optimization proposed in Section V-B was employed for this evaluation, and we set  $L = 57$ . For comparison, we used ECB operation mode with 256 bit AES. Fig. 9 shows average encryption/decryption time for PCB and ECB for plaintext = 50, 100, 500, and 1000. PCB operation mode outperforms ECB, with approximately 15% decreased delay for all data sizes. Encryption/decryption time differences between PCB and ECB increased when data size increased. Therefore, PCB always provided improved network performance compared to the other operation modes, and this becomes more significant as network data rate increases.

**B. PRA EVALUATION**

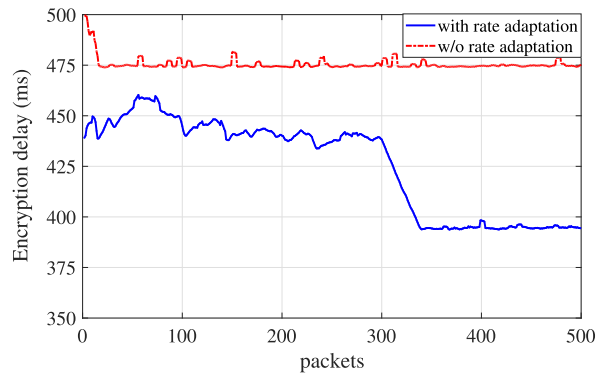
To verify the PRA designed in Section V-C, we augmented the algorithm for the cryptography implementation, setting data rate = 5Mbps in the TCP environment and RTT



**FIGURE 9.** Encryption/decryption delay for PCB and ECB with various data sizes.



**FIGURE 10.** Round trip time (RTT) of the cases using PRA and not using PRA.

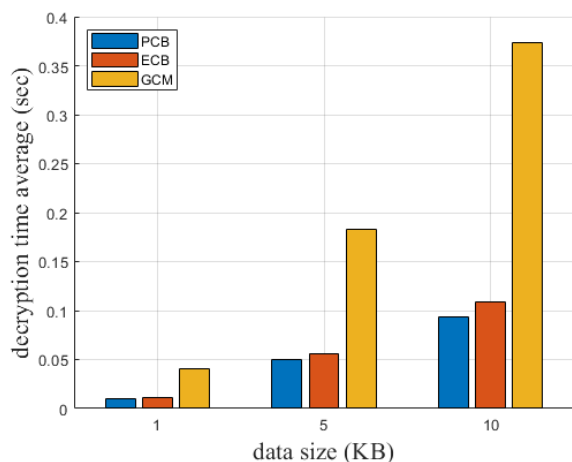


**FIGURE 11.** Encryption delay of the cases using PRA and not using PRA.

requirement = 0.5 seconds. Pattern regeneration and distribution was performed every second. Fig. 10 shows that RTT includes encryption/decryption delay, which is the same as the PRA metric. Fig. 10 and Fig. 11 show RTT and encryption/decryption delay for all packets with and without rate adaptation. The black line shows the required RTT. RTT with PRA approximately meets the RTT requirement, whereas RTT without PRA monotonically increases due to encryption/decryption delay. A remarkable RTT change near the 300th packet (Fig. 10) let PRA modify the pattern ratio and rapidly reduce encryption delay (Fig. 11) to meet the RTT

**TABLE 2.** Decryption time average and standard deviation.

Operation mode type	Data Size	Standard deviation
PCB	5KB	0.005954361648439
ECB		0.007797524262989
GCM		0.012532005396350

**FIGURE 12.** Decryption performance.

requirement. However, the system is unable to compensate the RTT changes without PRA, hence RTT steadily increases although encryption delay remains constant. Thus, PRA can not only vary the delay according to network requirements, but it can also sensitively detect current network status and dynamically reform the patterns to meet network requirements. Therefore, the experiment verified that the proposed rate adaptation algorithm can improve network performance by dynamically modifying the pattern.

## VII. COMPARISON

The operation mode we proposed has an independent structure similar to the ECB. This independent structure has the advantage of being able to provide fast performance. In terms of security, GCM is an operating mode that has already been recognized by a well-known organization (NIST). This is why we compared the operation mode we proposed with ECB and GCM among many other modes.

We tested ECB, PCB, and GCM performance using PCB and ECB with data size = 1, 5, and 10. Average PCB times to decrypt these  $\approx 10$ , 49, and 93 ms; whereas ECB average times  $\approx 11$ , 56, and 109 ms; and GCM average times = 40, 183, and 373 ms, respectively. TABLE 2 shows that PCB, ECB, and GCM standard deviations = 5, 8, and 13 ms, respectively, for data size = 5 kB. The PCB pattern was assigned with ratio 3:1.7:1 when calculating encryption/decryption time and security according to the key length. Fig. 12 shows that PCB was approximately four times faster to decrypt than GCM, despite GCM structure being close to stream cipher. This was due to GCM only needing to perform XOR after pre-calculating a counter value, similar to CTR mode. However, although it is possible to calculate the counter value in advance, if continuous communication is implemented with unpredictably varying

message size, counter value delay will eventually occur. Since GCM includes an integrity authentication process as one of the Authenticated Encryption with Associated Data (AEAD) algorithms, it must also perform MAC computation, and GCM's integrity authentication process cannot perform parallel operations.

## VIII. CONCLUSION

This paper proposed a novel cryptography scheme, PCB, that included patterned operation mode, message integrity authentication, and mutual authentication. PCB utilizes current symmetric key ciphers to alternately encrypt/decrypt messages, hence reducing delays and enhancing security. Since PCB can exploit any symmetric key cipher using shared keys, the design can be generically applied when any new symmetric key cipher is proposed. We also proposed mathematical models to derive PCB operational performance, and proposed optimization and rate adaptation algorithms. We verified the proposed cryptography and rate adaptation algorithm performance, and showed that the proposed scheme could be applied for various network requirements.

Future studies will investigate PRA effects and apply the proposed algorithm to HyperText Transfer Protocol (HTTP). We expect that the proposed approach will inspire future cryptography research and be a keystone for future security systems.

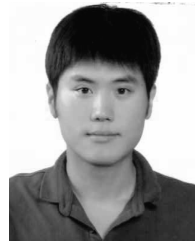
## REFERENCES

- [1] W. Abdallah, S. K. Lee, H. Kim, and N. Boudriga, "Adaptive QoS and security for video transmission over wireless networks: A cognitive-based approach," in *Proc. Int. Conf. Algorithms Archit. Parallel Process.* Dalian, China: Springer, 2014, pp. 138–151.
- [2] D. J. Bernstein, "Introduction to post-quantum cryptography," in *Post-Quantum Cryptography*. Cham, Switzerland: Springer, 2009, pp. 1–14.
- [3] A. Biryukov, O. Dunkelman, N. Keller, D. Khovratovich, and A. Shamir, "Key recovery attacks of practical complexity on aes-256 variants with up to 10 rounds," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* French Riviera, France: Springer, 2010, pp. 299–319.
- [4] L. Chen, L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, "Report on post-quantum cryptography," U.S. Dept. Commerce, Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. NISTIR 8105, 2016.
- [5] N. T. Courtois and W. Meier, "Algebraic attacks on stream ciphers with linear feedback," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Warsaw, Poland: Springer, 2003, pp. 345–359.
- [6] J. Daemen, "Cipher and hash function design strategies based on linear and differential cryptanalysis," Ph.D. dissertation, Dept. Comput. Secur. Ind. Cryptogr., KU Leuven, Leuven, Belgium, Mar. 1995.
- [7] J. Daemen and V. Rijmen, *The Design of Rijndael: AES—The Advanced Encryption Standard*. Cham, Switzerland: Springer, 2013.
- [8] W. Diffie, P. C. Van Oorschot, and M. J. Wiener, "Authentication and authenticated key exchanges," *Des., Codes Cryptogr.*, vol. 2, no. 2, pp. 107–125, Jun. 1992.
- [9] M. Dworkin, "Recommendation for block cipher modes of operation. Methods and techniques," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. ADA400014, 2001.
- [10] M. J. Dworkin, "Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMACINIST," NIST, Gaithersburg, MD, USA, Tech. Rep. 800-38D, 2007.
- [11] S. Gueron, A. Langley, and Y. Lindell. (2019). [Online]. Available: <https://csrc.nist.gov/CSRC/media/Projects/Block-Cipher-Techniques/documents/BCM/proposed-modes/aes-gcm-siv/aes-gcm-siv-may2019.pdf>
- [12] K. Hausken and G. Levitin, "Review of systems defense and attack models," *Int. J. Performability Eng.*, vol. 8, no. 4, pp. 355–366, 2012.

- [13] N. Hoque, M. H. Bhuyan, R. C. Baishya, D. K. Bhattacharyya, and J. K. Kalita, "Network attacks: Taxonomy, tools and systems," *J. Netw. Comput. Appl.*, vol. 40, pp. 307–324, Apr. 2014.
- [14] M. Salter and R. Housley, "Suite B profile for transport layer security (TLS)," IETF Request Comments 5430, 2009.
- [15] K. T. Huang, "A novel structure with dynamic operation mode for symmetric-key block ciphers," *Int. J. Netw. Secur. Appl.*, vol. 5, no. 1, pp. 17–36, Jan. 2013.
- [16] K. Igoe and J. Solinas, "AES Galois counter mode for the secure shell transport layer protocol," IETF Request Comments 5647, 2009.
- [17] S. Kremer and M. D. Ryan, "Analysing the vulnerability of protocols to produce known-pair and chosen-text attacks," *Electron. Notes Theor. Comput. Sci.*, vol. 128, no. 5, pp. 87–104, May 2005.
- [18] D. Kwon, J. Kim, S. Park, S. H. Sung, Y. Sohn, J. H. Song, Y. Yeom, E.-J. Yoon, S. Lee, and J. Lee, "New block cipher: Aria," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Springer, 2003, pp. 432–445.
- [19] L. Law and J. Solinas, "Suite B cryptographic suites for IPsec," IETF Request Comments 4869, 2007.
- [20] H. J. Lee, S. J. Lee, J. H. Yoon, D. H. Cheon, and J. I. Lee, "The SEED encryption algorithm," CiteSeerx, Tech. Rep., 2005.
- [21] J. Lennox, "Encryption of header extensions in the secure real-time transport protocol (SRTP)," IETF Request Comments 6904, 2011.
- [22] H. Lipmaa, P. Rogaway, and D. Wagner, "CTR-mode encryption," in *Proc. 1st NIST Workshop Modes Operation*, vol. 39. Citeseer, 2000.
- [23] D. McGrew and J. Viega, "The Galois/counter mode of operation (GCM)," *NIST Modes Oper. Process*, vol. 20, 2004.
- [24] C. J. Mitchell, "Cryptanalysis of two variants of PCBC mode when used for message integrity," in *Proc. Australas. Conf. Inf. Secur. Privacy.* Brisbane, QLD, Australia: Springer, 2005, pp. 560–571.
- [25] S. Mrdovic and B. Perunicic, "Kerckhoffs' principle for intrusion detection," in *Proc. 13th Int. Telecommun. Netw. Strategy Planning Symp.*, Sep. 2008, pp. 1–8.
- [26] R. Pereira and R. Adams, "The ESP CBC-mode cipher algorithms," Tech. Rep., 1998.
- [27] B. Preneel, M. Nuttin, V. Rijmen, and J. Buelens, "Cryptanalysis of the CFB mode of the DES with a reduced number of rounds," in *Proc. Annu. Int. Cryptol. Conf.* Santa Barbara, CA, USA: Springer, 1993, pp. 212–223.
- [28] E. Rescorla, "The transport layer security (TLS) protocol version 1.3," IETF Request Comments 8446, 2018.
- [29] M. Y. Rhee, *Cryptography and Secure Communications*. New York, NY, USA: McGraw-Hill, 1993.
- [30] Y. Sovyn, V. Khoma, and M. Podpora, "Comparison of three CPU-core families for IoT applications in terms of security and performance of AES-GCM," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 339–348, Jan. 2020.
- [31] Y. Xiao, B. Sun, H.-H. Chen, S. Guizani, and R. Wang, "NIS05-1: Performance analysis of advanced encryption standard (AES)," in *Proc. IEEE Globecom*, Nov. 2006, pp. 1–5.



**SEOUNGHWAN OH** was born in March 1992. He received the B.S.E. degree in electrical engineering from Korea University, Sejong, South Korea, in 2018, where he is currently pursuing the M.S.E. degree with the School of Electrical Engineering. His research interests include community wireless networks, network modeling and simulations, network security, cryptography, and cryptocurrency.



**SEONGJOON PARK** was born in September 1990. He received the B.S.E. degree in electrical engineering from Korea University, Seoul, South Korea, in 2015, where he is currently pursuing the Ph.D. degree with the School of Electrical Engineering. His research interests include community wireless networks, network modeling and simulations, and multiple UAV applications.



**HWANGNAM KIM** received the B.S.E. degree in computer engineering from Pusan National University, Busan, South Korea, in 1992, the M.S.E. degree in computer engineering from Seoul National University, Seoul, South Korea, in 1994, and the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign, in 2004. He is currently a Professor with the School of Electrical Engineering, Korea University, Seoul. His research interests are in the areas of wireless networks, unmanned aerial systems (UASs), UAS traffic management (UTM), counter UAS systems, the Internet of Things, and cyber-physical systems.

•••