Received 3 February 2024, accepted 21 February 2024, date of publication 7 March 2024, date of current version 19 March 2024.

*Digital Object Identifier 10.1109/ACCESS.2024.3374645*

RESEARCH ARTICLE

# Compact Hybrid Signature for Secure Transition to Post-Quantum Era

**HEE-YONG KWON, INDRA BAJUNA, AND MUN-KYU LEE, (Member, IEEE)**
Department of Electrical and Computer Engineering, Inha University, Incheon 22212, South Korea

Corresponding author: Mun-Kyu Lee (mklee@inha.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government [Ministry of Science and ICT (MSIT)] under Grant RS-2023-00209294, in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korea Government (MSIT) under Grant RS-2022-00155915, and in part by the Inha University Research Grant (2023).

**ABSTRACT** Recent advances in quantum-computing technology have threatened the security of classical cryptographic algorithms. This initiated research on Post-Quantum Cryptography (PQC), and the National Institute of Standards and Technology (NIST) PQC standardization is in progress. Coping with the current situation in which the security of existing cryptographic algorithms is already in question and that of new cryptographic algorithms is not yet certain, there has been active research on hybrid schemes combining two algorithms such that the security of the combined scheme is based on both underlying algorithms. For digital signatures, a naive solution for a hybrid scheme is to simply concatenate a classical signature and a quantum-resistant signature. In this paper, however, we propose a compact hybrid signature construction method that combines two randomized signatures such that the size of the combined signature is shorter than that of naive concatenation. Our construction allows for selective verification, which provides backward compatibility and conformance with existing regulations. We demonstrate the feasibility of the proposed method by combining ECDSA P-256 and Falcon-512, which are representative classical and post-quantum signature schemes, respectively. We prove that the combined signature is existentially unforgeable against an adaptive chosen-message attack, even if one of the underlying signature schemes is completely broken and only the other one remains secure. Through experiments on a desktop PC and Raspberry Pi 3 Model B, we verify that the proposed method effectively reduces the combined signature size with negligible computational overhead. Our experimental results demonstrate the proposed method is also applicable to PQC-PQC combinations.

**INDEX TERMS** Hybrid signature, post-quantum cryptography, ECDSA, Falcon.

## I. INTRODUCTION

Recent advances in quantum computers and quantum algorithms have posed serious risks to existing cryptographic algorithms. For example, Grover's quantum search algorithm [1] is expected to substantially accelerate the brute forcing of symmetric cryptographic schemes, and Shor's algorithm [2] is expected to break standard public-key cryptographic schemes based on prime factorization and discrete logarithm problems. Although they were proposed in 1996 and 1994, respectively, standard cryptographic algorithms such as RSA and DSA [3] have been used securely for several decades because no practical quantum computers have been developed thus far to effectively conduct quantum algorithms. However, the threats posed by quantum algorithms are being realized as quantum computer technology is rapidly developing. In 2022, IBM unveiled a 433-qubit quantum processor, Osprey, with the aim of developing a quantum processor with more than 4,000 qubits by 2025 [4]. In 2023, Google provided experimental results for suppressing quantum errors, although they introduced more qubits [5]. These results demonstrate the possibility of developing practical quantum computers in the near future. In addition, Intel provided better accessibility to quantum computing by publishing the Quantum Software Development Kit version 1.0 in 2023 [6].

The associate editor coordinating the review of this manuscript and approving it for publication was Engang Tian.

Owing to the potential threats of quantum computers, the American National Institute of Standards and Technology (NIST) started Post-Quantum Cryptography (PQC) standardization to standardize quantum-resistant public-key cryptographic algorithms. In 2016, the first round of this standardization was initiated with 45 key encapsulation mechanism (KEM) candidates and 19 digital signature algorithm candidates. After the second and third rounds of evaluation, CRYSTALS-Kyber [7] was selected as the KEM for standardization, and CRYSTALS-Dilithium [8], Falcon [9], and SPHINCS+ [10] were selected as the digital signature algorithms in 2022. In addition, four alternative KEM candidates advanced to the fourth round for future standardization.

It is desirable that the transition to PQC be completed as soon as possible. Mosca's inequality [11] is a well-known illustration of the risk of delay during this transition. Let $x$ be the life span of the data to be kept secure, $y$ be the time required for PQC transition, and $z$ be the time remaining until a large-scale quantum computer becomes available for cryptanalysis. If $z < x + y$, then the data are no longer secure. In particular, we may consider the store-now-decrypt-later (SNDL) attack against a classical encryption algorithm, in which an adversary stores a ciphertext containing valuable data and decrypts it with a quantum computer later [12]. Nevertheless, we must be conservative and cautious when adopting new cryptographic algorithms. For example, the SIKE algorithm [13] was recently broken [14], even though it had already advanced to the fourth round of NIST PQC standardization after a significant amount of analysis.

The two conflicting goals mentioned above, that is, a fast but conservative transition, can be achieved through hybrid cryptographic algorithms. A hybrid cryptographic algorithm combines two distinct algorithms. The two component algorithms are used simultaneously and the security of the combined algorithm is reduced to that of the component algorithms [15]. Therefore, the hybrid scheme should be secure if at least one of the two underlying components remains secure. Thus, the hybrid approach is effective when the security of a new primitive is not yet certain; however, the security of an old primitive is already in question [16].

Active research has been conducted on hybrid schemes that combine classical and PQC algorithms. For hybrid KEMs, Bos et al. proposed the hybrid KEM combining the Learning With Errors (LWE)-based Frodo algorithm and the classical Elliptic Curve Diffie-Hellman (ECDH) key exchange protocol in 2016 [17]. In 2021, Azarderakhsh et al. presented the first hardware implementation of a hybrid KEM comprising SIKE and ECDH [18]. For hybrid digital signatures, Komarova et al. suggested two hybrid digital signature schemes in 2021 that combine CRYSTALS-Dilithium with either the Rabin or Elgamal signature scheme [19]. Recently, Bindel et al. proposed various constructions for hybrid digital signature [15].

In this study, we focus on hybrid digital signatures. Let $\sigma_A(m)$ be a digital signature on message $m$ using the signature scheme $A$. It is obvious that simple concatenation $\sigma_A(m)\|\sigma_B(m)$ of individual signatures of scheme $A$ and scheme $B$ can be an effective hybrid signature that builds on the security of both $A$ and $B$. This can also be realized through nesting one method in the other, e.g., $\sigma_A(m)\|\sigma_B(\sigma_A(m))$. However, with these approaches, the signature length becomes the sum of those of the underlying signatures, which is undesirable in terms of the utilization of limited resources, such as network bandwidth. Therefore, a few alternatives have been proposed to merge signatures and redefine the signature generation and verification processes of the two underlying schemes [15]. However, significantly revising the original form of the individual signatures may raise backward compatibility and regulatory conformance issues. Therefore, we aim to design a hybrid digital signature scheme that is compatible with each underlying individual signature scheme.

In this study, we propose a new hybrid digital signature construction method that combines two randomized digital signature schemes. Our construction satisfies the following properties:

- The constructed hybrid signature is compact. That is, the hybrid signature length is less than the sum of those of the individual signatures. Furthermore, the computational overhead for this optimization is almost negligible.
- Selective verification is possible. When a hybrid signature $\sigma_{AB}(m)$ combining two methods $A$ and $B$ are given, a system that recognizes only the signature scheme $A$ can verify the signature, guaranteeing an equivalent level of security to verify a single signature $\sigma_A(m)$. The same holds for the other scheme $B$. This property provides backward compatibility for a legacy system that only recognizes the classical signature scheme. It may also conform to the current regulations that have not fully standardized the PQC scheme yet. It may also enable a brand-new system to ignore the signature part related to an out-of-date signature scheme.[1]
- The proposed signature scheme is existentially unforgeable under an adaptive chosen message attack. The hybrid signature is secure even when one of the two component signature schemes is completely broken if only the other component remains secure.

The remainder of this paper is organized as follows. In Section II, we recall the formal definition of a digital signature scheme and provide a slightly modified version of the randomized signature scheme of interest. In Section III, we present a general framework for constructing

[1]We remark that as stated in [15], there is inherent mutual exclusion between backward compatibility and non-separability. While Bindel et al. [15] chose non-separability, we chose backward compatibility. Thus, non-separability is not a technical goal in this paper, but we assume that it can be achieved with a well-defined security policy of an organization.

a randomized hybrid signature scheme. Although we explain hybridization in the context of a combination of classical and PQC schemes, our construction is general, which can be used for classical-classical and PQC-PQC combinations. Section IV presents a specific example of the proposed hybrid method using two representative signature schemes: an Elliptic Curve Digital Signature Algorithm (ECDSA) as a classical signature scheme and Falcon as a post-quantum signature scheme. In Section V, we prove that the proposed signature scheme is existentially unforgeable under an adaptive chosen message attack, even if one of the underlying signature schemes is completely broken. For example, the hybrid ECDSA-Falcon is at least as secure as Falcon, even when ECDSA is completely broken. Section VI verifies the compactness of the proposed method and demonstrates that the computational overhead of our hybridization is almost negligible, on both a desktop PC and Raspberry Pi. Section VII discusses possible applications, extensions and limitations of the proposed method. Finally, Section VIII concludes the paper.

## II. PRELIMINARIES: DIGITAL SIGNATURE

A digital signature is used to guarantee the security properties of transmitted data, such as authenticity, integrity, and non-repudiation. Generally, a digital signature scheme comprises the following four algorithms:

1) $\mathsf{ParamGen}\left(1^{\lambda}\right)$: The system parameters are generated based on security parameter $\lambda$.
2) $\mathsf{KeyGen}\left(1^{\lambda}\right)$: A key pair $(sk, pk)$ is generated, where $sk$ is a private key (a.k.a. secret key) and $pk$ is a public key.
3) $\mathsf{Sign}(sk, m)$: Given a private key $sk$ and a message $m$, a signature $\sigma$ on $m$ is generated.
4) $\mathsf{Ver}(pk, m, \sigma)$: Given a public key $pk$, a message $m$, and a signature $\sigma$, it is verified whether $\sigma$ is a valid signature on $m$ signed by the legitimate user of $sk$ corresponding to $pk$.

Among classical digital signature schemes, three classes are well known: (1) prime factorization-based schemes (such as RSA [20] and its probabilistic version, RSA-PSS [20]), (2) DLP-based schemes (such as Elgamal signature [21] and DSA [22]), and (3) elliptic curve discrete logarithm problem (ECDLP)-based schemes (such as ECDSA [23] and Modified ECDSA [24]). For post-quantum digital signature schemes, new signature schemes such as MQDSS [25], qTESLA [26], picnic [27], Falcon [9], CRYSTALS-Dilithium [8], etc., have recently been proposed. In this study, we consider randomized signature schemes whose $\mathsf{Sign}$ involves random value generation as follows (We denote this type of $\mathsf{Sign}$ as $\mathsf{Sign}'$):

1) $\mathsf{Sign}'$ takes $sk$ and $m$ as its inputs, where $sk$ is a private key of legitimate user and $m$ is a message to be signed.
2) $\mathsf{Sign}'$ generates a uniformly random value $k$, and transforms $k$ into $r$ by computing $r = T(k)$. $T$ may be an identity function, where the transformation is just an assignment, i.e., $r = k$.
3) $\mathsf{Sign}'$ calls subroutine $\mathsf{rSign}$ with $sk$, $m$, $k$, and $r$.
   a) $\mathsf{rSign}$ generates the remaining part of signature, $s = \Phi(sk, m, k, r)$.
   b) $\mathsf{rSign}$ returns a pair of $(r, s)$.
4) $\mathsf{Sign}'$ outputs a signature $\sigma = (r, s)$.

Following the above conditions, in this study we consider signature schemes that include $r$ transformed from $k$ as part of the signature, such as the Elgamal signature [21], DSA [22], ECDSA [23], Modified ECDSA [24], picnic [27], Falcon [9], etc. In this study, $S$ and $S'$ denote the signature schemes composed of $\langle\mathsf{ParamGen}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver}\rangle$ and $\langle\mathsf{ParamGen}, \mathsf{KeyGen}, \mathsf{Sign}', \mathsf{Ver}\rangle$, respectively.

## III. CONSTRUCTION OF HYBRID SIGNATURE

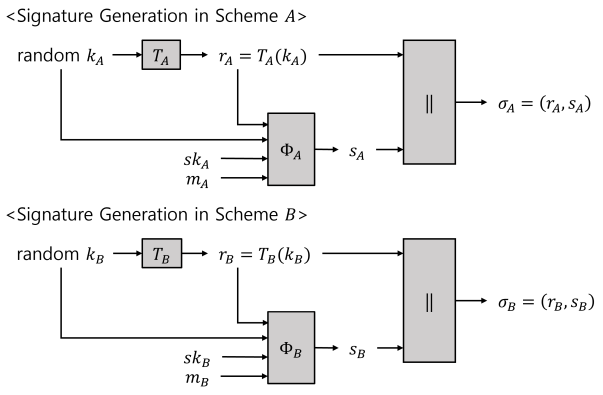


**FIGURE 1. Signature generation processes of two individual randomized signature schemes, *A* and *B*.**

In this section, we propose a general framework for constructing a hybrid signature scheme by using two individual randomized signature schemes. Fig. 1 shows two signature schemes that generate signatures separately. Let $u$ be a signature scheme ($u = A$ or $B$). The signature generation process $\mathsf{Sign}'_u$ of $u$ generates a uniformly random value, $k_u$, and computes $r_u = T_u(k_u)$. Subsequently, its subroutine $\mathsf{rSign}_u$ computes $s_u = \Phi_u(sk_u, m_u, k_u, r_u)$ for a message $m_u$ with the private key $sk_u$. When $\mathsf{rSign}_u$ returns the pair $(r_u, s_u)$, $\mathsf{Sign}'_u$ outputs it as the signature $\sigma_u = (r_u, s_u)$.

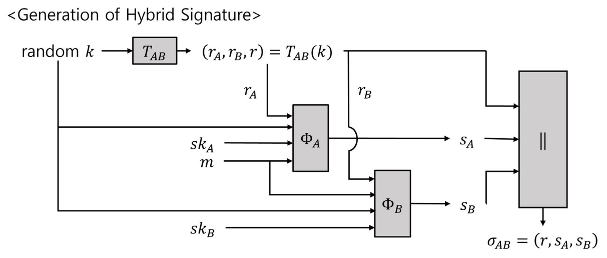

<Generation of Hybrid Signature>
random $k$
$T_{AB}$
$(r_A, r_B, r) = T_{AB}(k)$
$r_A$
$r_B$
$sk_A$
$m$
$\Phi_A$
$s_A$
$\Phi_B$
$s_B$
$sk_B$
||
$\sigma_{AB} = (r, s_A, s_B)$

**FIGURE 2. Signature generation process of the proposed hybrid signature scheme.**

Fig. 2 shows the signature generation process $\mathsf{Sign}'_{AB}$ of the hybrid signature scheme that combines schemes $A$ and $B$.

$\mathsf{Sign}'_{AB}$ generates a uniformly random value $k$ and computes the triple $(r_A, r_B, r) = T_{AB}(k)$ using the merged transformation function $T_{AB}$. $\mathsf{rSign}_A$ and $\mathsf{rSign}_B$ computes $s_A = \Phi_A(sk_A, m, k, r_A)$ and $s_B = \Phi_B(sk_B, m, k, r_B)$. Finally, $\mathsf{Sign}'_{AB}$ outputs a signature $\sigma_{AB} = (r, s_A, s_B)$.

The merged transformation $T_{AB}$ should be designed such that $r_A$ and $r_B$ are derived from $r$ uniquely and efficiently. Without loss of generality, we assume that $len(r_A) < len(r_B)$, where $len(x)$ is the length of the bit string $x$. Then, we define $r$ and $r_A$ such that they satisfy $r = r_A||r_\tau$, where $r_\tau$ is a random bit string with length $len(r_B) - len(r_A)$ and $||$ is concatenation. Next, we define $r_B$ as $r_B = f(r)$, with a bijective function $f : \{0,1\}^l \rightarrow \{0,1\}^l$ satisfying $f^{-1}(f(x)) = x$, where $l = len(r_B)$.

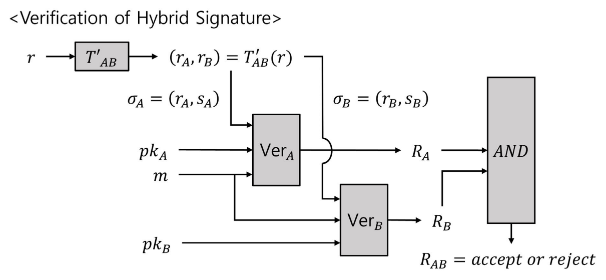


**FIGURE 3. Signature verification process of the proposed hybrid signature scheme.**

Fig. 3 illustrates the signature verification process $\mathsf{Ver}_{AB}$ of the hybrid signature scheme. $\mathsf{Ver}_{AB}$ takes the signature $\sigma_{AB} = (r, s_A, s_B)$ to be verified and recovers $r_A$ and $r_B$ from $r$ using a transformation function $T'_{AB}$ corresponding to $T_{AB}$. Now, $(r_A, s_A)$ and $(r_B, s_B)$ can be verified separately using $\mathsf{Ver}_A$ and $\mathsf{Ver}_B$ as follows: The verification algorithm $\mathsf{Ver}_A$ for the signature scheme $A$ verifies $\sigma_A = (r_A, s_A)$ for $m$ using public key $pk_A$, and $\mathsf{Ver}_B$ for the signature scheme $B$ verifies $\sigma_B = (r_B, s_B)$ for $m$ using public key $pk_B$. $\mathsf{Ver}_{AB}$ combines the verification results of $\mathsf{Ver}_A$ and $\mathsf{Ver}_B$, and rejects signature $\sigma_{AB}$ if one of the two signatures was not successfully verified.

Because the verification process of the proposed scheme accepts a hybrid signature only when both individual signatures are verified, we may expect the proposed scheme to guarantee the maximum security level among individual signature schemes $A$ and $B$. This implies that the proposed scheme is secure even if one of the signature schemes is broken. This is formally proven in Section V.

## IV. HYBRID FALCON-ECDSA

In this section, we present a specific example of the proposed hybrid signature construction using Falcon [9] and ECDSA [23] as the underlying PQC and classical signature schemes, respectively.

### A. PRELIMINARIES

#### 1) ELLIPTIC CURVE DIGITAL SIGNATURE ALGORITHM (ECDSA)

ECDSA [22], [23] is a standard digital signature algorithm based on elliptic curves [28], [29]. The security of

**Algorithm 1** ECDSA Signature Generation

```
Input: Elliptic curve $E(\mathbb{F}_q)$, private key $d$, message $m$
Output: Signature $\sigma = (r, s)$
1: Select $k \leftarrow_R [1, n-1]$
2: Compute $kP = (x_1, y_1)$ and convert $x_1$ to an integer $\bar{x}_1$
3: Compute $r = \bar{x}_1 \bmod n$. If $r = 0$, then go to step 1
4: Compute $e = \mathrm{H}(m)$
5: Compute $s = k^{-1}(e + dr) \bmod n$. If $s = 0$, then go to
   step 1
6: Return $(r, s)$
```

**Algorithm 2** ECDSA Signature Verification

```
Input: Elliptic curve $E(\mathbb{F}_q)$, public key $Q$, message $m$,
signature $\sigma = (r, s)$
Output: Accept or reject
1: Verify that $r$ and $s$ are integers in the interval $[1, n-1]$.
   If any verification fails, then return ("Reject the signature")
2: Compute $e = \mathrm{H}(m)$
3: Compute $w = s^{-1} \bmod n$
4: Compute $u_1 = ew \bmod n$ and $u_2 = rw \bmod n$
5: Compute $X = u_1 P + u_2 Q$
6: If $X = \infty$ then return ("Reject the signature")
7: Convert the $x$-coordinate $x_1$ of $X$ to an integer $\bar{x}_1$;
   Compute $v = \bar{x}_1 \bmod n$
8: If $v = r$ then return ("Accept the signature");
   Else return ("Reject the signature")
```

ECDSA is based on the computational hardness assumption for solving the Elliptic Curve Discrete Logarithm Problem (ECDLP) [28], [29]. Let $E$ be an elliptic curve over a finite field $\mathbb{F}_q$ and $P \in E(\mathbb{F}_q)$ be a point of order $n$. ECDLP is the problem of determining an integer $k$ that satisfies $Q = kP$ $(0 \leq k \leq n-1)$ when $P$ and $Q$ are given. It is widely believed that this problem cannot be solved in polynomial time. Compared with the RSA signature scheme, the size of the ECDSA signature is significantly shorter at the same security level. Algorithm 1 and Algorithm 2 illustrate the signature generation and verification processes of ECDSA, respectively.

Algorithm 1 takes an elliptic curve, a private key, and a message as inputs. First, it chooses a uniformly random integer $k \in [1, n-1]$. In line 2, it performs a point multiplication, $kP = (x_1, y_1)$, and converts $x_1$ into an integer $\bar{x}_1$. The algorithm computes $r$ using modular reduction on $\bar{x}_1$ with modulus $n$. If $r = 0$, it restarts the signature generation process. Next, it computes $s$ for the hashed message $e$ in line 5. If $s = 0$, the signature generation process is restarted. Finally, signature $\sigma = (r, s)$ is returned. Lines 1 to 3 of Algorithm 1 correspond to step 2 of $\mathsf{Sign}'$. Lines 2 and 3 correspond to the transformation of $k$ into $r$; that is, $r = T(k)$ of $\mathsf{Sign}'$. Lines 4 and 5 correspond to $s = \Phi(sk, m, k, r)$ of $\mathsf{rSign}$ in step 3 of $\mathsf{Sign}'$, where $sk = d$.

Algorithm 2 for signature verification takes an elliptic curve, a public key, a message, and a signature as inputs. First, it checks whether $r$ and $s$ are within the interval $[1, n-1]$. If any of the values are not within the interval, the verification fails. The algorithm then computes $u_1 = ew \bmod n$ and $u_2 = rw \bmod n$ using a hashed message $e$, $w = s^{-1} \bmod n$, and $r$. Next, it recovers point $X = u_1P + u_2Q$. If point $X$ is a point at infinity, the verification fails. Finally, the algorithm computes $v = \bar{x}_1 \bmod n$ where $\bar{x}_1$ is an integer converted from the $x$ coordinate of $X$. If $v = r$, then the signature is accepted; otherwise, it is rejected.

### 2) FALCON

The Falcon signature scheme [9] was proposed by Fouque et al. in 2018 to ensure the security of digital signatures in post-quantum computing environments. It is a promising digital signature scheme and has become a candidate for Round 4 of the NIST post-quantum cryptography standardization process [30]. Falcon is an attractive solution for resource-constrained environments as its signature generation and verification speeds are very fast. Falcon uses the GPV framework, a generic framework for building a secure hash-and-sign lattice-based signature scheme [31]. In addition, it was improved by combining the GPV framework with NTRU lattices [32] and applying fast Fourier sampling [33]. The security of Falcon is based on the computational hardness assumption for the short integer solution (SIS) problem [34] over NTRU lattices [35]. Let matrix $\mathbf{A} \in \mathbb{Z}_{q'}^{n' \times m'}$ be chosen uniformly at random, where $n'$, $m'$, and $q'$ are positive integers. The SIS problem is finding a nonzero integer vector $\mathbf{z} \in \mathbb{Z}^{m'}$ of the norm $||\mathbf{z}|| \leq \beta$ such that $\mathbf{Az} = \mathbf{0} \in \mathbb{Z}_{q'}^{n'}$, where $\beta$ is a positive real number [36]. Algorithm 3 and Algorithm 4 illustrate the signature generation and verification processes of Falcon, respectively. There are multiple possible parameter sets for Falcon. For simplicity, we explain the processes for Falcon-512 with a 120-bit classical security level and 108-bit quantum security level, following the original description in [9].

Algorithm 3 takes message $m$, private key $sk = (\hat{\mathbf{B}}, \mathsf{T})$, and acceptance bound $\lfloor \beta^2 \rfloor$ as inputs. First, in line 1, it generates a uniformly random salt $r$ from $\{0,1\}^{320}$. The salt $r$ is directly used as part of the signature in line 12. Algorithm 3 then hashes the concatenated string $(r||m)$ to a point $c$. Next, the algorithm computes a preimage $\mathbf{t}$ of $c$ using the fast Fourier transform (FFT), and it repeats fast Fourier sampling, $\mathbf{z} \leftarrow \mathsf{ffSampling}_{n'}(\mathbf{t}, \mathsf{T})$, until it finds a sufficiently short vector $\mathbf{s} = (\mathbf{t} - \mathbf{z})\hat{\mathbf{B}}$ satisfying $||\mathbf{s}||^2 \leq \lfloor \beta^2 \rfloor$. Using inverse FFT to $\mathbf{s}$, it obtains two short polynomials $(s_1, s_2)$ such that $s_1 + s_2h = c \bmod (\phi(x), q')$, where $\phi(x) = x^{n'} + 1$ is a cyclotomic polynomial and $h$ is the public key. Finally, $s_2$ is compressed to a bitstring $s$, and the algorithm outputs a Falcon signature $\sigma = (r, s)$. If $len(s) \neq 8 \cdot sbytelen - 328$, the algorithm goes back to line 4 and repeats the process. Considering the description of $\mathsf{Sign}'$ in Section II, we can say that $r$ in line 1 is the random value $k$ in step 2 of $\mathsf{Sign}'$, and $T$ is the identity function. Lines 2–11 correspond to $\Phi(sk = (\hat{\mathbf{B}}, \mathsf{T}), m, k = r, r)$.

**Algorithm 3** Falcon Signature Generation

```
Input: A message $m$, a private key $sk = (\hat{\mathbf{B}}, \mathsf{T})$, a bound $\lfloor \beta^2 \rfloor$
Output: Signature $\sigma = (r, s)$
 1: $r \leftarrow \{0,1\}^{320}$ uniformly.
 2: $c \leftarrow \mathrm{HashToPoint}(r||m, q', n')$
 3: $\mathbf{t} \leftarrow \left(-\frac{1}{q'}\mathsf{FFT}(c) \odot \mathsf{FFT}(F), \frac{1}{q'}\mathsf{FFT}(c) \odot \mathsf{FFT}(f)\right)$
 4: do
 5:     do
 6:         $\mathbf{z} \leftarrow \mathsf{ffSampling}_{n'}(\mathbf{t}, \mathsf{T})$
 7:         $\mathbf{s} = (\mathbf{t} - \mathbf{z})\hat{\mathbf{B}}$
 8:     while $||\mathbf{s}||^2 > \lfloor \beta^2 \rfloor$
 9:     $(s_1, s_2) \leftarrow \mathrm{invFFT}(\mathbf{s})$
10:     $s \leftarrow \mathrm{Compress}(s_2, 8 \cdot sbytelen - 328)$
        ▷ Remove 1 byte for the header, and 40 bytes for $r$
11: while $(s = \perp)$
12: return $\sigma = (r, s)$
```

**Algorithm 4** Falcon Signature Verification

```
Input: A message $m$, a signature $\sigma = (r, s)$, a public key $pk = h \in \mathbb{Z}_{q'}[x]/\phi(x)$, a bound $\lfloor \beta^2 \rfloor$
Output: Accept or reject
 1: $c \leftarrow \mathrm{HashToPoint}(r||m, q', n')$
 2: $s_2 \leftarrow \mathrm{Decompress}(s, 8 \cdot sbytelen - 328)$
 3: if $(s_2 = \perp)$, then
 4:     reject
    ▷ Reject invalid encodings
 5: $s_1 \leftarrow c - s_2h \bmod (\phi(x), q')$
    ▷ $s_1$ should be normalized between $\lceil -\frac{q'}{2} \rceil$ and $\lfloor \frac{q'}{2} \rfloor$
 6: if $||(s_1, s_2)||^2 \leq \lfloor \beta^2 \rfloor$ then
 7:     accept
 8: else
 9:     reject
    ▷ Reject signatures that are too long
```

Algorithm 4 for signature verification takes message $m$, signature $\sigma = (r, s)$, public key $pk = h$, and acceptance bound $\lfloor \beta^2 \rfloor$ as inputs. First, it computes point $c$ by hashing a concatenated string $(r||m)$, then $s$ is decompressed into a polynomial $s_2$. If $len(s) \neq 8 \cdot sbytelen - 328$ or the decompression fails, the signature is rejected in line 4. Otherwise, the algorithm computes $s_1 = c - s_2h \bmod (\phi(x), q')$. Finally, if $||(s_1, s_2)||^2 \leq \lfloor \beta^2 \rfloor$, then the signature is accepted; otherwise, it is rejected.

## B. HYBRID SIGNATURE USING FALCON AND ECDSA

In this section, we use the proposed framework described in Section III to construct a hybrid signature scheme using Falcon and ECDSA. Several possible parameter combinations exist based on the desired security level. In this study, we considered a hybrid signature scheme that combines Falcon-512 and ECDSA with a similar classical security level. Therefore, we selected ECDSA P-256, which has a 128-bit classical security level, according to [37].

Let $\sigma_E = (r_E, s_E)$ and $\sigma_F = (r_F, s_F)$ be the ECDSA and Falcon signatures, respectively. In the aforementioned combination of Falcon-512 and ECDSA P-256, the bit lengths of $r_F$ and $r_E$ should be 320 and 256, respectively, to satisfy $len(r_F) > len(r_E)$. Therefore, we first explain the hybrid signature generation and verification processes for the case that $len(r_F) > len(r_E)$, and then explain the other cases, $len(r_F) = len(r_E)$ and $len(r_F) < len(r_E)$. If the hybrid signature is composed of ECDSA and Falcon, it takes the form $(r, s_E, s_F)$. Hence, to enable signature verification using the process illustrated in Fig. 3, it should be possible to derive $r_E$ and $r_F$ from $r$. Considering the description of a merged transformation, $T_{AB}$, in Section III, it is easy to see that ECDSA and Falcon correspond to the signature schemes $A$ and $B$, respectively, because $len(r_E) < len(r_F)$. According to $T_{AB}$, $r$ can be parsed into $r = r_E || r_\tau$, and $r_E$ can be recovered by taking $len(r_E)$ leading bits from $r$. Next, $r_F$ can be computed by $f(r)$ using a bijective function $f$.

We now provide a concrete example of $f$. Pseudo-random permutations (PRPs), such as a block cipher, are good candidates for $f$. However, using a single block of a block cipher may not coincide with the bit lengths $r$ and $r_F$. For example, the block length of the Advanced Encryption Standard (AES) [38] is 128 bits. Therefore, we used the counter mode of AES such that a 320-bit salt $r_F$ can be produced by XORing the 320-bit $r$ and 320-bit keystream generated by AES block encryption operation. Alternatively, other length-preserving transforms, such as format-preserving encryption (FPE) [39], [40] may be used as a PRP.

Algorithm 5 presents the hybrid signature generation process. First, the algorithm takes message $m$ to be signed, Falcon private key $sk$, Falcon bound $\lfloor \beta^2 \rfloor$, elliptic curve $E(\mathbb{F}_q)$, ECDSA private key $d$, and the length parameter $\lambda_P$ of $PRP$ as inputs. In line 1, a uniformly random $k$ is selected from $[1, n-1]$. The algorithm then computes $r_E$ and $s_E$ following the original ECDSA signature generation process in lines 2–5. Using $r_E$, the algorithm generates $r$, the first part of the hybrid signature, in lines 6 and 7. Note that $r_E$ can also be recovered from $r$ when the signature is verified, although $r$ was generated from $r_E$ during the signature generation process.

Next, in line 8 of Algorithm 5, $r_F$ is computed by $PRP$ with salt $r$, key $\{0\}^{\lambda_P}$, and initial counter $\{0\}^{\lambda_P}$. From lines 9 through 18, the Falcon signature part $s_F$ is computed, and the algorithm returns the hybrid signature $(r, s_E, s_F)$. For the hybrid signature generation process $\mathsf{Sign}'_{AB}$, that is, $\mathsf{Sign}'_{EF}$, in Fig. 2, $k$ in line 1 corresponds to the random value $k$ in Fig. 2, lines 2–3 and 6–8 correspond to the transformation, $(r_E, r_F, r) = T_{EF}(k)$. Lines 4–5 and 9–18 correspond to $s_E = \Phi_E(sk_E, m, k, r_E)$ and $s_F = \Phi_F(sk_F, m, k, r_F)$, respectively, where $sk_E = d$, and $sk_F = sk = (\hat{\mathbf{B}}, \mathbf{T})$.

The uniqueness of the random component $r$ is crucial for the security of a randomized signature scheme. For example, it is known in ECDSA that if the same $r$ is used to generate signatures for two distinct messages, the private key can be recovered [41]. Therefore, we must examine the distributions of $r$, $r_E$, and $r_F$ generated by the above construction. It is straightforward that the distribution of $r_E$ in Algorithm 5 is the same as that of $r$ in the original ECDSA (Algorithm 1). Therefore, the proposed hybrid signature scheme does not affect the uniqueness of the $r$ in the original ECDSA. We now examine the distribution of $r_F$ in Algorithm 5.

*Lemma 1: Let $prob_r^H(z)$ be the probability that $r = z$ in Algorithm 5, and let $P_r^H = \max_{z \in \{0,1\}^{320}} prob_r^H(z)$. Let $prob_{r_F}^H(z)$ be the probability that $r_F = z$ in Algorithm 5, and let $P_{r_F}^H = \max_{z \in \{0,1\}^{320}} prob_{r_F}^H(z)$. Then, $P_r^H = P_{r_F}^H$.*

*Proof:* $PRP(z, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$ is distinct for each $z \in \{0,1\}^{320}$, based on the bijective property of PRP. Therefore, $prob_r^H(z) = prob_{r_F}^H(PRP(z, \{0\}^{\lambda_P}, \{0\}^{\lambda_P}))$ for all $z$, proving this lemma. □

Let $prob_r^F(z)$ be the probability that $r = z$ in Algorithm 3. The distribution of $r$ in the original Falcon is uniform. That is, $prob_r^F(z) = 1/2^{320}$ for all $z \in \{0,1\}^{320}$, and $P_r^F = \max_{z \in \{0,1\}^{320}} prob_r^F(z) = 1/2^{320}$. The uniqueness of $r_F$ in Algorithm 5 is guaranteed by the following theorem.

*Theorem 1: Let $prob_{r_F}^H(z)$ be the probability that $r_F = z$ in Algorithm 5, and let $P_{r_F}^H = \max_{z \in \{0,1\}^{320}} prob_{r_F}^H(z)$. Then, $P_{r_F}^H \leq 1/2^{317}$.*

*Proof:* Let $prob_{r_E}^H(z)$ be the probability that $r_E = z$ in Algorithm 5. We evaluate $P_{r_E}^H = \max_{z \in \{0,1\}^{256}} prob_{r_E}^H(z)$. Recall that in Algorithm 5, $r_E$ is derived from the $x$ coordinate of point $kP$ on elliptic curve $E(\mathbb{F}_q)$. There are $(n-1)$ candidates of $kP$, and the probability of occurrence of each candidate is $1/(n-1)$. In the worst case, four distinct $k$'s can be mapped onto the same $r_E$ because (1) there exist two distinct points $(x_1, y_1)$ and $(x_1, -y_1)$ for a valid $x_1$, making the relation between $k$ and $x_1$ a 2-to-1 mapping, except at most three extreme cases in which $y_1 = 0$; and (2) there can be two candidates for $x_1$ for a small $r_E$; that is, $r_E = \bar{x}_1$ and $r_E = \bar{x}_1 - n$, resulting in the relation between $\bar{x}_1$ and $r_E$ being a 2-to-1 mapping. (The case $r_E = \bar{x}_1 - 2n$ is not possible because the group order $n$ satisfies $n \leq q + 1 + 1\sqrt{q} \ll 2q$ according to Hasse's theorem for standard NIST prime curves whose cofactor is 1 [42].) This reasoning implies that $P_{r_E}^H \leq 4/(n-1) < 4/2^{255}$. Because the 64 random bits $r_\tau$ are concatenated to $r_E$ to generate $r$ in line 7 of Algorithm 5, $P_r^H = P_{r_E}^H \times 1/2^{64}$. By combining this with Lemma 1, we see that $P_{r_F}^H = P_r^H = P_{r_E}^H \times 1/2^{64} < 1/2^{317}$. □

From Theorem 1, there may be at most three bit losses in the security of $r_F$ by our construction. We may also easily compensate for this loss by slightly extending $r$ and selecting $r_\tau = \{0,1\}^{67}$. Then, the distribution of $r_F$ satisfies $P_{r_F}^H \leq 1/2^{320}$.

Algorithm 6 illustrates the signature verification process of the hybrid signature scheme. The algorithm takes as input a message $m$, a signature $\sigma = (r, s_E, s_F)$, a Falcon public key $pk$, a Falcon bound $\lfloor \beta^2 \rfloor$, an elliptic curve $E(\mathbb{F}_q)$, an ECDSA public key $Q$, and the length parameter

$\lambda_P$ of *PRP*. First, $r$ is parsed into $r_E$ and $r_\tau$, and $r_F$ is recovered by computing $PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$. The algorithm then performs individual ECDSA and Falcon verification algorithms with $(r_E, s_E)$ and $(r_F, s_F)$, respectively. If the verification fails, the hybrid signature is rejected; otherwise, it is accepted as valid.

Next, we examine the selective verifiability of the proposed scheme. We assume that the recipient of the signature does not recognize the new Falcon algorithm, but recognizes only the classical ECDSA signature. Then, the recipient may ignore the third component $s_F$ of the signature and skip steps 2 and 4 in Algorithm 6. This can provide backward compatibility for legacy systems. It may also conform to current regulations that have not fully standardized PQC systems yet. In contrast, assume that the recipient wants to ignore ECDSA whose long-term security is in question owing to the quantum ECDLP algorithms. Then, the recipient may ignore the second component $s_E$ of the signature and skip steps 1 and 3. The computational complexity of this selective verification is almost the same as that of each individual verification, as demonstrated in Section VI.

Algorithm 5 and Algorithm 6 assumed the case $len(r_F) > len(r_E)$. Now, we explain the cases where $len(r_F) = len(r_E)$ and $len(r_F) < len(r_E)$. If $len(r_F) = len(r_E)$, we can eliminate line 6 in Algorithm 5 and revise line 7 to $r \leftarrow r_E$. For signature verification, we can revise line 1 of Algorithm 6 to $r_E \leftarrow r$.

When $len(r_F) < len(r_E)$, the algorithms need to be changed slightly more. In Algorithm 5, lines 6 through 8 can be replaced as follows:

$$\begin{aligned} r &\leftarrow r_E, \\ r_{temp} &\leftarrow PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P}), \\ r_F &\leftarrow Trunc(r_{temp}, len(r_F)), \end{aligned}$$

where $Trunc(r_{temp}, len(r_F))$ is a truncation function that takes the $len(r_F)$ leading bits from $r_{temp}$. For the verification process in Algorithm 6, lines 1 and 2 can be replaced as follows:

$$\begin{aligned} r_E &\leftarrow r \\ r_{temp} &\leftarrow PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P}), \\ r_F &\leftarrow Trunc(r_{temp}, len(r_F)). \end{aligned}$$

Finally, we examine the length-saving effect of the proposed hybrid signature construction. Because the construction above produces a hybrid signature $(r, s_E, s_F)$, we can say that $r$ replaces $(r_E, r_F)$ in the simple concatenation of the two signatures, $(r_E, s_E, r_F, s_F)$. Because $len(r) = \max(len(r_E), len(r_F))$ and the size of $(r_E, r_F)$ is $len(r_E) + len(r_F)$, the proposed scheme reduces the signature size by $\min(len(r_E), len(r_F))$.

## V. SECURITY OF HYBRID SIGNATURE

In this section, we show that the proposed hybrid signature scheme is secure even when one of the two component signature schemes is completely broken if only the other component remains secure. For the security of the signature scheme, we use a slightly modified version of the definition in [20] and [43] as follows:

*Definition 1 (Forger of Signature Scheme): Assume that a forger $\mathcal{F}_S$ for a signature scheme $S = \langle \mathsf{ParamGen}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver} \rangle$ is given the public key computed by $\langle \mathsf{ParamGen}, \mathsf{KeyGen} \rangle$ and a signing oracle access. $\mathcal{F}_S$ can generate a message $m_i$ adaptively based on the previously queried message-signature pairs, $\{(m_1, \sigma_1), \ldots, (m_{i-1}, \sigma_{i-1})\}$, then the signing oracle returns a signature $\sigma_i$ for $m_i$. The goal of $\mathcal{F}_S$ is to output a valid*

**Algorithm 5** Hybrid Signature Generation

**Input:** A message $m$, a Falcon private key $sk$, a Falcon bound $\lfloor \beta^2 \rfloor$, an elliptic curve $E(\mathbb{F}_q)$, an ECDSA private key $d$, the length parameter $\lambda_P$ of *PRP*

**Output:** Signature $\sigma = (r, s_E, s_F)$

```
1:  Select $k \leftarrow_R [1, n-1]$
2:  Compute $kP = (x_1, y_1)$ and convert $x_1$ to an integer $\bar{x}_1$
3:  Compute $r_E \leftarrow \bar{x}_1 \bmod n$. **If** $r_E = 0$, **then** go to step 1
4:  Compute $e \leftarrow \mathrm{H}(m)$
5:  Compute $s_E \leftarrow k^{-1}(e + dr_E) \bmod n$. **If** $s_E = 0$, **then** go
    to step 1
6:  $r_\tau \leftarrow \{0,1\}^{64}$
7:  $r \leftarrow r_E || r_\tau$
8:  $r_F \leftarrow PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$
9:  $c \leftarrow \mathrm{HashToPoint}(r_F || m, q', n')$
10: $\mathbf{t} \leftarrow \left(-\frac{1}{q'}\mathsf{FFT}(c) \odot \mathsf{FFT}(F), \frac{1}{q'}\mathsf{FFT}(c) \odot \mathsf{FFT}(f)\right)$ ▷
    $\mathbf{t} = (\mathsf{FFT}(c), \mathsf{FFT}(0)) \cdot \hat{\mathbf{B}}^{-1}$
11: **do**
12:     **do**
13:         $\mathbf{z} \leftarrow \mathsf{ffSampling}_{n'}(\mathbf{t}, \mathsf{T})$
14:         $\mathbf{s} \leftarrow (\mathbf{t} - \mathbf{z})\hat{\mathbf{B}}$
15:     **while** $||\mathbf{s}||^2 > \lfloor \beta^2 \rfloor$
16:     $(s_1, s_1) \leftarrow \mathrm{invFFT}(\mathbf{s})$
17:     $s_F \leftarrow \mathrm{Compress}(s_2, 8 \cdot sbytelen - 328)$    ▷ Remove
    1 byte for the header, and 40 bytes for $r_F$
18: **while** $(s_F = \perp)$
19: return $\sigma = (r, s_E, s_F)$
```

**Algorithm 6** Hybrid Signature Verification

**Input:** A message $m$, a hybrid signature $\sigma = (r, s_E, s_F)$, a Falcon public key $pk$, a Falcon bound $\lfloor \beta^2 \rfloor$, an elliptic curve $E(\mathbb{F}_q)$, an ECDSA public key $Q$, the length parameter $\lambda_P$ of *PRP*

**Output:** Accept or reject

```
1: Parse $r$ into $r = r_E || r_\tau$
2: $r_F \leftarrow PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$
3: Verify $(r_E, s_E)$ using Algorithm 2.
4: Verify $(r_F, s_F)$ using Algorithm 4.
5: **If** any verification fails, reject the signature; **otherwise**,
   accept the signature
```

*signature $\sigma$ for message $m$ that has never been queried to the signing oracle. The forger $\mathcal{F}_S$ is said to $(t, q_S, \epsilon)$-break the signature scheme $S$ using an adaptive chosen message attack if in at most $t$ processing time and after at most $q_S$ signature queries to the signing oracle, it outputs a valid forgery with a probability of at least $\epsilon$ such that*

$$Pr\left[\begin{array}{l} \langle pk, sk\rangle \leftarrow \langle \mathsf{ParamGen}, \mathsf{KeyGen}\rangle\left(1^{\lambda}\right). \\ for\ i = 1, 2, \ldots, q_S \\ \quad m_i \leftarrow \mathcal{F}_S\left(pk, m_1, \sigma_1, \ldots, m_{i-1}, \sigma_{i-1}\right). \\ \quad \sigma_i \leftarrow \mathsf{Sign}\left(sk, m_i\right). \\ \langle m, \sigma\rangle \leftarrow \mathcal{F}_S\left(pk, m_1, \sigma_1, \ldots, m_{q_S}, \sigma_{q_S}\right). \\ m \notin \{m_1, \ldots, m_{q_S}\}\ and \\ \quad \mathsf{Ver}\left(pk, m, \sigma\right) = accept. \end{array}\right] \geq \epsilon.$$

Similarly, we define the security notion for a randomized signature scheme $S' = \langle \mathsf{ParamGen}, \mathsf{KeyGen}, \mathsf{Sign}', \mathsf{Ver}\rangle$.

*Definition 2 (Forger of Randomized Signature Scheme): A forger $\mathcal{F}_{S'}$ is said to $(t, q_S, \epsilon)$-break the signature scheme $S' = \langle \mathsf{ParamGen}, \mathsf{KeyGen}, \mathsf{Sign}', \mathsf{Ver}\rangle$ using an adaptive chosen message attack if in at most $t$ processing time and after at most $q_S$ signature queries to the signing oracle, it outputs a valid forgery with a probability at least $\epsilon$ such that*

$$Pr\left[\begin{array}{l} \langle pk, sk\rangle \leftarrow \langle \mathsf{ParamGen}, \mathsf{KeyGen}\rangle\left(1^{\lambda}\right). \\ for\ i = 1, 2, \ldots, q_S \\ \quad m_i \leftarrow \mathcal{F}_{S'}\left(pk, m_1, \sigma_1, \ldots, m_{i-1}, \sigma_{i-1}\right). \\ \quad k_i \leftarrow \{0, 1\}^{\lambda'}, r_i \leftarrow T(k_i). \\ \quad \sigma_i = (r_i, s_i) \leftarrow \mathsf{rSign}\left(sk, m_i, k_i, r_i\right). \\ \langle m, \sigma\rangle \leftarrow \mathcal{F}_{S'}\left(pk, m_1, \sigma_1, \ldots, m_{q_S}, \sigma_{q_S}\right). \\ m \notin \{m_1, \ldots, m_{q_S}\}\ and \\ \quad \mathsf{Ver}\left(pk, m, \sigma\right) = accept. \end{array}\right] \geq \epsilon,$$

*where $\lambda'$ is the number of random bits required according to the security parameter $\lambda$.*

*Definition 3 (Complete Forger): Assume that the forger $\mathcal{F}_{S'}$ for a randomized signature scheme $S' = \langle \mathsf{ParamGen}, \mathsf{KeyGen}, \mathsf{Sign}', \mathsf{Ver}\rangle$ with security parameter $\lambda$ is given message $m$ and fixed salt $r$ that should be the first part of the signature. The forger is defined as a complete forger if with no signing query to the oracle, it outputs a valid forgery with probability 1, in at most poly$(\lambda)$ processing time, such that*

$$Pr\left[\begin{array}{l} \langle pk, sk\rangle \leftarrow \langle \mathsf{ParamGen}, \mathsf{KeyGen}\rangle\left(1^{\lambda}\right). \\ s \leftarrow \mathcal{F}_{S'}\left(pk, m, r\right). \\ \mathsf{Ver}\left(pk, m, \sigma = (r, s)\right) = accept. \end{array}\right] = 1.$$

*We denote the complete forger for the randomized signature scheme $S'$ as $\mathcal{F}^c_{S'}$.*

Note that a complete forger is a very powerful attacker that can produce the result of $\mathsf{rSign}(sk, m, k, r)$ without the knowledge about $sk$ and $k$. In some cases, this may imply that this attacker can do something that even a private key owner cannot. For example, in ECDSA, computing $s = k^{-1}(e + dr) \bmod n$ without $k$ is not easy, even if $d$ is given as well as $m$ (equivalently, $e$) and $r$. This may involve solving ECDLP to find $k$ such that the $x$-coordinate of $kP$ is either $r$ or $r + n$.

Next, we define a secure signature scheme. For this purpose, we use the security notion defined in [44].

*Definition 4 (Secure Signatures [44]): A signature scheme $S$ is existentially unforgeable under an adaptive chosen message attack if there is no forger who $(t, q_S, \epsilon)$-breaks $S$ with non-negligible $\epsilon$ with polynomial $t$ and $q_S$.*

In Section IV, we explained the proposed hybrid signature scheme that combines ECDSA and Falcon as classical and PQC signature schemes, respectively. We now show that no attacker can forge a hybrid signature $\sigma = (r, s_E, s_F)$ even if one of the signature schemes is completely broken, that is, there exists a complete forger for one of the schemes.

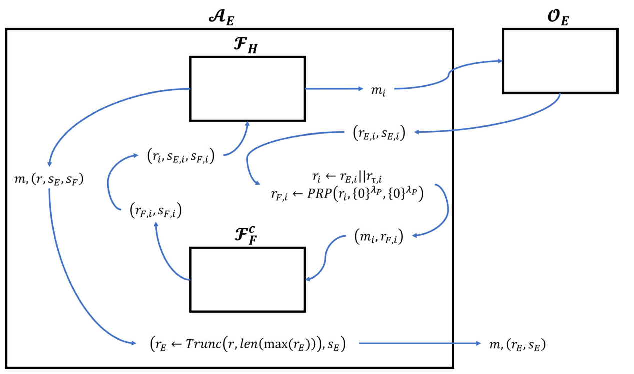

$\mathcal{A}_E$
$\mathcal{O}_E$
$\mathcal{F}_H$
$m_i$
$(r_{E,i}, s_{E,i})$
$(r_i, s_{E,i}, s_{F,i})$
$m, (r, s_E, s_F)$
$r_i \leftarrow r_{E,i}||r_{\tau,i}$
$r_{F,i} \leftarrow PRP(r_i, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$
$(r_{F,i}, s_{F,i})$
$\mathcal{F}_F^c$
$(m_i, r_{F,i})$
$(r_E \leftarrow Trunc(r, len(\max(r_E))), s_E)$
$m, (r_E, s_E)$

**FIGURE 4. Construction of an ECDSA forger $\mathcal{A}_E$ using a hybrid forger $\mathcal{F}_H$ and a complete Falcon forger $\mathcal{F}_F^c$.**

*Theorem 2: We assume that there exists a complete forger of Falcon, $\mathcal{F}_F^c$. If there exists a forger $\mathcal{F}_H$ using an adaptive chosen message attack who $(t, q_S, \epsilon)$-breaks the hybrid signature scheme, then there exists an attacker $\mathcal{A}_E$ who $(t + \alpha q_S + t(\mathcal{F}_F^c)q_S, q_S, \epsilon)$-breaks ECDSA, where $t(\mathcal{F}_F^c)$ is the execution time of $\mathcal{F}_F^c$ and $\alpha$ is a constant.*

*Proof:* From Definitions 1 and 2, an ECDSA signing oracle $\mathcal{O}_E$ returns a signature $\sigma_E = (r_E, s_E)$ on a message $m$ received as a signing query. According to Definition 3, $\mathcal{F}_F^c$, the complete forger of Falcon, can generate $s_F$, which is the second part of the Falcon signature, for a given message $m$ and a valid random value $r_F$. Subsequently, $\mathcal{A}_E$ can use $\mathcal{F}_H$ and $\mathcal{F}_F^c$ as subroutines to forge an ECDSA signature as shown in Fig. 4. The details of the forgery process are as follows:

1) $\mathcal{F}_H$ chooses a message $m_i$, then requests a hybrid signature on $m_i$ to $\mathcal{A}_E$. $\mathcal{A}_E$ will play the role of the signing oracle for the hybrid signature scheme.
2) $\mathcal{A}_E$ queries $m_i$ to the ECDSA signing oracle $\mathcal{O}_E$, then $\mathcal{O}_E$ returns the ECDSA signature $(r_{E,i}, s_{E,i})$.
3) $\mathcal{A}_E$ computes $r_i \leftarrow r_{E,i}||r_{\tau,i}$ where $r_{\tau,i}$ is a random bit string such that $len(r_{\tau,i}) = len(r_{F,i}) - len(\max(r_{E,i}))$, and computes $r_{F,i} \leftarrow PRP\left(r_i, \{0\}^{\lambda_P}, \{0\}^{\lambda_P}\right)$.
4) $\mathcal{A}_E$ sends $(m_i, r_{F,i})$ to $\mathcal{F}_F^c$.
5) $\mathcal{F}_F^c$ generates the second part of a Falcon signature, $s_{F,i}$, and returns the Falcon signature $\left(r_{F,i}, s_{F,i}\right)$.

6) $\mathcal{A}_E$ sends the hybrid signature $(r_i, s_{E,i}, s_{F,i})$ to $\mathcal{F}_H$.
7) $\mathcal{A}_E$ iterates 1)-6) at most $q_S$ times until $\mathcal{F}_H$ outputs a forged signature.
8) Finally, $\mathcal{F}_H$ outputs a valid message-signature pair $(m, \sigma_H) = (m, (r, s_E, s_F))$ where $m \notin \{m_1, m_2, \ldots, m_{q_S}\}$.
9) $\mathcal{A}_E$ outputs an ECDSA signature $(Trunc(r, len(\max(r_E))), s_E)$ on $m$.

It is obvious that the generated hybrid signature $(r_i, s_{E,i}, s_{F,i})$ in step 6) is valid, that is, it can be verified using Algorithm 6. In lines 1 through 2 of Algorithm 6, $r_{E,i}$ and $r_{F,i}$ are obtained by parsing $r_i$ and computing $PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$, respectively. Now, the individual verification for the Falcon signature $(r_{F,i}, s_{F,i})$ and ECDSA signature $(r_{E,i}, s_{E,i})$ will be successful. Therefore, $\mathcal{F}_H$ cannot distinguish $\mathcal{A}_E$ from an actual hybrid signature oracle. Clearly, the ECDSA signature returned in step 9) is valid.

The number of signature queries and success probability of $\mathcal{A}_E$ are the same as those of $\mathcal{F}_H$, that is, they are $q_S$ and $\epsilon$, respectively. Regarding the execution time, $\mathcal{A}_E$ consumes $(\alpha + t(\mathcal{F}_F^c))q_S$ time for the main loop in steps 1) through 6) for a constant $\alpha$, in addition to $t$, the time required for $\mathcal{F}_H$. In conclusion, $\mathcal{A}_E$, the ECDSA attacker, $\left(t + \left(\alpha + t(\mathcal{F}_F^c)\right) q_S, q_S, \epsilon\right)$-breaks ECDSA. □

The theorem implies that even if Falcon is completely broken, that is, there exists a complete forger for Falcon, the hybrid scheme is secure if only ECDSA is secure.

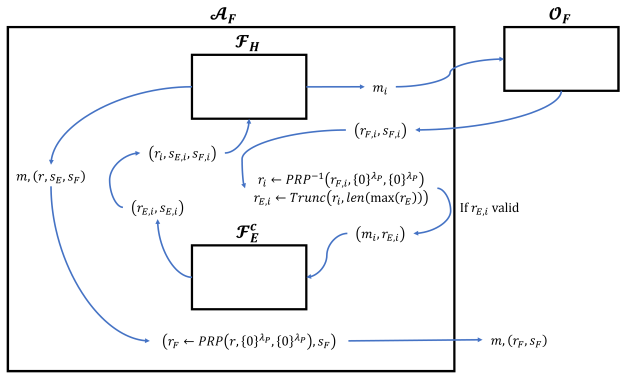

$\mathcal{A}_F$
$\mathcal{O}_F$
$\mathcal{F}_H$
$m_i$
$(r_{F,i}, s_{F,i})$
$(r_i, s_{E,i}, s_{F,i})$
$m, (r, s_E, s_F)$
$r_i \leftarrow PRP^{-1}(r_{F,i}, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$
$r_{E,i} \leftarrow Trunc(r_i, len(\max(r_E)))$
If $r_{E,i}$ valid
$(r_{E,i}, s_{E,i})$
$\mathcal{F}_E^c$
$(m_i, r_{E,i})$
$(r_F \leftarrow PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P}), s_F)$
$m, (r_F, s_F)$

**FIGURE 5. Construction of a Falcon forger $\mathcal{A}_F$ using a hybrid forger $\mathcal{F}_H$ and a complete ECDSA forger $\mathcal{F}_E^c$.**

*Theorem 3: Assume that there exists a complete forger of ECDSA, $\mathcal{F}_E^c$. If there exists a forger $\mathcal{F}_H$ using an adaptive chosen message attack who $(t, q_S, \epsilon)$-breaks the hybrid signature scheme, then there exists an attacker $\mathcal{A}_F$ who $\left(t + t(\mathcal{F}_E^c)q_S + \beta q_S \log_2 q_S, q_S \log_2 q_S, \epsilon\,(1 - 1/q_S)^{q_S}\right)$-breaks Falcon, where $t(\mathcal{F}_E^c)$ is the execution time of $\mathcal{F}_E^c$ and $\beta$ is a constant.*

*Proof:* From Definitions 1 and 2, a Falcon signing oracle $\mathcal{O}_F$ returns a signature $\sigma_F = (r_F, s_F)$ on a message $m$ received as a signing query. From Definition 3, $\mathcal{F}_E^c$, the complete forger of ECDSA, can generate $s_E$, the second part of the ECDSA signature, for a given $m$ and valid random value $r_E$. Subsequently, $\mathcal{A}_F$ can use $\mathcal{F}_H$ and $\mathcal{F}_E^c$ as subroutines to forge a Falcon signature as shown in Fig. 5. The details of the forgery process are as follows:

1) $\mathcal{F}_H$ chooses a message $m_i$, then requests a hybrid signature on $m_i$ to $\mathcal{A}_F$. $\mathcal{A}_F$ will play the role of the signing oracle for the hybrid signature scheme.
2) $\mathcal{A}_F$ queries $m_i$ to the Falcon signing oracle $\mathcal{O}_F$, then $\mathcal{O}_F$ returns the Falcon signature $(r_{F,i}, s_{F,i})$.
3) $\mathcal{A}_F$ computes $r_i \leftarrow PRP^{-1}(r_{F,i}, \{0\}^{\lambda_P}, \{0\}^{\lambda_P})$, then computes $r_{E,i} \leftarrow Trunc(r_i, len(\max(r_E)))$.
4) $\mathcal{A}_F$ checks whether $r_{E,i}$ is valid, i.e., there is a point $R_i \in E(\mathbb{F}_q)$ whose $x$-coordinate is either $r_{E,i}$ or $r_{E,i}+n$.
   a) If $r_{E,i}$ is valid, $\mathcal{A}_F$ sends $(m_i, r_{E,i})$ to $\mathcal{F}_E^c$.
   b) Otherwise, go back to step 2), and repeat 2)-4) at most $\log_2 q_S$ times until a valid $r_{E,i}$ is obtained. If no valid $r_{E,i}$ is obtained in $\log_2 q_S$ trials, $\mathcal{A}_F$ fails.
5) $\mathcal{F}_E^c$ generates the second part of an ECDSA signature, $s_{E,i}$, and returns the ECDSA signature $\left(r_{E,i}, s_{E,i}\right)$.
6) $\mathcal{A}_F$ sends the hybrid signature $(r_i, s_{E,i}, s_{F,i})$ to $\mathcal{F}_H$.
7) $\mathcal{A}_F$ iterates 1)-6) at most $q_S$ times until $\mathcal{F}_H$ outputs a forged signature.
8) Finally, $\mathcal{F}_H$ outputs a valid message-signature pair $(m, \sigma_H) = (m, (r, s_E, s_F))$ where $m \notin \{m_1, m_2, \ldots, m_{q_S}\}$.
9) $\mathcal{A}_F$ outputs a Falcon signature $\left(PRP(r, \{0\}^{\lambda_P}, \{0\}^{\lambda_P}), s_F\right)$ on $m$.

As in the proof of Theorem 2, the hybrid signature $(r_i, s_{E,i}, s_{F,i})$ generated in step 6) is valid, that is, it can be verified using Algorithm 6, and the individual verification of the Falcon signature $(r_{F,i}, s_{F,i})$ and ECDSA signature $(r_{E,i}, s_{E,i})$ will be successful. Therefore, $\mathcal{F}_H$ cannot distinguish $\mathcal{A}_F$ from an actual hybrid signature oracle. Clearly, the Falcon signature returned in step 9) is valid.

We now examine the success probability of $\mathcal{A}_F$. First, recall that the number of valid $r_{E,i}$'s is very close to the number of distinct $x$ coordinates of valid points on $E(\mathbb{F}_q)$, as the $x$ coordinate is almost always $r_{E,i}$ in step 4) [41]. (The case that the $x$ coordinate is $r_{E,i} + n$ is rare.) Number of distinct $x$ coordinates among valid points for a curve with group order $n$ is very close to $n/2$, because there always exist two distinct $y$-coordinates for a fixed $x$ except in a few extreme cases with $y = 0$. Therefore, the total number of valid $r_E$'s is very close to $n/2 \approx q/2 \approx 2^{len(\max(r_E))-1}$ for the standard NIST prime curves. Because $r_{E,i}$ computed in step 3) is a $len(\max(r_E))$-bit uniformly random string, the test of the validity of $r_{E,i}$ in step 4) succeeds with a probability of $2^{len(\max(r_E))-1}/2^{len(\max(r_E))} = 1/2$. Because $\mathcal{A}_F$ repeats steps 2) to 4) up to $\log_2 q_S$ times, $\mathcal{A}_F$ will get valid $r_{E,i}$ with the probability of $1 - \frac{1}{2^{\log_2 q_S}} = 1 - 1/q_S$ for each message $m_i \in \{m_1, \ldots, m_{q_S}\}$. Consequently, $q_S$ valid message-signature pairs are generated with the probability of $\epsilon\,(1 - 1/q_S)^{q_S}$, which is the success probability of $\mathcal{A}_F$.

It is easy to see that the execution time and the number of signature queries are $t + t(\mathcal{F}_E^c)q_S + \beta q_S \log_2 q_S$

and $q_S \log_2 q_S$, respectively, for a constant $\beta$. In conclusion, $\mathcal{A}_F$, the Falcon attacker, $(t + t(\mathcal{F}_E^c)q_S + \beta q_S \log_2 q_S, q_S \log_2 q_S, \epsilon\,(1 - 1/q_S)^{q_S})$-breaks Falcon. □

For $q_S \geq 2$, it holds that $1/4 \leq (1 - 1/q_S)^{q_S} < \lim_{q_S \to \infty} (1 - 1/q_S)^{q_S} = 1/e$. Therefore, the success probability of $\mathcal{A}_F$ is reduced from that of $\mathcal{F}_H$ by only a factor of at most 4. Although the number of queries of $A_F$ is larger than that of $\mathcal{F}_H$ by a factor of $\log_2 q_S$, the reduction is still valid because $q_S$ should be a polynomial according to Definition 4. This theorem implies that even if ECDSA is completely broken, that is, there exists a complete forger for ECDSA, the hybrid scheme is secure if only Falcon is secure.

## VI. PERFORMANCE EVALUATION

In this section, the performance of the proposed hybrid signature scheme is evaluated. First, we compare the signature size of the proposed hybrid signature scheme with that of the naive hybrid approach, which concatenates two individual signatures. Second, we examine the computational overhead of the proposed hybrid signature scheme involving operations such as PRP. For the experiments, we constructed the proposed hybrid signature scheme using ECDSA P-256 and Falcon-512 as a classical-PQC combination, as described in Section IV. In addition, we considered PQC-PQC combinations to address the concern that all classical signature schemes will be broken in the near future. Although NIST is standardizing CRYSTALS-Dilithium, Falcon, and SPHINCS+ as post-quantum digital signature schemes, NIST is also evaluating additional signature schemes based on various problems, such as multivariate and code-based signatures [45]. Therefore, we also considered the combinations involving additional post-quantum digital signature schemes. In this paper, we implemented the following three PQC-PQC combinations. (1) Unbalanced Oil and Vinegar (UOV) [46] and SNOVA [47], (2) UOV and Falcon, and (3) CRYSTALS-Dilithium and Falcon. UOV and SNOVA are multivariate signatures evaluated in the NIST additional digital signature standardization.

Experiments were conducted using two different systems. The first system was a desktop PC with an Intel(R) Core(TM) i7-7700 CPU (3.60GHz) and 8GB RAM. The other system was a Raspberry Pi 3 Model B Rev 1.2 with an ARM Cortex-A72 MP4 CPU (1.2GHz) and 1GB RAM. We implemented the hybrid signature scheme using Mbed-TLS 3.1.0 [48] for ECDSA P-256 and the reference implementations of Falcon-512 [49], UOV [50], SNOVA [51], and CRYSTALS-Dilithium [52] as building blocks. Because UOV and SNOVA support the parameter set satisfying NIST security level I, we used UOV-Is and SNOVA-(28, 17, 16, 2)-esk parameter sets corresponding to the security level. For CRYSTALS-Dilithium, we used the dilithium2 parameter set that satisfies NIST security level II because CRYSTALS-Dilithium did not support the parameter set for NIST security level I.

First, we evaluate the signature size and execution time for the ECDSA P-256 and Falcon-512 combination. The signature size of ECDSA P-256 is 64 bytes because $len(r_E) = len(s_E) = 256$ in a signature $\sigma_E = (r_E, s_E)$, whereas the signature size of Falcon-512 is 666 bytes for an uncompressed version, where 40 bytes are for $r_F$ and 626 bytes for $s_F$ in a signature $\sigma_F = (r_F, s_F)$. Therefore, the naive concatenation of ECDSA P-256 and Falcon-512 signatures consumes 730 bytes. By contrast, the size of the proposed hybrid signature is 698 bytes because the bit length of a hybrid signature is formulated as $\max(len(r_E), len(r_F)) + len(s_E) + len(s_F)$. In the experiments, we also applied a compressed signature setting for Falcon provided by the Falcon reference implementation. According to this setting, the size of the Falcon signature is reduced slightly by compressing a polynomial that derives $s_F$, and the signature size may vary depending on the private key, signed data, and random seed. This optimization reduces the signature sizes of both naive concatenation and proposed hybrid scheme.

**TABLE 1. Signature sizes of the ECDSA, Falcon, and hybrid signatures.**

| Signing algorithm | PC (in bytes) | Raspberry Pi (in bytes) |
|---|---|---|
| ECDSA | 64.00 | 64.00 |
| Falcon | 655.15 | 655.11 |
| Naive concatenation | 719.15 | 719.11 |
| **Proposed method** | **687.05** | **687.12** |

Table 1 presents the signature sizes of ECDSA, Falcon, naive concatenation, and proposed method in bytes. We measured the sizes of the signatures generated by each signing algorithm 1,000 times in the two systems, and the values in the table are averages. On the PC, because the signature sizes of ECDSA and Falcon were 64 and 655.15 bytes on average, the size of the naive concatenation was 719.15 bytes. The size of the proposed hybrid signature was 687.05 bytes, which is 4.46% (32 bytes) shorter than that of naive concatenation. The experimental results for the Raspberry Pi showed a similar trend. The signature sizes of ECDSA, Falcon, and naive concatenation are 64, 655.11, and 719.11 bytes on average, respectively. The proposed signature consumes 687.12 bytes, which is 4.45% (32 bytes) less than that of naive concatenation.

Table 2 lists the execution times required to generate and verify the signatures of ECDSA, Falcon, and the hybrid signature scheme in milliseconds. The values in the table are the averages of 1,000 measurements. On the PC, ECDSA, Falcon, and hybrid signatures were generated in 0.991 ms, 0.278 ms, and 1.278 ms, respectively. The signature generation time for naive concatenation is the sum of those for ECDSA and Falcon. Thus, the additional time required to generate a hybrid signature was $1.278 - 1.269 = 0.009$ ms, which was 0.71% of the signature generation time of naive concatenation. On Raspberry Pi, signatures were generated in 10.334, 2.022, and 12.372 ms on average. Thus, the additional time required to generate a hybrid signature

**TABLE 2. Signature generation and verification times of the ECDSA, Falcon, and hybrid signatures.**

| Algorithm | PC (in milliseconds) | Raspberry Pi (in milliseconds) |
|---|---|---|
| ECDSA signature generation | 0.991 | 10.334 |
| ECDSA signature verification | 1.974 | 20.602 |
| Falcon signature generation | 0.278 | 2.022 |
| Falcon signature verification | 0.028 | 0.217 |
| Naive concatenation sig. gen. | 1.269 | 12.356 |
| Naive concatenation sig. ver. | 2.002 | 20.819 |
| Hybrid signature generation | 1.278 | 12.372 |
| Hybrid signature verification | 2.009 | 20.832 |

was $12.372 - 12.356 = 0.016$ ms, which was 0.13% of the signature generation time of naive concatenation.

Next, we examine signature verification time. On the PC, ECDSA, Falcon, and hybrid signatures were verified in 1.974 ms, 0.028 ms, and 2.009 ms, respectively. The additional time required to verify a hybrid signature was $2.009 - 2.002 = 0.007$ ms, which was 0.35% of the signature verification time of naive concatenation. On Raspberry Pi, the signature verification times were 20.602, 0.217, and 20.832 ms, respectively. and the additional time required for hybrid signature verification was 0.06% of the sum of individual verification times. Therefore, the experimental results demonstrate that the time overhead for the proposed hybrid signature generation and verification is almost negligible.

**TABLE 3. Signature sizes of the UOV, SNOVA, and hybrid signatures.**

| Signing algorithm | PC (in bytes) | Raspberry Pi (in bytes) |
|---|---|---|
| UOV | 96.00 | 96.00 |
| SNOVA | 106.00 | 106.00 |
| Naive concatenation | 202.00 | 202.00 |
| **Proposed method** | **186.00** | **186.00** |

We also evaluate the performance of the three PQC-PQC combinations. The computational overhead was almost negligible for all combinations. Therefore, we only provide the detailed data on signature size reduction. The first combination was a hybrid signature of UOV-Is and SNOVA-(28, 17, 16, 2)-esk. Table 3 presents the signature sizes of UOV, SNOVA, naive concatenation, and the proposed method in bytes. As shown in the table, the signature sizes of UOV and SNOVA are 96 and 106 bytes, respectively, and the proposed method reduces the signature size by 16 bytes (7.92%).

The second combination was a hybrid signature using UOV-Is and Falcon-512. Table 4 presents the signature sizes of UOV, Falcon, naive concatenation, and the proposed method in bytes. The results show that the proposed hybrid signature scheme reduced the signature size of naive concatenation by 2.13% on both systems.

**TABLE 4. Signature sizes of the UOV, Falcon, and hybrid signatures.**

| Signing algorithm | PC (in bytes) | Raspberry Pi (in bytes) |
|---|---|---|
| UOV | 96.00 | 96.00 |
| Falcon | 655.16 | 654.97 |
| Naive concatenation | 751.16 | 750.97 |
| **Proposed method** | **735.21** | **735.04** |

**TABLE 5. Signature sizes of the Dilithium, Falcon, and hybrid signatures.**

| Signing algorithm | PC (in bytes) | Raspberry Pi (in bytes) |
|---|---|---|
| CRYSTALS-Dilithium | 2452.00 | 2452.00 |
| Falcon | 654.94 | 654.96 |
| Naive concatenation | 3106.94 | 3106.96 |
| **Proposed method** | **3075.10** | **3075.11** |

The final combination was a hybrid signature of CRYSTALS-Dilithium and Falcon.[2] Table 5 lists the signature sizes of CRYSTALS-Dilithium, Falcon, naive concatenation, and the proposed method in bytes. The signature size of CRYSTALS-Dilithium was 2452.00 bytes on both systems, while that of Falcon was 654.94 bytes on the PC and 654.96 bytes on Raspberry Pi. Thus, the proposed method reduces the signature size of naive concatenation by 1.03%.

## VII. DISCUSSIONS

In this paper, we proposed a new hybrid signature construction method that combines two individual signature schemes. We instantiated the proposed method with the combination of ECDSA P-256 and Falcon-512, which are representative classical and post-quantum signature schemes, respectively. However, the proposed method is not limited to this specific combination. For example, we can construct a hybrid signature scheme with two classical signature schemes, such as DSA-ECDSA, or two post-quantum signature schemes, such as UOV-SNOVA, UOV-Falcon, and Dilithium-Falcon, as shown in Section VI. Additionally, it may be possible to construct a hybrid scheme using more than two signature schemes. Because the time invested in the security analysis of new PQC schemes is relatively short compared to that of classical signature schemes such as RSA and ECDSA, there are still concerns regarding the security of new PQC candidates, as we have already seen from the example of SIKE [14]. To mitigate concerns about potential threats to post-quantum signature schemes, we can design a hybrid signature scheme using two PQC schemes and a classical scheme. This combination can also reduce the signature length because the salt in the hybrid signature is derived from the three salts in the underlying signatures. For example,

[2]We remark that the reduction proof is possible in only one direction for this combination. We can prove that even if there is a complete forger for Falcon, the hybrid scheme is secure if only CRYSTALS-Dilithium is secure. However, the other direction is not provable.

assume that there are three signatures: $\sigma_A = (r_A, s_A)$, $\sigma_B = (r_B, s_B)$, and $\sigma_C = (r_C, s_C)$, where the signatures are one classical signature and two post-quantum signatures. The proposed method can combine the three signatures into a hybrid signature $\sigma = (r, s_A, s_B, s_C)$. Therefore, in future work, we will pursue research to provide hybrid signature construction by combining more than two signature schemes.

We note that the application of the proposed hybrid signature construction method is limited to randomized signature schemes such as DSA, ECDSA, picnic, and Falcon. Therefore, it will be an interesting topic to expand hybrid signature construction to non-randomized signature schemes. Although Bindel et al. [15] already proposed various hybrid signature combinations including non-randomized signatures, their schemes do not allow for backward compatibility, because they considered a different design goal, i.e., non-separability. Furthermore, some of their constructions were not compact, i.e., they did not reduce the signature size compared to naive concatenation. Therefore, it will be a promising research topic to design a hybrid signature construction method involving non-randomized signatures that provides backward compatibility and compactness.

## VIII. CONCLUSION

In this study, we proposed a new hybrid signature construction method combining the classical and quantum-resistant signature schemes. The proposed method generates a signature shorter than the naive concatenation of two signatures with negligible computational overhead and guarantees the maximum security level among the two component signature schemes. We demonstrated the feasibility of the method with a practical instance combining ECDSA P-256 and Falcon-512 and formally proved that the hybrid ECDSA-Falcon scheme is secure even if one of the signature schemes is completely broken. According to the experimental results for a desktop PC and Raspberry Pi 3 Model B, the proposed scheme generated a signature that is 4.46% shorter than that of naive concatenation. The computational overhead of the proposed method was far less than 1% of the overall time for signature generation and verification and was as low as 0.06% for signature verification using Raspberry Pi. We also verified the feasibility of the proposed method for PQC-PQC combinations. For example, according to the experimental results, the proposed method reduces the signature size for a UOV-SNOVA hybrid signature by 7.92%. Therefore, the proposed hybrid signature scheme is expected to provide an attractive solution suitable for network resource-constrained environments, by generating a shortened hybrid signature with a negligible computational overhead.

## ACKNOWLEDGMENT

The authors would like to thank Turabek Gaybullaev for his assistance in implementing the proposed hybrid signature.

## REFERENCES

[1] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Comput. STOC*, 1996, pp. 212–219.

[2] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. Symp. Found. Comput. Sci.*, 1994, pp. 124–134.

[3] *IEEE Standard Specifications for Public-Key Cryptography*, IEEE Standard 1363-2000, 2020.

[4] C. Q. Choi, "IBM's quantum leap: The company will take quantum tech past the 1,000-qubit mark in 2023," *IEEE Spectr.*, vol. 60, no. 1, pp. 46–47, Jan. 2023.

[5] Google Quantum AI, "Suppressing quantum errors by scaling a surface code logical qubit," *Nature*, vol. 614, no. 7949, pp. 676–681, 7949.

[6] X.-C. Wu, P. Khalate, A. Schmitz, S. Premaratne, K. Rasch, S. Daraeizadeh, R. Kotlyar, S. Ren, J. Paykin, and F. Rose, "Intel quantum SDK version 1.0: Extended C++ compiler, runtime and quantum hardware simulators for hybrid quantum-classical applications," *Bulletin Amer. Phys. Soc.*, vol. 68, no. 3, Jan. 2023, Art. no. RR08.005.

[7] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehle, "CRYSTALS-kyber: A CCA-secure module-lattice-based KEM," in *Proc. IEEE Eur. Symp. Secur. Privacy*, Apr. 2018, pp. 353–367.

[8] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-dilithium: A lattice-based digital signature scheme," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2018, no. 1, pp. 238–268, Feb. 2018.

[9] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "Falcon: Fast-Fourier lattice-based compact signatures over NTRU," NIST's Post-Quantum Cryptogr. Standardization Process Round 2. Accessed: Jan. 18, 2024. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-2-submissions

[10] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe, "The SPHINCS+ signature framework," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 2129–2146.

[11] P. Kaye and R. Laflamme, *An Introduction to Quantum Computing*. Oxford, U.K.: Oxford Univ. Press, 2007.

[12] D. Joseph, R. Misoczki, M. Manzano, J. Tricot, F. D. Pinuaga, O. Lacombe, S. Leichenauer, J. Hidary, P. Venables, and R. Hansen, "Transitioning organizations to post-quantum cryptography," *Nature*, vol. 605, no. 7909, pp. 237–243, May 2022.

[13] R. Azarderakhsh, M. Campagna, C. Costello, L. De Feo, B. Hess, A. Jalali, D. Jao, B. Koziel, B. LaMacchia, and P. Longa, "Supersingular isogeny key encapsulation," NIST's Post-Quantum Cryptogr. Standardization Process Round 2. Accessed: Jan. 18, 2024. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-2-submissions

[14] W. Castryck and T. Decru, "An efficient key recovery attack on SIDH," in *Proc. 42nd Annu. Int. Conf. Theory Appl. Cryptograph. Techn. (EUROCRYPT)*. Berlin, Germany: Springer-Verlag, 2023, pp. 423–447.

[15] N. Bindel and B. Hale, "A note on hybrid signature schemes," *Cryptol. ePrint Arch.*, pp. 1–22, Jul. 2023.

[16] N. Bindel, U. Herath, M. McKague, and D. Stebila, "Transitioning to a quantum-resistant public key infrastructure," in *Post-Quantum Cryptography*. Cham, Switzerland: Springer, 2017, pp. 384–405.

[17] J. Bos, C. Costello, L. Ducas, I. Mironov, M. Naehrig, V. Nikolaenko, A. Raghunathan, and D. Stebila, "Frodo: Take off the ring! Practical, quantum-secure key exchange from LWE," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 1006–1018.

[18] R. Azarderakhsh, R. El Khatib, B. Koziel, and B. Langenberg, "Hardware deployment of hybrid PQC," *Cryptol. ePrint Arch.*, pp. 1–17, May 2021.

[19] A. V. Komarova, A. A. Menshchikov, and A. G. Korobeynikov, "New hybrid signature schemes with increasing level of resistance," *J. Phys., Conf. Ser.*, vol. 2094, no. 3, Nov. 2021, Art. no. 032039.

[20] M. Bellare and P. Rogaway, "The exact security of digital signatures-how to sign with RSA and rabin," in *Proc. Eurocrypt*, vol. 96. Berlin, Germany: Springer, 1996, pp. 399–416.

[21] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 4, pp. 469–472, Jul. 1985.

[22] C. F. Kerry and P. D. Gallagher, "Digital signature standard (DSS)," in *Proc. FIPS PUB*, 2013, pp. 4–186.

[23] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Secur.*, vol. 1, no. 1, pp. 36–63, Aug. 2001.

[24] A. Antipa, D. Brown, R. Gallant, R. Lambert, R. Struik, and S. Vanstone, "Accelerated verification of ECDSA signatures," in *Proc. Sel. Areas Cryptogr., 12th Int. Workshop (SAC)*, Kingston, ON, Canada. Cham, Switzerland: Springer, Aug. 2006, pp. 307–318.

[25] M.-S. Chen, A. Hülsing, J. Rijneveld, S. Samardjiska, and P. Schwabe, "MQDSS specifications," NIST's Post-Quantum Cryptogr. Standardization Process Round 2. Accessed: Jan. 18, 2024. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-2-submissions

[26] E. Alkim, P. S. Barreto, N. Bindel, J. Krämer, P. Longa, and J. E. Ricardini, "The lattice-based digital signature scheme qTESLA," in *Proc. Appl. Cryptogr. Netw. Secur., 18th Int. Conf.*, Rome, Italy. Cham, Switzerland: Springer, 2020, pp. 441–460.

[27] M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, and G. Zaverucha. (2020). *The Picnic Signature Scheme Design Document (version 2.2)*. [Online]. Available: https://microsoft.github.io/Picnic

[28] V. S. Miller, "Use of elliptic curves in cryptography," in *Proc. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 1985, pp. 417–426.

[29] N. Koblitz, "Elliptic curve cryptosystems," *Math. Comput.*, vol. 48, no. 177, pp. 203–209, 1987.

[30] G. Alagic, D. Apon, D. Cooper, Q. Dang, T. Dang, J. Kelsey, J. Lichtinger, C. Miller, D. Moody, and R. Peralta, "Status report on the third round of the NIST post-quantum cryptography standardization process," U.S. Dept. Commerce, NIST, Gaithersburg, MD, USA, Tech. Rep. NIST Interagency/Internal Report (NISTIR)—8413, 2022.

[31] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. 14th Annu. ACM Symp. Theory Comput.*, May 2008, pp. 197–206.

[32] L. Ducas, V. Lyubashevsky, and T. Prest, "Efficient identity-based encryption over NTRU lattices," in *Proc. Adv. Cryptol.–ASIACRYPT 20th Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Kaoshiung, Taiwan. Berlin, Germany: Springer, 2014, pp. 22–41.

[33] L. Ducas and T. Prest, "Fast Fourier orthogonalization," in *Proc. ACM Int. Symp. Symbolic Algebr. Comput.*, Jul. 2016, pp. 191–198.

[34] M. Ajtai, "Generating hard instances of lattice problems," in *Proc. ACM Symp. Theory Comput.*, 1996, pp. 99–108.

[35] D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry, "Random oracles in a quantum world," in *Proc. Adv. Cryptol.–ASIACRYPT 17th Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Seoul, South Korea. Berlin, Germany: Springer, Dec. 2011, pp. 41–69.

[36] C. Peikert, "A decade of lattice cryptography," *Found. Trends® Theor. Comput. Sci.*, vol. 10, no. 4, pp. 283–424, 2016.

[37] E. Barker, "NIST special publication 800-57 part 1 revision 5, recommendation for key management: Part 1—General," Special Publication (SP) 800-57 Part 1 Revision 5, NIST, Gaithersburg, MD, USA. Accessed: Jan. 18, 2024. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf

[38] National Institute of Standards and Technology, "FIPS 197: Advanced encryption standard (AES)," Federal Inf. Process. Standards Publication (FIPS) NIST FIPS 197-upd1, Gaithersburg, MD, USA, May 2023. Accessed: Jan. 18, 2024. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf

[39] M. Dworkin, "NIST special publication 800–38G, recommendation for block cipher modes of operation: Methods for format-preserving encryption," Special Publication (SP) 800-38G, NIST, Gaithersburg, MD, USA. Accessed: Jan. 18, 2024. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38G.pdf

[40] M. Dworkin, "Draft NIST special publication 800–38G revision 1, recommendation for block cipher modes of operation: Methods for format-preserving encryption," Special Publication (SP) 800-38G Revision 1, NIST, Gaithersburg, MD, USA. Accessed: Jan. 18, 2024. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38Gr1-draft.pdf

[41] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide To Elliptic Curve Cryptography*. Berlin, Germany: Springer, 2003.

[42] D. Hankerson and A. Menezes, "Elliptic curve cryptography," in *Encyclopedia of Cryptography, Security and Privacy*. Boston, MA, USA: Springer, 2021, pp. 1–2.

[43] F. Zhang, R. Safavi-Naini, and W. Susilo, "An efficient signature scheme from bilinear pairings and its applications," in *Proc. Public Key Cryptogr. PKC 7th Int. Workshop Theory Pract. Public Key Cryptogr.*, Singapore. Berlin, Germany: Springer, 2004, pp. 277–290.

[44] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM J. Comput.*, vol. 17, no. 2, pp. 281–308, Apr. 1988.

[45] NIST. *Post-quantum Cryptography: Digital Signature Schemes (Round 1 Additional Signatures)*. Accessed: Jan. 25, 2024. [Online]. Available: https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures

[46] A. Kipnis, J. Patarin, and L. Goubin, "Unbalanced oil and vinegar signature schemes," in *Proc. 99th Int. Conf. Theory Appl. Cryptograph. Techn. (EUROCRYPT)*. Berlin, Germany: Springer, 1999, pp. 206–222.

[47] L.-C. Wang, P.-E. Tseng, Y.-L. Kuan, and C.-Y. Chou, "A simple noncommutative UOV scheme," *Cryptol. ePrint Arch.*, pp. 1–30, Dec. 2022.

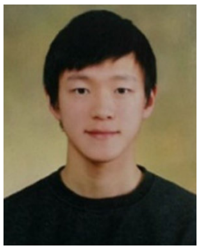
[48] *Mbed TLS V3.1.0*. Accessed: Aug. 29, 2022. [Online]. Available: https://github.com/ARMmbed/mbedtls/archive/refs/tags/v3.1.0.tar.gz

[49] *FALCON's Reference Implementation, Version: 2020-09-30*. Accessed: Aug. 29, 2022. [Online]. Available: https://falcon-sign.info/Falcon-impl-20211101.zip

[50] *UOV's Reference Implementation*. Accessed: Jan. 18, 2024. [Online]. Available: https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/UOV-submission.zip

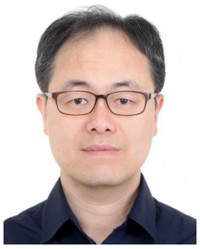
[51] *SNOVA's Reference Implementation*. Accessed: Jan. 18, 2024. [Online]. Available: https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/submission-pkg/SNOVAnoKATs-submission.zip

[52] *Dilithium's Reference Implementation*. Accessed: Jan. 18, 2024. [Online]. Available: https://github.com/pq-crystals/dilithium



**HEE-YONG KWON** received the B.S. and M.S. degrees in computer engineering from Inha University, South Korea, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree in computer engineering. His research interests include communication system security and information security.

**INDRA BAJUNA** is currently pursuing the B.S. degree in computer engineering with Inha University, South Korea. Her research interests include post-quantum cryptographic algorithms and blockchain security.



**MUN-KYU LEE** (Member, IEEE) received the B.S. and M.S. degrees in computer engineering and the Ph.D. degree in electrical engineering and computer science from Seoul National University, in 1996, 1998, and 2003, respectively. From 2003 to 2005, he was a Senior Engineer with the Electronics and Telecommunications Research Institute, South Korea. He is currently a Professor with the Department of Computer Engineering, Inha University, South Korea. His research interests include cryptographic algorithms, information security, and theory of computation.