The Security of Hash-and-Sign with Retry against Superposition Attacks

Haruhisa Kosuge¹ and Keita Xagawa²[0000-0002-6832-9940]

¹ NTT Social Informatics Laboratories, Japan hrhs.kosuge@ntt.com
² Technology Innovation Institute, UAE keita.xagawa@tii.ae

Abstract. Considering security against quantum adversaries, while it is important to consider the traditional existential unforgeability (EUF-CMA security), it is desirable to consider security against adversaries making quantum queries to the signing oracle: Plus-one security (PO security) and blind unforgeability (BU security) proposed by Boneh and Zhandry (Crypto 2013) and Alagic et al. (EUROCRYPT 2020), respectively. Hash-and-sign is one of the most common paradigms for constructing EUF-CMA-secure signature schemes in the quantum random oracle model, employing a trapdoor function and a hash function. It is known that its derandomized version is PO- and BU-secure. A variant of hash-and-sign, known as hash-and-sign with retry (HSwR), formulated by Kosuge and Xagawa (PKC 2024), is widespread since it allows for weakening the security assumptions of a trapdoor function. Unfortunately, it has not been known whether HSwR can achieve PO- and BU-secure even with derandomization.

In this paper, we apply a derandomization with bounded loops to HSwR. We demonstrate that HSwR can achieve PO and BU security through this approach. Since derandomization with bounded loops offers advantages in some implementations, our results support its wider adoption, including in NIST PQC candidates.

Keywords: Post-quantum cryptography · Quantum random oracle model · Superposition attack · Digital signature · Hash-and-sign.

1 Introduction

Security Models of Digital Signatures against Quantum Adversaries: Digital signatures are crucial for ensuring the integrity and authenticity of digital communications. The standard and traditional security notion for digital signatures is existential unforgeability against a chosen-message attack (EUF-CMA) [13]. Roughly speaking, we say a signature scheme is EUF-CMA-secure if no efficient adversary can forge a signature even if the adversary has access to a signing oracle, thereby capturing both non-repudiation and authentication.

The advent of quantum computers has raised concerns about the security of digital signatures due to Shor's algorithm [24]. Consequently, there has been

growing interest in post-quantum cryptography (PQC). In 2022, NIST selected three candidates of digital signatures, namely Falcon, Dilithium (ML-DSA), and SPHINCS+ (SLH-DSA), for standardization [22]. Furthermore, NIST initiated an additional call for digital signatures [21].

Given that post-quantum signatures must withstand attacks from quantum computers, their security proofs must be conducted in the quantum random oracle model (QROM) [5], rather than in the random oracle model (ROM), since the QROM models quantum adversaries having offline access to the hash function. The EUF-CMA security in the QROM allows an adversary to make quantum queries to the random oracle and *classical* queries to the signing oracle.

If quantum computing becomes ubiquitous, EUF-CMA might not adequately capture the necessary security requirements for signatures, as end-users may use personal quantum computers for signing. In that case, the adversary may carry out superposition attacks, which force the generation of quantum superpositions of signatures. Even in such a future, the security model of Boneh and Zhandry [6] remains valid, as it assumes *quantum* queries to the signing oracle. This model is called plus-one unforgeability (PO security, in short) [1] since the adversary needs to generate q+1 pairs of message/signature with q quantum queries to the signing oracle. Another model, proposed by Alagic et al. [1], is called blind unforgeability (BU security, in short). In this model, certain messages are *blinded*, meaning that the signing oracle is designed not to return signatures for these messages, and the adversary must forge signatures corresponding to the blinded messages. Note that there are strong (PO, sBU, and sEUF-CMA) and weak (wPO, BU, and EUF-CMA) security notions ³. Since a weak variant of PO is not defined, we introduce a new definition: weak PO (wPO).

Regarding the relationship between security models, a MAC scheme that is PO-secure but BU-insecure has been demonstrated [1, ePrint], as well as a MAC/signature scheme that is BU-secure but PO-insecure [26, Appendix C, ePrint]. We illustrate the relationship between these security notions in Fig. 1. As shown in the diagram, the relationship between these security models remains partially understood, necessitating independent evaluation of both PO and BU security.

Hash-and-Sign with Retry: Two paradigms are typically employed to construct EUF-CMA-secure signatures: Hash-and-sign (also known as full domain hash (FDH)) [3, 4] and Fiat-Shamir [10]. This paper focuses on hash-and-sign. Hash-and-sign constructs digital signatures from a trapdoor function (TDF) and a hash function. For provable security, hash-and-sign requires a special TDF called preimage-sampleable function (PSF) [12]. Since PSFs are generally hard to build, hash-and-sign with retry (HSwR) [23, 19], which can construct signatures using non-PSF TDFs, has been widely adopted among multivariate and code-based

³ The distinction between *strong* and *weak* security is determined by whether an adversary's forgery is considered successful if it targets messages for which the attacker has already obtained information about the corresponding signatures through signing queries (strong) or targets completely new messages (weak).



Fig. 1: Relationship between different security notions

signatures, including new NIST PQC candidates, QR-UOV [11] and PROV [14]. In the signature generation, the paradigm repeatedly performs loop iterations until certain conditions are met. Kosuge and Xagawa [19] have provided a reduction from the non-invertibility and preimage-simulatability of the underlying TDF to the EUF-CMA security in the QROM ⁴.

Current Status of PO and BU Security Proofs: Boneh and Zhandry [6] and Chatterjee, Chung, Liang, and Malavolta [7] showed that the derandomized version of hash-and-sign (without retry) is PO- and BU-secure, respectively. Also, Xagawa [26] demonstrated PO and BU security proofs of the derandomized version of Fiat-Shamir with aborts [20] that has a similar structure as HSwR. However, whether HSwR achieves PO/BU security even with derandomization remains unclear. Thus, it is natural to pose the following question:

Can HSwR achieve PO and BU security (with derandomization)?

1.1 Contribution

We affirmatively answer the question. Applying the derandomization and bounding the number of retries to HSwR, we show that the variant is PO- and BU- secure. ⁵ We refer to this version of HSwR as *derandomized hash-and-sign* with bounded retry (DHSwBR). Additionally, we demonstrate that DHSwBR is EUF-CMA-secure under the same assumption as the existing proof for the original HSwR [19], along with the pseudorandomness of the PRF used for the derandomization.

Note that we evaluate both strong and weak security notions for PO, BU, and EUF-CMA. Additionally, our proofs are reductions from the EUF-NMA security of the original HSwR, where NMA stands for No Message Attack. Kosuge and Xagawa [19] have shown a reduction from non-invertibility of TDF

⁴ In general, *non-invertibility* of TDFs is called *one-wayness*. We make a distinction between them depending on the way to choose challenges (non-invertibility follows [16] and one-wayness follows [3]).

⁵ Derandomized HSwR (with unbounded retry) is available as an option in PROV [14], and its security has been evaluated in the ROM by Cogliati et al. [8]. (Unfortunately, their proof contains an error).

Table 1: Summary of the existing and our security proofs in the QROM. In "Paradigm", HS, DHS, PHS, DPHS denote original, derandomized, probabilistic, and derandomized probabilistic hash-and-sign, respectively, and +RF denotes that it replaces PRFs with random functions (see the definitions in Appendix B). In "Assumptions", PSF indicates that the TDF is PSF, while INJ, CR, SPR, and PS represent, in decreasing order of strength, the injection, collision resistance, second-preimage resistance, and preimage-simulatability of the TDF, respectively. Here, "(C-)" denotes that a computational bound may be used for the preimage-simulatability. (q)PRF denotes (quantum) pseudo-randomness of PRFs.

Proof	Paradigm	Security	Assumptions
[5]	HS + RF	seuf-cma	CR
[19]	PHS	EUF-CMA	EUF-NMA, (C-)PS
[19]	HSwR	EUF-CMA	EUF-NMA, (C-)PS
[19]	HSwR	SEUF-CMA	EUF-NMA, INJ, (C-)PS
[6]	DHS	PO	PSF, CR, qPRF
[26]	DHS	PO	PSF, CR, qPRF
[26]	DHS + RF	PO	PSF, CR
[7]	DHS	BU	PSF, CR, qPRF
[26]	DHS	$_{\mathrm{sBU}}$	PSF, CR, qPRF
Section 4.1	DHSwBR/DPHS	PO	EUF-NMA, CR, PS, qPRF
Section 4.1	DHSwBR/DPHS	wPO	EUF-NMA, PS, q PRF
Section 4.2	DHSwBR/DPHS	$_{\rm sBU}$	EUF-NMA, CR, PS, q PRF
Section 4.2	DHSwBR/DPHS	BU	EUF-NMA, PS, q PRF
Section 4.3	DHSwBR/DPHS	SEUF-CMA	EUF-NMA, SPR, (C-)PS, PRF
Section 4.3	DHSwBR/DPHS	EUF-CMA	EUF-NMA, (C-)PS, PRF

to EUF-NMA. By demonstrating reductions from the EUF-NMA security, we not only establish reductions from non-invertibility but also enable adaptation to new security properties that have yet to be discovered. We summarize the results and their comparison to the existing proofs in Table 1. By setting the number of retries as one, our proof can be applied to the security proof of the derandomized probabilistic hash-and-sign.

Implications of Our Results: Since DHSwBR is interoperable with the original HSwR, it can be considered an option for signature schemes adopting HSwR. In addition to the security advantage of provable security against superposition attacks, the option also offers advantages in certain implementations. When signature generation depends on the entropy of randomness, security is inherently tied to the quality of the implementation. By the derandomization, security can be maintained without relying on the random number generation. This is particularly beneficial for platforms where sufficient entropy in random number generation advantage.

tages, derandomization with bounded loops should be recognized as a major option for HSwR signatures.

1.2 Technical Overview

Before presenting the technical overview of our proof for (W)PO and (S)BU security, we briefly explain HSwR, its variant DHSwBR, and preimage-simulatability. HSwR uses a TDF that consists of (Gen, F, Inv). Gen generates a public/secret key pair (vk, sk) that is also a key pair of the signature scheme. Taking vk and $x \in \mathcal{X}$ as inputs, a hard-to-invert function F deterministically outputs $y \in \mathcal{Y}$. The function Inv is a probabilistic function that, given sk and y as input, returns an x such that F(vk, x) = y with high probability, or outputs \perp (indicating inversion failure). For a message m and a uniformly chosen salt r, the signing algorithm computes Inv(sk, H(r, m)), where H is a random function. If Inv(sk, H(r, m)) fails in inversion, a new r is chosen, and this process is repeated until the inversion succeeds. Then, (r, x) is output as the signature. A signature (r, x) is verified if F(vk, x) = H(r, m) holds. As for DHSwBR, in addition to sk, the signing key includes keys s and s' for PRFs PRF and PRF'. We use a counter k, which increments by 1 with each loop iteration, to derive a salt as $r := \mathsf{PRF}(s, m, k)$ and a random coin $r' := \mathsf{PRF}'(s', m, k)$ for Inv. Also, the number of retries is bounded by a parameter B. Aside from the above derandomization in the signature generation, DHSwBR is identical to HSwR. An important property of HSwR/DHSwBR is the preimage-simulatability, which assumes that the following two are statistically or computationally indistinguishable [19]:

- x obtained after retrying y until y becomes invertible by lnv(sk, y).
- -x obtained by a simulator that does not use the secret key sk.

Let us now proceed with the technical overview of our proof. In the reduction, the EUF-NMA adversary, which does not possess the signing key, must simulate the signing oracle. To achieve this, the following two steps are required:

- First, it must modify the output of the random function H to make simulated signatures generated in the signing oracle valid.
- Second, the message/signature pair output by the (w)PO or (s)BU adversary must be verified using the original random function. This condition is essential for the EUF-NMA adversary to win its own game.

For the second step, this can be achieved by utilizing the techniques used by Xagawa [26] in proving the PO and sBU security of Fiat-Shamir with aborts taking derandomization with bounded loops. Let (m^*, r^*, x^*) be one of the message/signature pairs output by a (w)PO or (s)BU adversary, and let (r_m, x_m) be a signature generated by the signing oracle taking m. The random function H is modified such that $H(r_m, m) = F(vk, x_m)$ holds for any m to accept signatures generated by the signing oracle. We can modify the game so that the adversary can win if and only if $r^* \neq r_{m^*}$ holds ⁶. Since the values of H(r, m) for $r \neq r_m$

⁶ In the PO/sBU security proofs, collision-resistance of the TDF is required.

remain unchanged from the original, the signature (m^*, r^*, x^*) can be verified using the original random function if $r^* \neq r_{m^*}$.

However, modifying outputs of H to accept simulated signatures in the first step cannot be achieved using the techniques from [9, 26]; they rely on the strong assumption of statistical or divergence honest-verifier zero-knowledge (HVZK), which requires simulation of *succeeding and failing attempts*, while preimagesimulatability only requires that of succeeding attempt. We explain the difficulty in simulating the signing oracle. In the real experiment, for a message m, when inversion first succeeds at the k-th iteration, $H(r_i, m) = y_i$ holds for each $\{(r_i, y_i)\}_{i=1,...,k}$, where r_i and y_i are generated sequentially from i = 1. To simulate this signing oracle, the EUF-NMA adversary must simulate $\{y_i\}_{i=1,...,k}$ without using sk; however, preimage-simulatability only assures that the last y_k is simulated by F(vk, x) for some x.

To address this problem, we employ the one-way-to-hiding (O2H) lemma [25, 2]. Assuming that the guessing probability of $\{r_i\}_{i=1,...,k-1}$ is negligible, we can eliminate the need for simulation of $\{y_i\}_{i=1,...,k-1}$, allowing the EUF-NMA adversary to simulate signatures under the assumption of preimage-simulatability. Note that we can only use statistical preimage-simulatability since we cannot perform adaptive reprogramming [15] in the quantum signing oracle setting, while it is a common technique for establishing computational bound for simulating signatures in the classical signing oracle [15, 9, 19].

1.3 Open Problems

In our proofs of (W)PO/(s)BU security, a computational bound for the preimagesimulatability cannot be used. Since there are cryptographic schemes for which statistical properties cannot be achieved, the relaxation from the statistical bound to the computational one would expand the applicability of our security proofs. One possible way is assuming *quantum* preimage-simulatability as in the case of the Fiat-Shamir signatures [27]; however, this is a strong assumption.

1.4 Organization

Section 2 gives notations and definitions. Section 3 introduces the QROM and its existing proof techniques used for our proofs. Section 4 presents the new security proofs of DHSwBR.

2 Preliminaries

2.1 Notations and Terminology

For $n \in \mathbb{N}$, we let $[n] \coloneqq \{1, \ldots, n\}$. We write any symbol for sets in calligraphic font. For a finite set \mathcal{X} , $|\mathcal{X}|$ is the cardinality of \mathcal{X} and $U(\mathcal{X})$ is the uniform distribution over \mathcal{X} . By $x \leftarrow_{\$} \mathcal{X}$ and $x \leftarrow \mathcal{D}_{\mathcal{X}}$, we denote the sampling of an element from $U(\mathcal{X})$ and $\mathcal{D}_{\mathcal{X}}$ (distribution on \mathcal{X}). We denote a set of functions having a domain \mathcal{X} and a range \mathcal{Y} by $\mathsf{Func}(\mathcal{X}, \mathcal{Y})$. For a set of distributions over \mathcal{Y} indexed by $\mathcal{D} = \{\mathcal{D}_x : x \in \mathcal{X}\}$, we define $\mathsf{Func}_{\mathcal{X},\mathcal{Y}}(\mathcal{D})$ as a distribution of f in $\mathsf{Func}(\mathcal{X},\mathcal{Y})$ such that, for each $x \in \mathcal{X}$, f(x) is independently drawn from \mathcal{D}_x .

We write any symbol for functions in sans-serif font, oracles in small capitals, and adversaries in calligraphic font. If a function F is deterministic (resp., probabilistic), we write $y := \mathsf{F}(x)$ (resp., $y \leftarrow \mathsf{F}(x)$). We denote by $y \leftarrow \mathcal{A}^{\mathsf{ORCL}}(x)$ (resp., $y \leftarrow \mathcal{A}^{|\mathsf{ORCL}\rangle}(x)$) probabilistic computations of \mathcal{A} on input x with a classical (resp., quantum) oracle access to an oracle ORCL. For a random function H, we denote by $\mathsf{H}^{x^* \mapsto y^*}$ a function such that $\mathsf{H}^{x^* \mapsto y^*}(x) = \mathsf{H}(x)$ for $x \neq x^*$ and $\mathsf{H}^{x^* \mapsto y^*}(x^*) = y^*$. The notation $G^{\mathcal{A}} = y$ denotes an event in which a game Gplayed by \mathcal{A} returns y. For *i*-th game G_i , we denote W_i as an event $G_i^{\mathcal{A}} = 1$.

We denote 1 if the Boolean statement is true (\top) and 0 if the statement is false (\perp) . For a statement P, $\llbracket P \rrbracket$ denotes the truth value of P.

2.2 Digital Signature

We define the syntax of digital signature schemes as follows.

Definition 1 (Digital Signature). A digital signature scheme Sig consists of three algorithms:

- KeyGen (1^{λ}) : This algorithm takes 1^{λ} , where λ is the security parameter, as input and outputs a verification key vk and a signing key sk.
- Sign(sk, m): This algorithm takes a signing key sk and a message m as input and outputs a signature σ .
- Vrfy (vk, m, σ) : This algorithm takes a verification key vk, a message m, and a signature σ as input, and outputs \top (acceptance) or \perp (rejection).

Traditionally, the security of digital signatures is analyzed under EUF-CMA (Existential UnForgeability against Chosen-Message Attack) or its stronger variant, sEUF-CMA (strong EUF-CMA), both of which consider an adversary with access to a signing oracle attempting to forge a signature. Additionally, EUF-NMA (No Message Attack) is used, where the adversary does not have access to the signing oracle.

Definition 2 (Traditional Security of Signature). Let Sig be a signature scheme. Using games given in Fig. 2, we define advantage functions of adversaries playing EUF-CMA, SEUF-CMA and EUF-NMA games against Sig as $\operatorname{Adv}_{Sig}^{EUF-CMA}(\mathcal{A}) = \Pr[\mathsf{EUF-CMA}^{\mathcal{A}}=1], \operatorname{Adv}_{Sig}^{SEUF-CMA}(\mathcal{A}) = \Pr[\mathsf{SEUF-CMA}^{\mathcal{A}}=1], and \operatorname{Adv}_{Sig}^{EUF-NMA}(\mathcal{A}) = \Pr[\mathsf{EUF-NMA}^{\mathcal{A}}=1], respectively. We say Sig is EUF-CMA-secure, sEUF-CMA-secure, or EUF-NMA-secure if its corresponding advantage is negligible in the security parameter for any efficient adversary.$

Security models that allow quantum queries to the signing oracle, which are prohibited in traditional security models, have been actively studied in recent years. Boneh and Zhandry [6] defined the security notion called EUF-qCMA. We call the security notion as plus-one (PO) security following [1]. Also, we define its weakened version as wPO (weak PO) security.

$\frac{\text{GAME (s)EUF-CMA/EUF-NMA}}{1 \mathcal{Q} := \emptyset}$	$\frac{\text{SIGN}(m)}{10 \ \sigma \leftarrow} \text{Sign}(sk, m_i)$	
$\begin{array}{l} 2 (vk,sk) \leftarrow KeyGen(1^{\lambda}) \\ 3 (m^*,\sigma^*) \leftarrow \mathcal{A}^{\mathrm{SIGN}}(vk) //(\mathbf{s})EUF\text{-CMA} \\ 4 (m^*,\sigma^*) \leftarrow \mathcal{A}(vk) //EUF\text{-NMA} \\ 5 \mathbf{if} m^* \in \mathcal{Q} \mathbf{hen} //EUF\text{-CMA} \\ 6 \operatorname{return} \perp //EUF\text{-CMA} \\ 7 \mathbf{if} (m^*,\sigma^*) \in \mathcal{Q} \mathbf{hen} //sEUF\text{-CMA} \\ 8 \operatorname{return} \perp //sEUF\text{-CMA} \\ 9 \operatorname{return} \mathrm{Vrfy}(vk,m^*,\sigma^*) \end{array}$	11 $\mathcal{Q} \coloneqq \mathcal{Q} \cup \{m\}$ 12 $\mathcal{Q} \coloneqq \mathcal{Q} \cup \{(m, \sigma)\}$ 13 return σ	//EUF-CMA //sEUF-CMA

Fig. 2: (s)EUF-CMA and EUF-NMA games

GAME wPO/PO	$\operatorname{SIGN}(m)$
$\boxed{1 \mathcal{Q} := \emptyset}$	/* generate r on each query. */
2 $(vk, sk) \leftarrow \text{KeyGen}(1^{\wedge})$	r is fixed. */
$\begin{array}{c} 3 \text{run } \mathcal{A}_{1} = (vk) \\ 4 \text{return } \left[\mathcal{O} > a_{\text{s}} \right] \end{array}$	10 $\sigma := \operatorname{Sign}(sk, m; r)$
	11 return o
$\frac{\text{FORGE}(m,\sigma)}{5 \text{ if } Vrfv(vk,m,\sigma)} = \top \text{ then}$	
$\begin{vmatrix} \mathbf{s} & \ \mathbf{if} & \mathbf{m} \notin \mathcal{Q} \text{ then} \end{vmatrix} / l$	/wPO
$\begin{vmatrix} 7 & \mathcal{Q} \coloneqq \mathcal{Q} \cup \{m\} \\ \mathbf{s} & \mathbf{if} \ (m, \sigma) \ \mathcal{Q} \ \mathcal{O} \ \mathbf{then} \end{vmatrix} $	/wPO
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	//PO

Fig. 3: PO and wPO games

Definition 3 (Plus-One Unforgeability). Let Sig be a signature scheme. Using games given in Fig. 3, we define advantage functions of adversary playing PO and WPO games against Sig as $\operatorname{Adv}_{\operatorname{Sig}}^{\operatorname{PO}}(\mathcal{A}) = \Pr\left[\operatorname{PO}^{\mathcal{A}} = 1\right]$ and $\operatorname{Adv}_{\operatorname{Sig}}^{\operatorname{WPO}}(\mathcal{A}) = \Pr\left[\operatorname{wPO}^{\mathcal{A}} = 1\right]$. We say Sig is PO-secure or WPO-secure if its corresponding advantage is negligible in the security parameter for any efficient adversary.

In the PO game, the adversary must output $q_{\rm S} + 1$ distinct pairs of message/signature from $q_{\rm S}$ signing queries. In contrast, in the wPO game, the messages in the $q_{\rm S} + 1$ pairs must be distinct. Since the condition for a successful attack becomes more stringent, the security definition becomes weaker.

Alagic et al. [1] defined another security notion called blind unforgeability (BU security) and its stronger version sBU (strong BU).

Definition 4 (Blind Unforgeability). Let Sig be a signature scheme. Using games given in Fig. 4, we define advantage functions of adversary playing BU (Blind Unforgeability) and sBU (strong BU) games against Sig as $\operatorname{Adv}_{Sig}^{BU}(\mathcal{A}) = \Pr[\mathsf{BU}^{\mathcal{A}} = 1]$ and $\operatorname{Adv}_{Sig}^{SBU}(\mathcal{A}) = \Pr[\mathsf{sBU}^{\mathcal{A}} = 1]$. We say Sig is BU-secure or sBU-secure if its corresponding advantage is negligible in the security parameter for any efficient adversary.

The Security of Hash-and-Sign with Retry against Superposition Attacks

			-
GAME BU/sBU		B_{ϵ} SIGN (m)	
$1 B_{\epsilon} \leftarrow Func_{\mathcal{M},\{0,1\}}(Ber_{\epsilon})$	//BU	/* generate r on each query	. */
2 $B_{\epsilon} \leftarrow \operatorname{Func}_{M \times \Sigma} f_{0,11}(\operatorname{Ber}_{\epsilon})$	//sBU	/* for m queried in superpo	osition,
a win -		r is fixed.	*/
3 will $= \pm$		12 if $m \in B_{\epsilon}$ then	//BU
4 $(vk, sk) \leftarrow Sig.KeyGen(1^{\circ})$		13 return ⊥	//BU
5 run $\mathcal{A}^{ B_{\epsilon} \text{SIGN}\rangle, \text{FORGE}}(vk)$		14 $\sigma := \operatorname{Sig.Sign}(sk, m; r)$	
6 return win		15 if $(m, \sigma) \in B_{\epsilon}$ then	//sBU
		16 return \perp	//sBU
$FORGE(m, \sigma)$		17 return σ	
7 if Sig.Vrfv $(vk, m, \sigma) = \top$ then			
8 if $m \in B_c$ then	//BU		
$ \mathbf{y} \mathbf{win} \coloneqq \top$	//BU		
10 if $(m, \sigma) \in B$, then	//sBU		
11 win := T	//sBU		
	//300		

Fig. 4: BU and sBU games

GAME PRF_b	GAME $q PRF_b$
1 if $b = 0$ then	1 if $b = 0$ then
$\begin{array}{c c} 2 & k \leftarrow_{\$} \mathcal{K} \\ 3 & f \leftarrow_{\$} PRF(k, \cdot) \end{array}$	$\begin{array}{c c c c c c c c c c c c c c c c c c c $
4 else	4 else
$ 5 \mid f \leftarrow_{\$} Func(\mathcal{X}, \mathcal{Y}) $	$5 \mid f \leftarrow_{\$} Func(\mathcal{X}, \mathcal{Y})$
$\begin{array}{ccc} 6 & b & \leftarrow \mathcal{A}^{*} \\ 7 & \text{return } b^{*} \end{array}$	$\begin{array}{ccc} 6 & \mathbf{b}^* \leftarrow \mathcal{A}^{(j)}() \\ 7 & \mathbf{return} & \mathbf{b}^* \end{array}$

Fig. 5: PRF and qPRF games

Let $\epsilon \in \{0, \frac{1}{2^p}, ..., \frac{2^p-1}{2^p}\}$ for some parameter p. In the BU game, B_{ϵ} is a random subset of \mathcal{M} , where each $m \in B_{\epsilon}$ is independently selected with probability ϵ . The BU adversary attempts to find a valid pair (m, σ) such that $m \notin B_{\epsilon}$, given access to the signing oracle blinded by B_{ϵ} . In the sBU game, B_{ϵ} is a random subset of $\mathcal{M} \times \Sigma$, where Σ represents the signature space. Thus, messages are not blinded independently in the sBU game.

2.3 Pseudorandom Function

Definition 5 ((Quantum) Pseudorandom Function). Let $PRF: \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a deterministic function. Using games given in Fig. 5, we define advantage functions of adversaries playing the PRF and qPRF games against PRF as $Adv_{PRF}^{PRF}(\mathcal{A}) = |Pr[PRF_0^{\mathcal{A}}=1] - Pr[PRF_1^{\mathcal{A}}=1]|$ and $Adv_{PRF}^{qPRF}(\mathcal{A}) = |Pr[qPRF_0^{\mathcal{A}}=1] - Pr[qPRF_1^{\mathcal{A}}=1]|$. We say PRF is pseudorandom or quantum pseudorandom if its corresponding advantage is negligible in the security parameter for any efficient adversary.

2.4 Hash-and-Sign with Retry

We define the syntax of the trapdoor function (TDF) as follows.

GAME SPR	GAME CR
1 $(vk, sk) \leftarrow Gen(1^{\lambda})$	1 $(vk, sk) \leftarrow Gen(1^{\lambda})$
$2 \ \hat{x} \leftarrow \mathcal{D}_{\mathcal{X}}$	2 $(x_1^*, x_2^*) \leftarrow \mathcal{A}(vk)$
3 $x^* \leftarrow \mathcal{A}(vk, \hat{x})$	3 return
4 return $\llbracket x^* \neq \hat{x} \land F(vk, x^*) = F(vk, \hat{x}) \rrbracket$	$[\![x_1^* \neq x_2^* \wedge F(vk, x_1^*) = F(vk, x_2^*)]\!]$

Fig. 6: SPR and CR games

Definition 6 (Trapdoor Function). A TDF \top consists of three algorithms:

- Gen (1^{λ}) : This algorithm takes 1^{λ} , where λ is the security parameter, as input and a public key vk and a secret key sk.
- $\mathsf{F}(vk, x)$: This algorithm takes a public key vk and $x \in \mathcal{X}$ as input and deterministically outputs $y \in \mathcal{Y}$.

Inv(sk, y): This algorithm takes a secret key sk and $y \in \mathcal{Y}$ and outputs $x \in \mathcal{X}$ or \perp .

T is (γ, β) -correct if for every $(vk, sk) \leftarrow \text{Gen}(1^{\lambda})$, the following holds:

 $\Pr[y \leftarrow_{\$} \mathcal{Y}, x \leftarrow \mathsf{Inv}(sk, y) : \mathsf{F}(vk, x) = y | x \neq \bot] \ge \gamma,$ and $\Pr[y \leftarrow_{\$} \mathcal{Y}, x \leftarrow \mathsf{Inv}(sk, y) : x = \bot] \le \beta.$

There are some security notions for TDFs. In this paper, we use the following:

Definition 7 (Second-Preimage Resistance and Collision Resistance).

Let T be a TDF. Using games given in Fig. 6, we define advantage functions of adversaries playing the SPR (Second-Preimage-Resistance) and CR (Collision-Resistance) games against T as $\operatorname{Adv}_{T}^{\operatorname{SPR}}(\mathcal{A}) = \Pr[\operatorname{SPR}^{\mathcal{A}}=1]$ and $\operatorname{Adv}_{T}^{\operatorname{CR}}(\mathcal{A}) = \Pr[\operatorname{CR}^{\mathcal{A}}=1]$, respectively. We say T is second-preimage-resistant or collision-resistant if its corresponding advantage is negligible in the security parameter for any efficient adversary.

Let SampDom be a function to output $x \leftarrow \text{SampDom}(vk)$ that simulates Inv. By adding SampDom to the function set of Definition 6, we can define a preimage sampleable function (PSF) [12]. In this paper, we consider preimagesimulatability [19], which relaxes the conditions of a PSF.

Definition 8 (Preimage Simulatablity [19, **Definition 7**]). Let T be a *TDF with* SampDom. Using a game defined in Fig. 7, we define an advantage function of an adversary playing the PS (Preimage Sampling) game against T as $\operatorname{Adv}_{T}^{PS}(\mathcal{A}) = |\Pr[\mathsf{PS}_{0}^{\mathcal{A}}=1] - \Pr[\mathsf{PS}_{1}^{\mathcal{A}}=1]|$. We say T is preimage-simulatable if its advantage is negligible for any efficient adversary. Also, if SAMPLE₀ and SAMPLE₁ are δ -close ⁷, we say T is δ -PS.

Note that PSF is always preimage-simulatable since it can statistically simulate an honestly generated preimage without retry.

The Security of Hash-and-Sign with Retry against Superposition Attacks 11

GAME PS_b	$SAMPLE_0()$	SAMPLE ₁ ()
1 $(vk, sk) \leftarrow Gen(1^{\lambda})$	4 repeat	9 $x \leftarrow SampDom(vk)$
2 $b^* \leftarrow \mathcal{A}^{\text{SAMPLE}_b}(vk)$	$5 \mid y \leftarrow_{\$} \mathcal{Y}$	10 return x
\mathbf{s} return b^*	$6 \mid x \leftarrow Inv(sk, y)$	
	7 until $x \neq \bot$	
	s return x	

Fig. 7: PS game

$KeyGen(1^{\lambda})$	$KeyGen(1^{\lambda})$
$\boxed{1 \ (vk, sk)} \leftarrow Gen(1^{\lambda})$	$1 (vk, sk) \leftarrow Gen(1^{\lambda})$
2 return (vk, sk)	$2 \ (s,s') \leftarrow_{\$} \mathcal{K} \times \mathcal{K}$
Sign(sk,m)	3 return $(vk, (sk, s, s'))$
3 $k \coloneqq 0$	Sign((sk, s, s'), m)
4 repeat	$4 k \coloneqq 0$
5 k := k + 1	5 repeat
$\begin{array}{c c} 6 & r_k \leftarrow_{\$} \mathcal{R} \\ \end{array}$	$6 \mid \hat{k} := k + 1$
7 $y_k \coloneqq H(r_k, m)$	$7 \mid r_k \coloneqq PRF(s, (m, k))$
$\mathbf{s} \mid x_k \leftarrow Inv(sk, y_k)$	$\mathbf{s} \mid y_k \coloneqq H(r_k, m)$
9 until $x_k \neq \bot$	9 $x_k \coloneqq \operatorname{Inv}(sk, y_k; PRF'(s', (m, k)))$
10 return (r_k, x_k)	10 until $x_k \neq \perp \lor k > B$
Vrfy(vk, m, (r, x))	11 return (r_k, x_k)
$\boxed{11 \mathbf{return} \ \llbracket F(vk, x) = H(r, m) \rrbracket}$	$\underline{Vrfy(vk,m,(r,x))}$
	12 return $\llbracket F(vk, x) = H(r, m) \rrbracket$

Fig. 8: Algorithms of hash-and-sign with retry (HSwR) and derandomized hashand-sign with bounded retry (DHSwBR)

Kosuge and Xagawa [19] formulated a paradigm used in signature schemes proposed by Sakumoto et al. [23] as probabilistic hash-and-sign with retry, which we refer to in this paper as hash-and-sign with retry (HSwR). In this paper, we propose a variant referred to as derandomized hash-and-sign with bounded retry (DHSwBR), as shown in Fig. 8. Let HSR[T, H] and DHSR_B[T, H, PRF, PRF'] be HSwR and DHSwBR composing of a TDF T, a hash function $H: \mathcal{R} \times \mathcal{M} \to \mathcal{Y}$, and PRFs PRF: $\mathcal{K} \times \mathcal{M} \times [B] \to \mathcal{R}$ and PRF': $\mathcal{K} \times \mathcal{M} \times [B] \to \mathcal{R}'$, where B denotes the maximum number of retries. PRF generates a salt $r \in \mathcal{R}$ for H and PRF' generates a random coin used for derandomizing Inv.

3 Quantum Random Oracle Model (QROM) and Proof Techniques

In the ROM/QROM, a hash function $H: \mathcal{X} \to \mathcal{Y}$ is modeled as a random function $H \leftarrow_{\$} \mathsf{Func}(\mathcal{X}, \mathcal{Y})$. The random function is under the control of the challenger, and the adversary makes queries to the random oracle (random oracle

⁷ For distributions \mathcal{D} and \mathcal{D}' over $y \in \mathcal{Y}$, we say \mathcal{D} is δ -close to \mathcal{D}' if $\sum_{y \in \mathcal{Y}} |\mathcal{D}(y) - \mathcal{D}'(y)| \leq \delta$.

12 H. Kosuge and K. Xagawa

```
\begin{array}{l} \underline{\text{GAME GSPB}_{\lambda}} \\ \hline 1 \quad \{\lambda(x)\}_{x \in \mathcal{X}} \leftarrow \mathcal{A}_{1} \\ 2 \quad \text{if } \exists x \in \mathcal{X}, \lambda(x) > \lambda \text{ then} \\ 3 \quad | \text{ return } \bot \\ 4 \quad \text{for } x \in \mathcal{X} \text{ do} \\ 5 \quad | \quad g(x) \leftarrow \text{Ber}_{\lambda(x)} \\ 6 \quad x^{*} \leftarrow \mathcal{A}_{2}^{|g\rangle} \\ 7 \quad \text{return } g(x^{*}) \end{array}
```

Fig. 9: Generic search problem with bounded probabilities (GSPB)

GAME AR_b	$\operatorname{Repro}(x)$
$1 H_0 \leftarrow_{\$} Func(\mathcal{R} \times \mathcal{X}, \mathcal{Y})$	$5 r \leftarrow_{\$} \mathcal{R}$
$2 \ \mathbf{H}_1 \coloneqq \mathbf{H}_0$	$6 \ y \leftarrow_{\$} \mathcal{Y}$
3 $b^* \leftarrow \mathcal{A}^{ H_b\rangle, \operatorname{Repro}}()$	7 $H_1 \coloneqq H_1^{(r,x) \mapsto y}$
4 return b^*	\mathbf{s} return \dot{r}

Fig. 10: AR (Adaptive Reprogramming) game

queries) to compute the hash values. In the ROM, the challenger can choose $y \leftarrow_{\$} \mathcal{Y}$ and program $\mathsf{H} \coloneqq \mathsf{H}^{x \mapsto y}$ for queried x on-the-fly instead of choosing $\mathsf{H} \leftarrow_{\$} \mathsf{Func}(\mathcal{X}, \mathcal{Y})$ at the beginning (lazy sampling technique). In the QROM, the adversary makes queries to H in a superposition of many different values, e.g., $\sum_{x} \alpha_x |x\rangle |y\rangle$. The challenger computes H and gives a superposition of the results to the adversary, $\sum_{x} \alpha_x |x\rangle |y \oplus \mathsf{H}(x)\rangle$. Due to the nature of superposition queries and other constraints of quantum computation, traditional techniques in the ROM cannot be directly applied to the QROM. However, recent advancements in QROM research have led to the discovery of many proof techniques. This section introduces the techniques used in this paper.

Generic Quantum Search [28, 17, 18]: Let \mathcal{X} be a finite set. The generic search problem (GSP, in short) is finding $x \in \mathcal{X}$ satisfying g(x) = 1 given access to an oracle $g: \mathcal{X} \to \{0, 1\}$, where for each $x \in \mathcal{X}$, g(x) is drawn independently according to Ber_{λ} (Bernoulli distribution parameterized by λ).

Lemma 1. Let $\lambda \in [0, 1]$. For any quantum algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ making at most q queries to $|g\rangle$, we have

$$\Pr\left[\mathsf{GSPB}^{\mathcal{A}}_{\lambda} = 1\right] \le 8(q+1)^2\lambda,$$

where GSPB_{λ} is defined in Fig. 9.

Tight Adaptive Reprogramming [15]: Let $H_0, H_1: \mathcal{R} \times \mathcal{X} \to \mathcal{Y}$ be random functions. Fig. 10 shows a game called AR (Adaptive Reprogramming) game, in which the adversary \mathcal{A}_{ar} attempts to distinguish H_0 (no reprogramming) from H_1 (reprogrammed by REPRO). For a reprogramming query, the challenger reprograms H_1 for $r \leftarrow_{\$} \mathcal{R}$ and $y \leftarrow_{\$} \mathcal{Y}$, and gives r to \mathcal{A} . A distinguishing advantage of the AR game is defined by $\operatorname{Adv}_{H}^{AR}(\mathcal{A}_{ar}) = |\Pr[\mathsf{AR}_0^{\mathcal{A}} = 1] - \Pr[\mathsf{AR}_1^{\mathcal{A}} = 1]|$. **Lemma 2** (Tight Adaptive Reprogramming [15, Proposition 2]). For any quantum AR adversary \mathcal{A} issuing at most q_R classical reprogramming queries and q_H (quantum) random oracle queries to H_b , the distinguishing advantage of the AR game is bounded by

$$\operatorname{Adv}_{\mathsf{H}}^{\operatorname{AR}}(\mathcal{A}) \leq \frac{3}{2} q_{\mathsf{R}} \sqrt{\frac{q_{\mathsf{H}}}{|\mathcal{R}|}}.$$

One-way to Hiding (O2H) [25, 2]: We consider two functions H_0 and H_1 such that $H_0(x) = H_1(x)$ for $x \notin S$. We can show the indistinguishability of H_0 and H_1 using the following lemma.

Lemma 3 (Original O2H [2, Theorem 3]). Let $H_0, H_1: \mathcal{X} \to \mathcal{Y}$ be functions. Assume that $H_0(x) = H_1(x)$ for all $x \notin S$. Let z be a random bitstring. (S, H_0, H_1, z may have arbitrary joint distribution.) Let \mathcal{A} be a quantum algorithm with q quantum queries to H_0 or H_1 . Then, there exists a quantum algorithm \mathcal{B} that, given access to the oracle H_0 and \mathcal{A} , finds an element in S such that

$$\left|\Pr\left[\mathcal{A}^{|\mathsf{H}_0\rangle}(z)=1\right] - \Pr\left[\mathcal{A}^{|\mathsf{H}_1\rangle}(z)=1\right]\right| \le 2q\sqrt{\Pr\left[x\leftarrow \mathcal{B}^{|\mathsf{H}_0\rangle,\mathcal{A}}(z): x\in \mathcal{S}\right]}$$

When using the tight adaptive reprogramming shown in Lemma 2, Lemma 3 cannot be directly applied because it does not assume reprogrammed random functions. Therefore, we extend the original O2H as follows.

Lemma 4 (O2H with Adaptive Reprogramming). Let $H_0, H_1: \mathcal{X} \to \mathcal{Y}$ be functions that are reprogrammed depending on classical queries to an oracle O (H_0 and H_1 may be reprogrammed differently). Assume that $H_0(x) = H_1(x)$ for all $x \notin S$ when O is queried the same number of times with the same inputs. Let z be a random bitstring. (S, H_0, H_1, z may have arbitrary joint distribution.) Let \mathcal{A} be a quantum algorithm with q quantum queries to H_0 or H_1 and some classical queries to O. Then, there exists a quantum algorithm \mathcal{B} that, given access to the oracle H_0 and \mathcal{A} , finds an element in \mathcal{S} such that

$$\left|\Pr\left[\mathcal{A}^{|\mathsf{H}_{0}\rangle,\mathsf{O}}(z)=1\right]-\Pr\left[\mathcal{A}^{|\mathsf{H}_{1}\rangle,\mathsf{O}}(z)=1\right]\right| \leq 2q\sqrt{\Pr\left[x\leftarrow\mathcal{B}^{|\mathsf{H}_{0}\rangle,\mathsf{O},\mathcal{A}}(z):x\in\mathcal{S}_{i}\right]}$$

We show the proof in Appendix A. We can use the semi-classical O2H [2, Theorems 1 and 2] for the same purpose. Since the multiplicative factor of the technique is 4q, Lemma 4 is tighter by a factor of 2.

Other Techniques: We introduce two lemmas proven by Boneh and Zhandry [6].

Lemma 5 (Oracle Indistinguishability [6, Lemma 2.5, ePrint]). Let \mathcal{X} and \mathcal{Y} be two finite sets. Let $\mathcal{D} = \{\mathcal{D}_x\}$ and $\mathcal{D}' = \{\mathcal{D}'_x\}$ be two sets of efficiently sampleable distributions over \mathcal{X} indexed by $x \in \mathcal{X}$. Let \mathcal{A} be a quantum adversary making q (quantum) queries to an oracle $f: \mathcal{X} \to \mathcal{Y}$. If for each $x \in \mathcal{X}$, \mathcal{D}_x and \mathcal{D}'_x are ϵ -close, then

$$\left|\Pr\left[f \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(\mathcal{D}) : \mathcal{A}^{|f\rangle} = 1\right] - \Pr\left[f \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(\mathcal{D}') : \mathcal{A}^{|f\rangle} = 1\right]\right| \leq \sqrt{(6q)^3\epsilon}$$

Lemma 6 ([6, Lemma 2.6, ePrint]). Let \mathcal{X} and \mathcal{Y} be two finite sets. Fix a set \mathcal{D} of distributions \mathcal{D}_x over \mathcal{Y} indexed by $x \in \mathcal{X}$. Let α be the minimum over all $x \in \mathcal{X}$ of the min-entropy of the distribution \mathcal{D}_x and $f: \mathcal{X} \to \mathcal{Y}$ be a function chosen according to $\mathsf{Func}_{\mathcal{X},\mathcal{Y}}(\mathcal{D})$. Then, any q-query quantum algorithm can only produce (q+1) input/output pairs of f with probability at most $\frac{q+1}{|2^{\alpha}|}$.

4 Security Proofs for Derandomized Hash-and-sign with Bounded Retry

In this section, we show reductions from the EUF-NMA security of HSwR to the (w)PO, (s)BU, and (s)EUF-CMA security of DHSwBR. As Kosuge and Xagawa [19] have shown reductions from the non-invertibility of the underlying TDF to the EUF-NMA security of HSwR, these reductions are extended to the reduction from the non-invertibility. Note that our proofs are applied to derandomized probabilistic hash-and-sign by setting B = 1, where we can remove terms only related to DHSwBR from the bounds.

4.1 (Weak) Plus-One Unforgeability

We show that $\mathsf{DHSR}_B[\mathsf{T},\mathsf{H},\mathsf{PRF},\mathsf{PRF}']$ shown in Fig. 8 is PO-secure.

Theorem 1 (EUF-NMA + CR + qPRF \Rightarrow **PO).** For any quantum PO adversary \mathcal{A}_{po} of DHSR_B[T, H, PRF, PRF'] issuing at most q_H quantum queries to H, q_S quantum queries to SIGN, and q_F classical queries to FORGE, there exist an EUF-NMA adversary \mathcal{A}_{nma} of HSR[T, H], a CR adversary \mathcal{A}_{cr} of T, and qPRF adversaries \mathcal{A}_{prf} of PRF and \mathcal{A}'_{prf} of PRF' issuing at most Bq_S queries such that

$$\begin{aligned} \operatorname{Adv}_{\mathsf{DHSR}}^{\mathsf{PO}}(\mathcal{A}_{\mathsf{po}}) &\leq \operatorname{Adv}_{\mathsf{HSR}}^{\mathsf{EUF}-\mathsf{NMA}}(\mathcal{A}_{\mathsf{nma}}) + \operatorname{Adv}_{\mathsf{T}}^{\mathsf{CR}}(\mathcal{A}_{\mathsf{cr}}) + \operatorname{Adv}_{\mathsf{PRF}}^{q\operatorname{PRF}}(\mathcal{A}_{\mathsf{prf}}) \\ &+ \operatorname{Adv}_{\mathsf{PRF}'}^{q\operatorname{PRF}}(\mathcal{A}_{\mathsf{prf}}') + 8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^{2} \left(1 - \gamma \left(1 - \beta^{B}\right)\right) \\ &+ \frac{q_{\mathsf{S}} + 1}{\lfloor |\mathcal{R}| / B \rfloor} + 2(q_{\mathsf{H}} + q_{\mathsf{F}}) \sqrt{\frac{B - 1}{|\mathcal{R}|}} + 2(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B - 1)}{|\mathcal{R}|}} \\ &+ \sqrt{6 \left(q_{\mathsf{H}} + 2q_{\mathsf{F}}\right)^{3} \left(\delta + 2\left(1 - \gamma \left(1 - \beta^{B}\right)\right)\right)}, \end{aligned}$$

where T is (γ, β) -correct and δ -PS, and the running times of \mathcal{A}_{nma} , \mathcal{A}_{cr} , \mathcal{A}_{prf} , and \mathcal{A}'_{prf} are about that of \mathcal{A}_{po} .

Proof. We use the sequence of games shown in Fig. 11. Note that adversaries who simulate the games employ 2q-wise independent functions [29] to simulate random functions, and this applies to all the proofs in this paper as well.

GAME G_0 : This is the original PO game, where we execute GetLogs for SIGN, and H is defined as RF_{H} . We have $\Pr[W_0] = \operatorname{Adv}_{\mathsf{DHSR}}^{\mathsf{PO}}(\mathcal{A}_{\mathsf{PO}})$.

GAMES GO-GII	SIGN(m)
$\frac{\operatorname{GHR}}{1 \operatorname{RE}} \leftarrow_{\mathfrak{C}} \operatorname{Func}(\mathcal{R} \times \mathcal{M}, \mathcal{V})$	$\frac{\operatorname{sid}(m)}{\operatorname{ss}}$ if $\operatorname{Get}(m) = \operatorname{d}$ then $1/(\operatorname{Get}(m))$
2 RF _{salt} $\leftarrow_{\$}$ Func $(\mathcal{M} \times [B+1], \mathcal{R}) //G_1 - G_{11}$	$\begin{array}{c} 23 \text{ In Gettegs}(m) = 1 \text{ then} \\ 24 \text{ return } \bot \\ \end{array} // G_3 - G_8 \end{array}$
3 $RF_{inv} \leftarrow_{\$} Func(\mathcal{M} \times [B], \mathcal{R}') //G_1 - G_{10,0}$	25 $(r_k, y_k, x_k) \coloneqq \operatorname{GetLogs}(m)$
4 $RF'_{H} \leftarrow_{\$} Func(\mathcal{M} \times [B], \mathcal{Y})$ // G_7 - $G_{10,0}$	$//G_0 - G_{5.1} \cdot G_8 - G_{11}$
5 $RF_{sd} \leftarrow_{\$} Func(\mathcal{M}, \mathcal{R}'')$ // $G_{10,1}$ - G_{11}	26 $\{(r_i, y_i, x_i)\}_{i \in [k]} \coloneqq \text{GetLogs}(m) //G_6 - G_7$
$6 \mathcal{Q} \coloneqq \emptyset$	27 if $x_k = \bot$ then
7 win := \perp // $G_{5.0}$ - G_{11}	28 return \perp
$\mathbf{s} (vk, sk) \leftarrow Gen(1^{\lambda})$	29 return (r_k, x_k)
9 $(s,s') \leftarrow_{\$} \mathcal{K} \times \mathcal{K}$ // G_0	FORGE(m, (r, x))
10 run $\mathcal{A}_{no}^{ H\rangle, Sign\rangle,Forge}(vk)$	$\overline{\text{30 if GetLogs}(m)} = \exists \text{ then } //G_3 - G_8$
11 return $\llbracket \mathcal{Q} > q_{S} \rrbracket$ // G_0 - G_4	31 return \perp // G_3 - G_8
12 return $\llbracket \mathcal{Q} > q_{\mathrm{S}} \rrbracket \land \mathrm{win}$ // $G_{5.0}$	32 $(r_k, y_k, x_k) \coloneqq GetLogs(m)$
13 return win $//G_{5.1}-G_{11}$	$//G_2 - G_{5.1} \cdot G_8 - G_{11}$
	33 { (r_i, y_i, x_i) } $\in [k] := \text{GetLogs}(m) //G_6 - G_7$
$\left \frac{\Pi(r,m)}{160}\right = (1)$	34 If $F(vk, x_k) \neq H(r_k, m)$ then $7/G_4-G_{11}$
14 if $GetLogs(m) = \exists$ then $//G_3-G_8$	36 if $F(vk, x) = H(r, m)$ then
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	37 if $(m, (r, x)) \notin \mathcal{Q}$ then
$//G_2-G_{5,1} \cdot G_8-G_{11}$	$38 \mid \mathcal{Q} \coloneqq \mathcal{Q} \cup \{(m, (r, x))\}$
17 if $r = r_k$ then $//G_2 - G_{5.1} \cdot G_8 - G_{11}$	39 if $(r, x) \neq (r_k, x_k)$ then $//G_{5.0}$ - $G_{10.1}$
18 return y_k // G_2 - $G_{5.1} \cdot G_8$ - G_{11}	40 win = \top //G _{5.0} -G _{10.1}
19 $\{(r_i, y_i, x_i)\}_{i \in [k]} := \text{GetLogs}(m) //G_6 - G_7$	41 If $r \neq r_k$ then $//G_{11}$
20 if $\exists i, r = r_i$ then $//G_6 - G_7$	$42 will = 1$ // G_{11}
$\begin{bmatrix} 21 & \text{return } y_i \\ g_i & g_i \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ g_i & g_i \end{bmatrix}$	
22 return $KFH(r, m)$	
GetLogs (m) for G_0	$GetLogs(m)$ for G_1 - $G_{10.0}$
$\boxed{43 \ k \coloneqq 0}$	55 $k \coloneqq 0$
44 repeat	56 repeat
$45 k \coloneqq k + 1 \\ ppr((k + 1))$	57 $k \coloneqq k+1$
$\begin{array}{c} 46 \\ 47 \\ 47 \\ 47 \end{array} \stackrel{r_k := PRF(s, (m, k))}{RE} \\ \mathbf{RE}(r, m) \end{array}$	58 $r_k \coloneqq RF_{salt}(m,k)$ //G1-G9
$\begin{bmatrix} 4T \\ g_k & = \text{IN} \left((k, m) \right) \\ 4s & x_k & = \text{In} \left((k, m) \right) \\ y_k & = \text{In} \left((k, m) \right) \\ 0$	$g_{k} := \operatorname{RF}_{H}(r_{k}, m) \qquad 7761-66$
40 until $x_k \leftarrow m(sk, g_k, m(s, s_k, m, k)))$	61 $y_k := \ln(m, k)$ (m, k)
50 return (r_k, y_k, x_k)	62 until $x_k \neq \forall k > B$
	63 $r_k \coloneqq RF_{salt}(m, B+1)$ // $G_{10.0}$
GetLogs (m) for $G_{10.1}$ - G_{11}	64 if $\exists (i, j), r_i = r_j$ then $//G_3 - G_8$
$\boxed{51 \ r_k \coloneqq RF_{salt}(m, B+1)}$	65 return \exists // G_3 - G_8
52 $x_k \coloneqq SampDom(vk; RF_{sd}(m))$	66 return $(r_k, y_k, x_k) / G_1 - G_{5.1} \cdot G_8 - G_{10.0}$
$53 \ y_k := F(vk, x_k)$	67 return $\{(r_i, y_i, x_i)\}_{i \in [k]}$ // G ₆ -G ₇
54 return (r_k, y_k, x_k)	

The Security of Hash-and-Sign with Retry against Superposition Attacks

15

Fig. 11: Games for EUF-NMA \Rightarrow PO

GAME G_1 : We replace PRFs with random functions $\mathsf{RF}_{\mathsf{salt}}$ (salt generation) and $\mathsf{RF}_{\mathsf{inv}}$ (randomization of Inv).

Lemma 7. There exist qPRF adversaries \mathcal{A}_{prf} of PRF and \mathcal{A}'_{prf} of PRF' such that

$$|\Pr[W_0] - \Pr[W_1]| \le \mathrm{Adv}_{\mathsf{PRF}}^{q\mathrm{PRF}}(\mathcal{A}_{\mathsf{prf}}) + \mathrm{Adv}_{\mathsf{PRF}'}^{q\mathrm{PRF}}(\mathcal{A}'_{\mathsf{prf}}).$$

Proof. Since different keys are used for PRF and PRF', each can be replaced by a random function. The *q*PRF adversaries \mathcal{A}_{prf} and \mathcal{A}'_{prf} can simulate the outputs of PRF or PRF' using the outputs of their oracles. Thus, the advantage gap due to the above transformation is bounded by the *q*PRF advantages.

GAME G_2 : The random function H(r, m) computes $(r_k, y_k, x_k) := \text{GetLogs}(m)$ and returns y_k if $r = r_k$. Otherwise, H(r, m) returns $\text{RF}_H(r, m)$. Also, FORGE computes GetLogs. Since $H(r_k, m)$ is still computed by $\text{RF}_H(r_k, m)$, this modification changes nothing. Therefore, $\Pr[W_1] = \Pr[W_2]$ holds.

This is the first step in programming H to ensure that simulated signatures are accepted. Note that the game hops of G_2 - G_3 and G_6 - $G_{10.1}$ are dedicated to this purpose.

GAME G_3 : We modify GetLogs to check if there is a collision among $\{r_i\}_{i \in [k]}$. If a collision is detected, GetLogs returns a special symbol \exists . When GetLogs returns \exists , H, SIGN, and FORGE will return \bot , where we add \bot to the range of H. This step is required for G_8 , where we replace RF_H with RF'_H, which takes only (m, k) as input. This replacement becomes infeasible if a salt collision occurs. Additionally, in the next game hop, excluding salt collisions simplifies the bound.

Lemma 8. We have

$$|\Pr[W_2] - \Pr[W_3]| \le (q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B-1)}{|\mathcal{R}|}}$$

Proof. The difference between G_2 and G_3 lies solely in the outputs of GetLogs when there is a collision among $\{r_i\}_{i\in[k]}$. Hence, we apply the original O2H (see Lemma 3) to GetLogs in G_2 and G_3 . The outputs of GetLogs differ only in $S = \{m | \exists (i, j) \in [k] \times [k], r_i = r_j\}$, where $\{(r_i, y_i, x_i)\}_{i\in[k]}$ represents all intermediate and final results inside GetLogs(m). Let \mathcal{B}_{o2h} be an adversary that finds $m \in S$ by running \mathcal{A}_{cma} in G_3 . Since each r_i is generated by the random function RF_{salt} with distinct inputs, each r_i is uniformly distributed. Tightening the bound of [9, Lemma 11] slightly, the probability that $\{r_i\}_{i\in[k]}$ contains a collision is bounded by $\frac{\mathcal{B}(B-1)}{2|\mathcal{R}|}$. Given that \mathcal{B}_{o2h} has no information about S, the bound in this lemma follows from Lemma 3.

GAME G_4 : We modify FORGE such that it returns \exists if $\mathsf{F}(vk, x_k) \neq \mathsf{H}(r_k, m)$ holds for $(r_k, y_k, x_k) \coloneqq \mathsf{GetLogs}(m)$ before checking the validity of the submitted query. This step is necessary for the simulation by the CR adversary in bounding the advantage gap between $G_{10.1}$ and G_{11} . Note that G_4 - $G_{5.1}$ and G_{11} are dedicated to ensuring that the EUF-NMA adversary simulating G_{11} can obtain a winning message/signature pair from those in \mathcal{Q} .

Lemma 9. Suppose that T is (γ, β) -correct. We have

$$|\Pr[W_3] - \Pr[W_4]| \le 8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^2 (1 - \gamma (1 - \beta^B)).$$

Proof. If FORGE does not return \exists , then G_3 and G_4 are indistinguishable. Let **bad**₄ be an event where $\mathsf{F}(vk, x_k) \neq \mathsf{H}(r_k, m)$ holds (see Line 34 in Fig. 11) and FORGE returns \exists in G_4 . Let $\mathcal{B}_{\mathsf{gspb}} = (\mathcal{B}_1, \mathcal{B}_2)$ be a GSPB adversary shown in Fig. 12, where the target function g outputs 1 if and only if the output of

$\underline{\mathcal{B}_1}$	$\frac{RF_{salt}(m,k)}{RF_{salt}(m,k)}$
1 $(vk, sk) \leftarrow Gen(1^{\lambda})$	15 $\{(r_i, y_i, r'_i)\}_{i \in [B]} \coloneqq Samp(m)$
² compute S_{all} and S_{bad}	16 return r_k
3 $orall m \in \mathcal{M}, \ \lambda_{sk}(m) \coloneqq rac{ \mathcal{S}_{bad} }{ \mathcal{S}_{all} }$	$RF_{H}(r,m)$
4 return $\{\lambda_{sk}(m)\}_{m\in\mathcal{M}}, (vk, sk)$	$\frac{1}{17} \frac{1}{\{(r_i, y_i, r'_i)\}_{i \in [B]}} \coloneqq Samp(m)$
$ \mathbf{p} g\rangle$	18 if $\exists i, r = r_i$ then
$\left \frac{\mathcal{B}_{2}^{-+}}{2}\right $	19 return y_i
$5 \text{ RF}_{U} \leftarrow_{\$} \text{Func}(\mathcal{M}, \mathcal{R}_{U})$	20 return $RF''_{\mu}(r,m)$
6 $RF''_{H} \leftarrow_{\$} Func(\mathcal{R} \times \mathcal{M}, \mathcal{Y})$	
$7 \ \hat{m} \coloneqq 0$	$RF_{inv}(m,k)$
s run $\mathcal{A}^{ H\rangle, Sign\rangle,Forge}(vk)$	$\overline{21} \{(r_i, y_i, r'_i)\}_{i \in [B]} := \text{Samp}(m)$
9 return \hat{m}	22 return r'_1
	k
$\underline{Samp(m)}$	Forge(m,(r,x))
10 if $g(m) = 0$ then	23 $\{(r_i, y_i, x_i)\}_{i \in [k]} \coloneqq GetLogs(m)$
$11 \{(r_i, y_i, r'_i)\}_{i \in [B]} \coloneqq U(\mathcal{S}_{all}; RF_{U}(m))$	24 if $F(vk, x_k) \neq H(r_k, m)$ then
12 if $g(m) = 1$ then	$25 \mid \hat{m} \coloneqq m$
$13 \mid \{(r_i, y_i, r'_i)\}_{i \in [B]} \coloneqq U(\mathcal{S}_{bad}; RF_{U}(m))$	
14 return $\{(r_i, y_i, r'_i)\}_{i \in [B]}$	

Fig. 12: Simulation by GSPB adversary

 $\mathsf{GetLogs}(m)$ is invalid, that is, $x_i = \bot$ for all $i \in [B]$ or there exist i such that $x_i \neq \bot \land \mathsf{F}(vk, x_i) \neq y_i$. We define $\mathcal{S}_{\mathsf{all}} \subset (\mathcal{R} \times \mathcal{Y} \times \mathcal{R}')^B$ as:

$$S_{\text{all}} = \{\{(r_i, y_i, r'_i)\}_{i \in [B]} | \forall i, j \in [B], r_i = r_j \Rightarrow y_i = y_j\}.$$

Note that S_{all} is consistent between r_i and y_i . Such consistency is required to simulate RF_{H} since $\mathsf{RF}_{\mathsf{H}}(r_i, m) = \mathsf{RF}_{\mathsf{H}}(r_j, m)$ must hold if $r_i = r_j$. Then, we define S_{bad} as:

$$\mathcal{S}_{\mathsf{bad}} = \left\{ \{ (r_i, y_i, r'_i) \}_{i \in [B]} \in \mathcal{S}_{\mathsf{all}} \middle| \begin{array}{l} x_i \coloneqq \mathsf{Inv}(sk, y_i; r'_i) : (\forall i \in [B], x_i = \bot) \\ \lor (\exists i \in [B], \ x_i \neq \bot \land \mathsf{F}(vk, x_i) \neq y_i) \end{array} \right\}.$$

 \mathcal{B}_1 sets $\lambda_{sk}(m) = \lambda_{sk} = \frac{|S_{bad}|}{|S_{all}|}$ for all m. Using the oracle access to g, \mathcal{B}_2 defines a function Samp which outputs $\{(r_i, y_i, r'_i)\}_{i \in [B]}$ according to the value of g(m). Samp uniformly chooses $\{(r_i, y_i, r'_i)\}_{i \in [B]}$ from \mathcal{S}_{all} or \mathcal{S}_{bad} , where the uniformity is ensured by a random function RF_U. Since Samp is used to simulate the random functions, RF_{salt}, RF_H, and RF_{inv}, \mathcal{B}_2 can simulate G_4 . \mathcal{B}_2 outputs \hat{m} that stores m satisfying $\mathsf{F}(pk, x_k) \neq \mathsf{H}(r_k, m)$ in Line 25 of Fig. 12.

If **bad**₄ occurs, **GetLogs** does not output \exists (see Lines 30 and 31 in Fig. 11); therefore, the salts $\{r_i\}_{i\in[B]}$ do not collide in **GetLogs**. Hence, we can assume that each element of $\{y_i\}_{i\in[B]}$ is randomly generated and we have $\operatorname{Exp}[\lambda_{sk}] \leq$ $1 - \gamma (1 - \beta^B)^8$. From Lemma 1, fixing (vk, sk), we have $\Pr[\mathbf{bad}_4|(vk, sk)] \leq$

⁸ In [26, Lemma 5.2], which this proof is based on, a difference lies in consideration of potential collisions among $\{r_i\}_{i \in [B]}$. By not considering collisions, we remove the need to add an extra term related to collisions to the bound on $\text{Exp}[\lambda_{sk}]$.

 $8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^2 \lambda_{sk}$. Averaging over keys, we have $\Pr[\mathbf{bad}_4] \leq 8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^2 (1 - \gamma (1 - \beta^B))$, and complete the proof.

GAME $G_{5.0}$: Let **win** be an event that $(r, x) \neq (r_k, x_k)$ holds for queried (r, x)and $(r_k, y_k, x_k) \coloneqq \text{GetLogs}(m)$ in FORGE. In $G_{5.0}$, **win** = \top is necessary to win. With this modification, the adversary must forge at least one signature that is not derived from the message m in order to achieve **win** = \top .

Lemma 10. We have

$$|\Pr[W_4] - \Pr[W_{5.0}]| \le \frac{q_{\mathsf{S}} + 1}{\lfloor |\mathcal{R}|/B \rfloor}$$

Proof. G_4 and $G_{5.0}$ differ when $|\mathcal{Q}| > q_S$ and $\min = \bot$ hold simultaneously (i.e., $(r, x) = (r_k, x_k)$ holds for all $q_S + 1$ queries). We define this event as **bad**₅. The event **bad**₅ implies that the adversary obtains at least $q_S + 1$ input/output pairs of $\mathsf{RF}_{\mathsf{salt}}$. The outputs of $\mathsf{RF}_{\mathsf{salt}}$ are obtained only through SIGN queries. Therefore, the adversary produces $q_S + 1$ input/output pairs from q_S queries to $\mathsf{RF}_{\mathsf{salt}}$ when bad_5 occurs. From [26, Proposition 4.1], we have $\max_{r \in \mathcal{R}} \Pr[r_k \coloneqq \mathsf{RF}_{\mathsf{salt}}(m, k) : r_k = r|\mathsf{H}] \leq \frac{B}{|\mathcal{R}|}$ for any m, where H in the condition denotes that the adversary knows the whole table of H . Therefore, the probability of bad_5 is bounded by $\frac{q_S+1}{||\mathcal{R}|/B|}$ from Lemma 6.

- GAME $G_{5.1}$: We eliminate $[[|\mathcal{Q}| > q_S]]$ from the winning condition. Since the winning condition is relaxed, we have $\Pr[W_{5.0}] \leq \Pr[W_{5.1}]$. Then, the adversary can win the game without submitting $q_5 + 1$ valid message/signature pairs.
- GAME G_6 : GetLogs outputs intermediate/final results $\{(r_i, y_i, x_i)\}_{i \in [k]}$ generated during loop iteration instead of outputting only the final result. Moreover, H(r, m) outputs y_i if there exists *i* such that $r = r_i$. This modification does not affect the adversary's view, and we have $\Pr[W_{5.1}] = \Pr[W_6]$. This step is necessary for replacing RF_{H} in GetLogs in the next step.
- GAME G_7 : We change the way of generating y_k in GetLogs from $\mathsf{RF}_{\mathsf{H}}(r_k, m)$ to $\mathsf{RF}'_{\mathsf{H}}(m, k)$. Since there are no collisions among $\{r_i\}_{i \in [k]}$, both $\mathsf{RF}_{\mathsf{H}}(r_k, m)$ and $\mathsf{RF}'_{\mathsf{H}}(m, k)$ follow the uniform distribution. Therefore, the view of the adversary does not change, and $\Pr[W_6] = \Pr[W_7]$ holds. Due to this modification, the value of y_k is independently chosen for each (m, k), satisfying one of the necessary conditions for signature simulation using δ -PS.
- GAME G_8 : We modify **GetLogs** so that it outputs only the final result (r_k, y_k, x_k) . Then, only the outputs for which simulation is possible using δ -PS have been programmed for H; thus, the preparation for simulation is complete.

Lemma 11. We have

$$|\Pr[W_7] - \Pr[W_8]| \le 2(q_{\mathsf{H}} + q_{\mathsf{F}}) \sqrt{\frac{B-1}{|\mathcal{R}|}}.$$

Proof. This modification only affects the outputs of H. For $\{(r_i, y_i, x_i)\}_{i \in [k]}$ generated inside $\mathsf{GetLogs}(m)$, we define $S = \{(r, m) | \exists i \in [k-1], r = r_i\}$. As the outputs of $\mathsf{H}(r, m)$ are different between G_7 and G_8 if and only if $(r, m) \in S$, we apply the original O2H shown in Lemma 3. Let \mathcal{B}'_{o2h} be an adversary that executes $\mathcal{A}_{\mathsf{cma}}$ in G_8 and identifies an element in S. Since \mathcal{B}'_{o2h} has no prior information about S, the probability that \mathcal{B}'_{o2h} outputs $(r, m) \in S$ is at most $\frac{B-1}{|\mathcal{R}|}$. Following Lemma 3, we obtain the bound $2(q_{\mathsf{H}} + q_{\mathsf{F}})\sqrt{\frac{B-1}{|\mathcal{R}|}}$. Note that H is called twice in FORGE in Lines 34 and 36. However, $(r_k, m) \notin S$ is always true since it is guaranteed that there are no collisions among $\{r_i\}_{i\in[k]}$. At the time when GetLogs is executed, it is known that there are no collisions among $\{r_i\}_{i\in[k]}$. Therefore, the H-query in Line 34 can be excluded from consideration.

GAME G_9 : We remove the collision check among $\{r_i\}_{i \in [k]}$ from GetLogs since, from the next hop, GetLogs is modified not to perform loop iterations. From Lemma 8, we have

$$|\Pr[W_8] - \Pr[W_9]| \le (q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B-1)}{|\mathcal{R}|}}$$

GAME $G_{10.0}$: In GetLogs, $r_k := \mathsf{RF}_{\mathsf{salt}}(m,k)$ is generated by $\mathsf{RF}_{\mathsf{salt}}(m,B+1)$. Note that we modify GetLogs to be performed without using sk in $G_{10.0}$ and $G_{10.1}$.

Lemma 12. We have $\Pr[W_9] = \Pr[W_{10.0}]$.

Proof. Since both $\mathsf{RF}_{\mathsf{salt}}(m, k)$ and $\mathsf{RF}_{\mathsf{salt}}(m, B+1)$ are uniformly distributed, and the adversary obtains only $\mathsf{RF}_{\mathsf{salt}}(m, k)$ (resp., $\mathsf{RF}_{\mathsf{salt}}(m, B+1)$) in G_9 (resp., $G_{10.0}$), the adversary's view remains unchanged.

GAME $G_{10.1}$: We simulate GetLogs using SampDom, where $\mathsf{RF}_{\mathsf{sd}} \leftarrow_{\$} \mathsf{Func}(\mathcal{M}, \mathcal{R}'')$ is used for generating a random coin for SampDom.

Lemma 13. Suppose that T is (γ, β) -correct and δ -PS. We have

$$|\Pr[W_{10.0}] - \Pr[W_{10.1}]| \le \sqrt{6 (q_{\mathsf{H}} + 2q_{\mathsf{F}})^3 (\delta + 2 (1 - \gamma (1 - \beta^B)))}.$$

Proof. We consider the oracle-indistinguishability of GetLogs in $G_{10.0}$ and $G_{10.1}$ by considering the difference in distributions of $(x, y) \in \mathcal{X}' \times \mathcal{Y}$ output from GetLogs, where $\mathcal{X}' = \mathcal{X} \cup \{\bot\}$. Let \mathcal{D}_m and \mathcal{D}'_m be distributions of $(x, y) \in \mathcal{X}' \times \mathcal{Y}$ output from GetLogs(m) in $G_{10.0}$ and $G_{10.1}$, respectively. Since T is δ -PS (see Definition 8), the distance between x generated by Inv after unbounded retries and $x \leftarrow \text{SampDom}(vk)$ is bounded by δ . We define \mathcal{D}_m^{∞} as the distribution of $(x, y, k) \in \mathcal{X} \times \mathcal{Y} \times \mathbb{Z}_{>0}$, where GetLogs(m) retries without any limit, and an additional variable k denotes the number of retries within GetLogs(m). By marginalizing \mathcal{D}_m^{∞} and \mathcal{D}'_m over $y \in \mathcal{Y}$, we have

$$\sum_{x \in \mathcal{X}} \left| \sum_{y \in \mathcal{Y}} \left(\sum_{k \in \mathbb{Z}_{>0}} \mathcal{D}_m^{\infty}(x, y, k) - \mathcal{D}_m'(x, y) \right) \right| \le \delta.$$
(1)

If GetLogs in $G_{10.0}$ outputs $x \neq \bot$, then the number of retries will be less than or equal to B; therefore, for any $(x, y) \in \mathcal{X} \times \mathcal{Y}$, we have

$$\mathcal{D}_m(x,y) = \sum_{k \in [B]} \mathcal{D}_m^{\infty}(x,y,k).$$
(2)

In addition, the probability of outputting $x \neq \bot$ such that $\mathsf{F}(vk, x) = y$ is at least $\gamma(1 - \beta^B)$ due to the (γ, β) -correctness. Therefore, we have

$$\sum_{x \in \mathcal{X}} \mathcal{D}_m(x, \mathsf{F}(vk, x)) \ge \gamma \left(1 - \beta^B\right).$$
(3)

Then, we can derive a bound on $\delta' = \sum_{(x,y) \in \mathcal{X}' \times \mathcal{Y}} |\mathcal{D}_m(x,y) - \mathcal{D}'_m(x,y)|$ as follows.

$$\begin{split} \delta' &= \sum_{\substack{(x,y) \in \mathcal{X}' \times \mathcal{Y} \\ :x \neq \perp \wedge \mathsf{F}(vk,x) = y}} |\mathcal{D}_m(x,y) - \mathcal{D}'_m(x,y)| + \sum_{\substack{(x,y) \in \mathcal{X}' \times \mathcal{Y} \\ :x = \perp \vee \mathsf{F}(vk,x) \neq y}} \mathcal{D}_m(x,y) \\ &= \sum_{x \in \mathcal{X}} |\mathcal{D}_m(x,\mathsf{F}(vk,x)) - \mathcal{D}'_m(x,\mathsf{F}(vk,x))| + 1 - \sum_{x \in \mathcal{X}} \mathcal{D}_m(x,\mathsf{F}(vk,x)) \\ \stackrel{(2)}{=} \sum_{x \in \mathcal{X}} \left| \sum_{k \in [B]} \mathcal{D}_m^{\infty}(x,\mathsf{F}(vk,x),k) - \mathcal{D}'_m(x,\mathsf{F}(vk,x)) \right| + 1 - \sum_{x \in \mathcal{X}} \mathcal{D}_m(x,\mathsf{F}(vk,x)) \\ \stackrel{(*)}{\leq} \sum_{x \in \mathcal{X}} \left| \sum_{y \in \mathcal{Y}} \left(\sum_{k \in \mathbb{Z}_{>0}} \mathcal{D}_m^{\infty}(x,y,k) - \mathcal{D}'_m(x,y) \right) \right| + 1 - \sum_{x \in \mathcal{X}} \mathcal{D}_m(x,\mathsf{F}(vk,x)) \\ &+ \sum_{x \in \mathcal{X}} \left(\sum_{y \in \mathcal{Y}} \sum_{k \in \mathbb{Z}_{>0}} \mathcal{D}_m^{\infty}(x,y,k) - \sum_{k \in [B]} \mathcal{D}_m^{\infty}(x,\mathsf{F}(vk,x),k) \right) \\ \stackrel{(1)(2)}{\leq} \delta + 2 \left(1 - \sum_{x \in \mathcal{X}} \mathcal{D}_m(x,\mathsf{F}(vk,x)) \right) \stackrel{(3)}{\leq} \delta + 2 \left(1 - \gamma \left(1 - \beta^B \right) \right) \end{split}$$

Here, (*) follows from $\mathcal{D}'_m(x,\mathsf{F}(vk,x)) = \sum_{y\in\mathcal{Y}} \mathcal{D}'_m(x,y)$ and the triangle inequality, where we do not take the absolute value for the last term because $\sum_{y\in\mathcal{Y}}\sum_{k\in\mathbb{Z}_{>0}}\mathcal{D}^\infty_m(x,y,k)$ includes all the terms of $\sum_{k\in[B]}\mathcal{D}^\infty_m(x,\mathsf{F}(vk,x),k)$. Using Lemma 5, we can derive the bound on $|\Pr[W_{10.0}] - \Pr[W_{10.1}]|$.

GAME G_{11} : We change the condition of $\mathbf{win} = \top$ from $(r, x) \neq (r_k, x_k)$ to $r \neq r_k$ in FORGE. Though the condition $(r, x) \neq (r_k, x_k)$ allows for the possibility that H may not match RF_{H} for a queried pair (r, m) in FORGE, this modification ensures that they do. Then, all the necessary conditions for the simulation by the EUF-NMA adversary are now complete.

Lemma 14. There exists a CR adversary \mathcal{A}_{cr} of T such that

$$|\Pr[W_{10.1}] - \Pr[W_{11}]| \le \operatorname{Adv}_{\mathsf{T}}^{\operatorname{CR}}(\mathcal{A}_{\mathsf{cr}}).$$

The	Security	of Hash-and-Sign	with Retry	against S	Superposition At	tacks 21
	•/		•/		1 1	

$\frac{\mathcal{A}_{nma}^{\hat{H}}(vk)}{1 \ RF_{salt} \leftarrow_{\$} \ Func(\mathcal{M} \times [B+1], \mathcal{R})}$ $2 \ RF_{sd} \leftarrow_{\$} \ Func(\mathcal{M}, \mathcal{R}'')$ $3 \ win = \bot$ $sum \ \mathcal{A}^{ H\rangle, SIGN\rangle,FORGE}(uk)$	$\frac{\text{SIGN}(m)}{12 (r_k, y_k, x_k)} \coloneqq \text{GetLogs}(m)$ 13 if $x_k = \bot$ then 14 return \bot 15 return (r_k, x_k)
5 if win then 6 return $(m^*, (r^*, x^*))$ 7 return \perp	$\frac{\text{FORGE}(m, (r, x))}{\text{16} (r_k, y_k, x_k) := \text{GetLogs}(m)}$ 17 if $F(vk, x_k) \neq H(r_k, m)$ then
$\begin{array}{c} \displaystyle \frac{H(r,m)}{\mathbf{s}~(r_k,y_k,x_k)}\coloneqq GetLogs(m)\\ \mathbf{s}~\mathrm{if}~r=r_k~\mathrm{then} \end{array}$	18 return \exists 19 if $F(vk, x) = H(r, m) \land r \neq r_k$ then 20 win = \top 21 $(m^*, (r^*, x^*)) := (m, (r, x))$
10 return y_k 11 return $\hat{H}(r, m)$	$\begin{array}{l} \underline{GetLogs(m)} \\ \underline{22} r_k \coloneqq RF_{salt}(m, B+1) \\ \underline{23} x_k \coloneqq SampDom(vk; RF_{sd}(m)) \\ \underline{24} y_k \coloneqq F(vk, x_k) \end{array}$
	25 return (r_k, y_k, x_k)

Fig. 13: Simulation of the modified PO game by EUF-NMA adversary

Proof. $G_{10,1}$ and G_{11} differ only if the adversary submits (r, x) such that $r = r_k$ and $x \neq x_k$ (i.e., **win** = \top holds only in $G_{10,1}$). Let **bad**_{11} denote this event, and note that $|\Pr[W_{10,1}] - \Pr[W_{11}]| \leq \Pr[\mathbf{bad}_{11}]$ holds.

We now bound $\Pr[\mathbf{bad}_{11}]$. Due to the modification introduced in G_4 , we have $\mathsf{H}(r_k, m) = \mathsf{F}(vk, x_k)$ when FORGE does not return \exists . Therefore, when \mathbf{bad}_{11} occurs, a collision pair (x, x_k) satisfying $\mathsf{F}(vk, x) = \mathsf{F}(vk, x_k) = \mathsf{H}(r, m)$ is found. If \mathbf{bad}_{11} occurs during the CR adversary's simulation, the adversary can obtain the colliding pair (x, x_k) . Thus, we conclude that $\Pr[\mathbf{bad}_{11}] \leq \operatorname{Adv}_{\mathsf{T}}^{\mathrm{CR}}(\mathcal{A}_{\mathsf{cr}})$.

We conclude the proof by the EUF-NMA adversary's simulation.

Lemma 15. There exists an EUF-NMA adversary \mathcal{A}_{nma} of $\mathsf{HSR}[\mathsf{T},\mathsf{H}]$ such that

$$\Pr[W_{11}] \leq \operatorname{Adv}_{\mathsf{HSR}}^{\mathrm{EUF-NMA}}(\mathcal{A}_{\mathsf{nma}}).$$

Proof. To avoid a confusion, we consider $\mathsf{HSR}[\mathsf{T}, \hat{\mathsf{H}}]$ instead of $\mathsf{HSR}[\mathsf{T}, \mathsf{H}]$, where $\hat{\mathsf{H}}: \mathcal{R} \times \mathcal{M} \to \mathcal{Y}$ is a random oracle. The EUF-NMA adversary $\mathcal{A}_{\mathsf{nma}}$ against $\mathsf{HSR}[\mathsf{T}, \hat{\mathsf{H}}]$ with oracle access to $\hat{\mathsf{H}}$ can simulate G_{11} since it can simulate all the oracles by using SampDom as in Fig. 13. The EUF-NMA adversary outputs $(m^*, (r^*, x^*))$ such that $\mathsf{F}(vk, x^*) = \mathsf{H}(r^*, m^*)$ and $r^* \neq r_k = \mathsf{RF}_{\mathsf{salt}}(m^*, B+1)$ hold in FORGE. Note that $\mathsf{H}(r^*, m^*) = \hat{\mathsf{H}}(r^*, m^*)$ holds since $r^* \neq r_k$. Therefore, $\mathcal{A}_{\mathsf{nma}}$ can win the game and $\Pr[W_{11}]$ is bounded by the EUF-NMA advantage. Since $\mathsf{HSR}[\mathsf{T}, \hat{\mathsf{H}}]$ is equivalent to $\mathsf{HSR}[\mathsf{T}, \mathsf{H}]$, this shows the lemma.

The wPO-security does not require the collision-resistance of T as follows.

Corollary 1 (EUF-NMA + **qPRF** \Rightarrow **wPO).** For any quantum wPO adversary \mathcal{A}_{po} of DHSR_B[T, H, PRF, PRF'] issuing at most q_{H} quantum queries to H, q_{S} quantum queries to SIGN, and q_{F} classical queries to FORGE, there exist an EUF-NMA adversary \mathcal{A}_{nma} of HSR[T, H] and qPRF adversaries \mathcal{A}_{prf} of PRF and \mathcal{A}'_{prf} of PRF' issuing at most Bq_{S} queries such that

$$\begin{aligned} \operatorname{Adv}_{\mathsf{DHSR}}^{\mathrm{wPO}}(\mathcal{A}_{\mathsf{po}}) &\leq \operatorname{Adv}_{\mathsf{HSR}}^{\mathrm{EUF}\operatorname{-NMA}}(\mathcal{A}_{\mathsf{nma}}) + \operatorname{Adv}_{\mathsf{PRF}}^{q\mathrm{PRF}}(\mathcal{A}_{\mathsf{prf}}) + \operatorname{Adv}_{\mathsf{PRF}}^{q\mathrm{PRF}}(\mathcal{A}_{\mathsf{prf}}') \\ &+ 8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^{2} \left(1 - \gamma \left(1 - \beta^{B}\right)\right) + \frac{q_{\mathsf{S}} + 1}{\lfloor |\mathcal{R}| / B \rfloor} \\ &+ 2(q_{\mathsf{H}} + q_{\mathsf{F}}) \sqrt{\frac{B - 1}{|\mathcal{R}|}} + 2(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B - 1)}{|\mathcal{R}|}} \\ &+ \sqrt{6 \left(q_{\mathsf{H}} + 2q_{\mathsf{F}}\right)^{3} \left(\delta + 2 \left(1 - \gamma \left(1 - \beta^{B}\right)\right)\right)}, \end{aligned}$$

where T is (γ, β) -correct and δ -PS, and the running times of \mathcal{A}_{nma} , \mathcal{A}_{prf} , and \mathcal{A}'_{prf} are about that of \mathcal{A}_{po} .

Proof. In the proof of Theorem 1, we can modify $G_{5,0}$ by changing the winning condition from $(r, x) \neq (r_k, x_k)$ to $r \neq r_k$ (see Line 39 in Fig. 11) and remove G_{11} . Note that G_{11} is unnecessary for the simulation by the EUF-NMA adversary since the condition of **win** = \top has already been F(vk, x) = H(r, m) and $r \neq r_k$ in FORGE. Hence, the collision-resistance of \top is not necessary.

Remark 1. By assuming quantum preimage-simulatability, we can use a computational bound. This is an adaptation of the quantum special HVZK used in the BU security proof of Fiat-Shamir [27] to the HSwR context. We define quantum preimage-simulatability by allowing quantum queries in the PS game. We modify the oracles in the PS game to take a message as input and perform preimage sampling corresponding to the message. The advantage of the quantum version of the PS game can be used to bound $|\Pr[W_{10.0}] - \Pr[W_{10.1}]|$ in Theorem 1. See Appendix C for details.

4.2 (Strong) Blind Unforgeability

We show that $\mathsf{DHSR}_B[\mathsf{T},\mathsf{H},\mathsf{PRF},\mathsf{PRF}']$ is also sBU-secure.

Theorem 2 (EUF-NMA + CR + qPRF \Rightarrow **sBU).** For any quantum SBU adversary \mathcal{A}_{bu} of DHSR_B[T, H, PRF, PRF'] issuing at most q_{H} quantum queries to H, q_{S} quantum queries to SIGN, and q_{F} classical queries to FORGE, there exist an EUF-NMA adversary \mathcal{A}_{nma} of HSR[T, H], a CR adversary \mathcal{A}_{cr} of T, and qPRF adversaries \mathcal{A}_{prf} of PRF and \mathcal{A}'_{prf} of PRF' issuing at most Bq_{S} queries such that

$$\begin{aligned} \operatorname{Adv}_{\mathsf{DHSR}}^{\operatorname{SBU}}(\mathcal{A}_{\mathsf{bu}}) &\leq \operatorname{Adv}_{\mathsf{HSR}}^{\operatorname{EUF}\operatorname{-NMA}}(\mathcal{A}_{\mathsf{nma}}) + \operatorname{Adv}_{\mathsf{T}}^{\operatorname{CR}}(\mathcal{A}_{\mathsf{cr}}) + \operatorname{Adv}_{\mathsf{PRF}}^{q\operatorname{PRF}}(\mathcal{A}_{\mathsf{prf}}) \\ &+ \operatorname{Adv}_{\mathsf{PRF}'}^{q\operatorname{PRF}}(\mathcal{A}_{\mathsf{prf}}') + 8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^{2} \left(1 - \gamma \left(1 - \beta^{B}\right)\right) \\ &+ \frac{Bq_{\mathsf{F}}}{|\mathcal{R}|} + 2(q_{\mathsf{H}} + q_{\mathsf{F}}) \sqrt{\frac{B - 1}{|\mathcal{R}|}} + 2(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B - 1)}{|\mathcal{R}|}} \\ &+ \sqrt{6 \left(q_{\mathsf{H}} + 2q_{\mathsf{F}}\right)^{3} \left(\delta + 2 \left(1 - \gamma \left(1 - \beta^{B}\right)\right)\right)}, \end{aligned}$$

where T is (γ, β) -correct and δ -PS, and the running times of \mathcal{A}_{nma} , \mathcal{A}_{cr} , \mathcal{A}_{prf} , and \mathcal{A}'_{prf} are about that of \mathcal{A}_{bu} .

Proof. We use the sequence of games shown in Fig. 14. Note that this proof is almost identical to Theorem 1, with the only difference lying in G_5 . Here, the effect of introducing G_5 in Theorem 2 is the same as that of $G_{5.0}$ and $G_{5.1}$ in Theorem 1. However, the method for bounding $|\Pr[W_4] - \Pr[W_5]|$ differs due to the distinctions between the PO and sBU games.

GAME G_0 : This is the original sBU game: $\Pr[W_0] = \operatorname{Adv}_{\mathsf{DHSR}}^{\operatorname{sBU}}(\mathcal{A}_{\mathsf{bu}})$. GAME G_1 : We replace PRFs with random functions $\mathsf{RF}_{\mathsf{salt}}$ and $\mathsf{RF}_{\mathsf{inv}}$. From Lemma 7, we have $|\Pr[W_0] - \Pr[W_1]| \leq \operatorname{Adv}_{\mathsf{PRF}}^{q\operatorname{PRF}}(\mathcal{A}_{\mathsf{prf}}) + \operatorname{Adv}_{\mathsf{PRF}'}^{q\operatorname{PRF}}(\mathcal{A}'_{\mathsf{prf}})$. GAME G_2 : The oracles H and FORGE compute GetLogs. Since this is a concep-

tual change, we have $\Pr[W_1] = \Pr[W_2]$.

GAME G_3 : We check if a collision occurs among $\{r_i\}_{i \in [k]}$ in GetLogs. If a collision occurs, GetLogs returns \exists . From Lemma 8, we have

$$|\Pr[W_2] - \Pr[W_3]| \le (q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B-1)}{|\mathcal{R}|}}$$

GAME G_4 : FORGE returns \exists if $\mathsf{F}(vk, x_k) \neq \mathsf{H}(r_k, m)$ holds, where (r_k, x_k) is a signature generated by $\mathsf{GetLogs}(m)$. From Lemma 9, we have $|\Pr[W_3] - \Pr[W_4]| \leq 8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^2 (1 - \gamma (1 - \beta^B))$.

GAME G_5 : We add an additional condition to let $\mathbf{win} = \top$, that is, $(r, x) \neq (r_k, x_k)$ holds.

Lemma 16. We have

$$\left|\Pr[W_4] - \Pr[W_5]\right| \le \frac{Bq_{\mathsf{F}}}{|\mathcal{R}|}.$$

Proof. G_4 and G_5 differ only when the adversary submits a query (m, (r, x))such that $\mathsf{F}(vk, x) = \mathsf{H}(r, m)$, $(m, (r, x)) \in B_{\epsilon}$, and $(r, x) = (r_k, x_k)$ holds, because **win** becomes \top in G_4 but remains \bot in G_5 . Let **bad**_5 denote this event. We have $|\Pr[W_4] - \Pr[W_5]| \leq \Pr[\mathbf{bad}_5]$. Since $(m, (r_k, x_k)) \in B_{\epsilon}$, the adversary cannot obtain $r_k := \mathsf{RF}_{\mathsf{salt}}(m, k)$ from the queries to B_{ϵ} SIGN. Therefore, the adversary needs to guess $r = r_k$ without knowing r_k . As shown in Lemma 10, $\max_{r \in \mathcal{R}} \Pr[r_k := \mathsf{RF}_{\mathsf{salt}}(m, k) : r_k = r|\mathsf{H}| \leq \frac{B}{|\mathcal{R}|}$ holds. Since the adversary makes q_{F} queries to FORGE, $\Pr[\mathbf{bad}_5] \leq \frac{Bq_{\mathsf{F}}}{|\mathcal{R}|}$ holds.

23

\mathbf{G}	AMES G_0 - G_{11}	B_{ϵ}	SIGN(m)
1	$RF_{H} \leftarrow_{\$} Func(\mathcal{R} \times \mathcal{M}, \mathcal{Y})$	21	if GetLogs $(m) = \exists$ then $//G_3 - G_8$
2	$RF_{salt} \leftarrow_{\$} Func(\mathcal{M} \times [B+1], \mathcal{R}) //G_1 - G_{11}$	22	$ return \perp$ // G_3 - G_8
3	$RF_{inv} \leftarrow_{\$} Func(\mathcal{M} \times [B], \mathcal{R}') // G_1 \text{-} G_{10.0}$	23	$(r_k, y_k, x_k) \coloneqq GetLogs(m)$
4	$RF'_{H} \leftarrow_{\$} Func(\mathcal{M} \times [B], \mathcal{Y}) \qquad //G_7 \text{-} G_{10.0}$		$//G_0 - G_5 \cdot G_8 - G_{11}$
5	$RF_{sd} \leftarrow_{\$} Func(\mathcal{M}, \mathcal{R}'') \qquad //G_{10.1} \text{-} G_{11}$	24	$\{(r_i, y_i, x_i)\}_{i \in [k]} \coloneqq \operatorname{GetLogs}(m) //G_6 - G_7$
6	$B_{\epsilon} \leftarrow Func_{\mathcal{M} \times (\mathcal{R} \times \mathcal{X}), \{0,1\}}(Ber_{\epsilon})$	25	If $x_k = \perp \lor (m, (r_k, x_k)) \in B_{\epsilon}$ then
7	$\mathbf{win}\coloneqq ot$	26	$ $ return \perp
8	$(vk, sk) \leftarrow Gen(1^{\lambda})$	21	(r_k, x_k)
9	$(s, s') \leftarrow_{\$} \mathcal{K} \times \mathcal{K} $ // G_0	Fo	$\operatorname{DRGE}(m,(r,x))$
10	run $\mathcal{A}_{i}^{ H\rangle, B_{\epsilon}\mathrm{Sign}\rangle,\mathrm{Forge}}(vk)$	28	if $GetLogs(m) = \exists$ then $//G_3-G_8$
11	return win	29	$ return \perp$ $//G_3-G_8$
	<u>`</u>	30	$(r_k, y_k, x_k) \coloneqq GetLogs(m)$
Η(r,m)		$//G_2 - G_5 \cdot G_8 - G_{11}$
12	if $GetLogs(m) = \exists$ then $//G_3-G_8$	31	$\{(r_i, y_i, x_i)\}_{i \in [k]} \coloneqq GetLogs(m) //G_6 - G_7$
13	$ $ return \perp // G_3 - G_8	32	if $F(vk, x_k) \neq H(r_k, m)$ then $//G_4-G_{11}$
14	$(r_k, y_k, x_k) \coloneqq GetLogs(m)$	33	if $E(wk, m) = H(m, m) \land (m, (m, m)) \subset B$
15	$//G_2 - G_5 \cdot G_8 - G_{11}$ if $r = r_1$ then $//G_2 - G_7 \cdot G_8 - G_{11}$	34	then $\Pi \cap (v_{k}, x) = \Pi(r, m) \land (m, (r, x)) \in D_{\epsilon}$
10	$17 = 7_k$ then $7762-65 \cdot 68-611$	35	$ $ win = \top // G_0 - G_4
17	$\{(r_i, u_i, x_i)\}_{i \in [h]} \coloneqq \operatorname{GetLogs}(m) //Ge-G_7$	36	if $(r, x) \neq (r_k, x_k)$ then $//G_5 - G_{10,1}$
18	if $\exists i, r = r_i$ then $//G_6-G_7$	37	$ win = \top$ //G ₅ -G _{10.1}
19	roturn u	38	if $r \neq r_h$ then $1/G_{11}$
	10000007		
20	return $RF_{H}(r,m)$	39	$ \mathbf{win} = \top \qquad //G_{11}$
20 Ge	return $RF_{H}(r,m)$ tLogs(m) for G_0	39 Ge	$ \mathbf{win} = \top \qquad //G_{11}$ ttLogs(m) for G_1 - $G_{10,0}$
20 Ge	return $RF_{H}(r,m)$ $\frac{tLogs(m) \text{ for } G_0}{k \coloneqq 0}$	39 Ge	$ \mathbf{win} = \top \qquad //G_{11}$ $tLogs(m) \text{ for } G_1 - G_{10.0}$ $k \coloneqq 0$
20 Ge 40 41	return $RF_{H}(r,m)$ $\frac{tLogs(m) \text{ for } G_0}{k \coloneqq 0}$ repeat	39 Ge 52 53	$ \mathbf{win} = \top \qquad //G_{11}$ $\frac{tLogs(m) \text{ for } G_1 - G_{10.0}}{k \coloneqq 0}$ repeat
20 Ge 40 41 42	return $RF_{H}(r, m)$ $\frac{tLogs(m) \text{ for } G_0}{k \coloneqq 0}$ repeat $ k \coloneqq k + 1$	39 Ge 52 53 54	$ \mathbf{win} = \top \qquad //G_{11}$ $\frac{tLogs(m) \text{ for } G_1 - G_{10.0}}{k \coloneqq 0}$ $\mathbf{k} \coloneqq k + 1$
20 Ge 40 41 42 43	return $RF_{H}(r, m)$ $t \frac{Logs(m) \text{ for } G_0}{k \coloneqq 0}$ repeat $k \coloneqq k + 1$ $r_k \coloneqq PRF(s, (m, k))$	39 Ge 52 53 54 55	$ \mathbf{win} = \top \qquad //G_{11}$ $\frac{tLogs(m) \text{ for } G_{1}-G_{10.0}}{k \coloneqq 0}$ $\mathbf{k} \coloneqq k + 1$ $ k \coloneqq k + 1$ $r_{k} \coloneqq RF_{salt}(m, k) \qquad //G_{1}-G_{8}$
20 Ge 40 41 42 43 44	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $k \coloneqq k + 1$ $r_k \coloneqq RF(s, (m, k))$ $y_k \coloneqq RF_{H}(r_k, m)$	39 Ge 52 53 54 55 56	$ \begin{vmatrix} \mathbf{w} \mathbf{n} &= \top & //G_{11} \\ \hline \mathbf{tLogs}(m) \text{ for } G_1 \text{-} G_{10.0} \\ \hline \mathbf{k} &\coloneqq 0 \\ \hline \mathbf{repeat} \\ \begin{vmatrix} k &\coloneqq k+1 \\ r_k &\coloneqq RF_{sat}(m,k) & //G_1 \text{-} G_8 \\ y_k &\coloneqq RF_{H}(r_k,m) & //G_1 \text{-} G_2 \end{vmatrix} $
20 Ge 40 41 42 43 44 45	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $k \coloneqq k+1$ $r_k \coloneqq RFF(s, (m, k))$ $y_k \coloneqq RF_{H}(r_k, m)$ $x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$	39 52 53 54 55 56 57	$ \begin{vmatrix} \mathbf{w} & \mathbf{n} = \top & //G_{11} \\ \hline \mathbf{tLogs}(m) & \text{for } G_1 - G_{10.0} \\ \hline \mathbf{k} &\coloneqq 0 \\ \hline \mathbf{repeat} \\ \begin{vmatrix} k &\coloneqq k+1 \\ r_k &\coloneqq RF_{salt}(m,k) & //G_1 - G_8 \\ y_k &\coloneqq RF_{H}(r_k,m) & //G_1 - G_2 \\ y_k &\coloneqq RF_{H}(m,k) & //G_7 - G_{10.0} \\ \end{vmatrix} $
20 40 41 42 43 44 45 46	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $k \coloneqq k+1$ $r_k \coloneqq PRF(s, (m, k))$ $y_k \coloneqq RF_{H}(r_k, m)$ $x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$ until $x_k \neq \bot \lor k \ge B$	39 52 53 54 55 56 57 58	$ \begin{vmatrix} & & & & & & & & & $
20 Ge 40 41 42 43 44 45 46 47	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $k \coloneqq k+1$ $r_k \coloneqq PRF(s, (m, k))$ $y_k \coloneqq RF_{H}(r_k, m)$ $x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$ until $x_k \neq \bot \lor k \ge B$ return (r_k, y_k, x_k)	39 52 53 54 55 56 57 58 59	$ \begin{vmatrix} \mathbf{w} \mathbf{n} = \top & //G_{11} \\ \hline \mathbf{tLogs}(m) \text{ for } G_{1}\text{-}G_{10.0} \\ \hline \mathbf{k} \coloneqq 0 \\ \hline \mathbf{repeat} \\ \begin{vmatrix} k \coloneqq k+1 \\ r_k \coloneqq RF_{salt}(m,k) & //G_{1}\text{-}G_{2} \\ y_k \coloneqq RF_{H}(r_k,m) & //G_{1}\text{-}G_{2} \\ y_k \coloneqq RF_{H}(m,k) & //G_{7}\text{-}G_{10.0} \\ x_k \coloneqq Inv(sk, y_k; RF_{inv}(m,k)) \\ until x_k \neq \bot \lor k \ge B \\ \end{vmatrix} $
20 Ge 40 41 42 43 44 45 46 47	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $k \coloneqq k + 1$ $r_k \coloneqq PRF(s, (m, k))$ $y_k \coloneqq RF_{H}(r_k, m)$ $x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$ until $x_k \neq \perp \lor k \geq B$ return (r_k, y_k, x_k)	39 52 53 54 55 56 57 58 59 60	$ \begin{vmatrix} \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \hline \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} \\ \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} \\$
20 Ge 40 41 42 43 44 45 46 47 Ge	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $\begin{vmatrix} k \coloneqq k+1 \\ r_k \coloneqq PRF(s, (m, k)) \\ y_k \coloneqq RF_{H}(r_k, m) \\ x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$ until $x_k \neq \bot \lor k \ge B$ return (r_k, y_k, x_k) $tLogs(m)$ for $G_{10.1}$ - G_{11}	39 Ge 52 53 54 55 56 57 58 59 60 61	$ \begin{vmatrix} \mathbf{k} &:= \mathbf{k} \\ \mathbf{k} \\ \mathbf{k} &:= \mathbf{k} \\ \mathbf{k}$
$ \begin{array}{c} 20 \\ \hline Ge \\ 40 \\ 41 \\ 42 \\ 43 \\ 44 \\ 45 \\ 46 \\ 47 \\ \hline Ge \\ 48 \\ \hline \end{array} $	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $\begin{vmatrix} k \coloneqq k+1 \\ r_k \coloneqq PRF(s, (m, k)) \\ y_k \coloneqq RF_{H}(r_k, m) \\ x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$ until $x_k \neq \perp \lor k \ge B$ return (r_k, y_k, x_k) $tLogs(m)$ for $G_{10.1}$ - G_{11} $r_k \coloneqq RF_{salt}(m, B + 1)$	Ge 52 53 54 55 56 57 58 59 60 61 62 62	$ win = \top //G_{11}$ $\frac{ win = \top //G_{11}}{tLogs(m) \text{ for } G_1 - G_{10.0}}$ $k \coloneqq 0$ repeat $ k \coloneqq k + 1$ $ r_k \coloneqq RF_{salt}(m, k) //G_1 - G_2$ $y_k \coloneqq RF_{H}(r_k, m) //G_1 - G_2$ $y_k \coloneqq RF_{H}(m, k) //G_7 - G_{10.0}$ $x_k \coloneqq Inv(sk, y_k; RF_{inv}(m, k))$ until $x_k \neq \bot \lor k \ge B$ $r_k \coloneqq RF_{salt}(m, B + 1) //G_1 - G_3$ $ return \exists //G_3 - G_8$ $ return \exists //G_3 - G_8$
20 Ge 40 41 42 43 44 45 46 47 Ge 48 49	return $RF_{H}(r, m)$ $tLogs(m)$ for G_0 $k \coloneqq 0$ repeat $k \coloneqq k + 1$ $r_k \coloneqq PRF(s, (m, k))$ $y_k \coloneqq RF_{H}(r_k, m)$ $x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$ until $x_k \neq \perp \lor k \ge B$ return (r_k, y_k, x_k) $tLogs(m)$ for $G_{10.1}$ - G_{11} $\overline{r_k} \coloneqq RF_{salt}(m, B + 1)$ $x_k \coloneqq SampDom(vk; RF_{sd}(m))$	Ge 52 53 54 55 56 57 58 59 60 61 62 63 64	$ win = \top //G_{11}$ $\frac{tLogs(m) \text{ for } G_1 - G_{10.0}}{k := 0}$ $repeat$ $ k := k + 1$ $ r_k := RF_{salt}(m, k) //G_1 - G_2$ $y_k := RF_H(r_k, m) //G_1 - G_2$ $y_k := RF_H(m, k) //G_7 - G_{10.0}$ $x_k := \lnv(sk, y_k; RF_{inv}(m, k))$ until $x_k \neq \perp \lor k \ge B$ $r_k := RF_{salt}(m, B + 1) //G_1 - G_3$ $ return \exists //G_3 - G_8$ $ return \exists //G_3 - G_8$ $ return f(r_k, y_k, x_k) //G_1 - G_5 \cdot G_8 - G_1 - G_1$
$ \begin{array}{c} 20 \\ \hline Ge \\ 40 \\ 41 \\ 42 \\ 43 \\ 44 \\ 45 \\ 46 \\ 47 \\ \hline Ge \\ 48 \\ 49 \\ 50 \\ 51 \\ \end{array} $	return $\mathbb{R}_{H}(r, m)$ $\overline{tLogs(m) \text{ for } G_0}$ $k \coloneqq 0$ repeat $ k \coloneqq k + 1$ $ r_k \coloneqq PRF(s, (m, k))$ $ y_k \coloneqq RF_{H}(r_k, m)$ $ x_k \coloneqq Inv(sk, y_k; PRF'(s', (m, k)))$ $until x_k \neq \perp \lor k \ge B$ $return (r_k, y_k, x_k)$ $\overline{tLogs(m) \text{ for } G_{10.1} - G_{11}}$ $\overline{r_k} \coloneqq RF_{salt}(m, B + 1)$ $x_k \coloneqq SampDom(vk; RF_{sd}(m))$ $y_k \coloneqq F(vk, x_k)$ $return (r_k, y_k, r_k)$	39 <u>Ge</u> 52 53 54 55 56 57 58 59 60 61 62 63 64	$ \begin{array}{ l l l l l l l l l $

Fig. 14: Games for EUF-NMA \Rightarrow sBU

- GAME G_6 : GetLogs outputs $\{(r_i, y_i, x_i)\}_{i \in [k]}$ generated during loop iteration instead of outputting the final result, and $\mathsf{H}(r, m)$ outputs y_i if there exists r_i such that $r = r_i$. This modification does not affect the adversary's view, and we have $\Pr[W_5] = \Pr[W_6]$.
- GAME G_7 : Instead of $\mathsf{RF}_{\mathsf{H}}(r_k, m)$, $\mathsf{RF}'_{\mathsf{H}}(m, k)$ generates y_k in GetLogs. Since there are no collisions among $\{r_i\}_{i \in [k]}$, the adversary's view does not change; therefore, we have $\Pr[W_6] = \Pr[W_7]$.

GAME G_8 : GetLogs only outputs the final result after retries. From Lemma 11, we have

$$|\Pr[W_7] - \Pr[W_8]| \le 2(q_{\mathsf{H}} + q_{\mathsf{F}}) \sqrt{\frac{B-1}{|\mathcal{R}|}}.$$

GAME G_9 : GetLogs does not check collisions among $\{r_i\}_{i \in [k]}$. From Lemma 8, we have

$$|\Pr[W_8] - \Pr[W_9]| \le (q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B-1)}{|\mathcal{R}|}}.$$

GAME $G_{10.0}$: In GetLogs, we change the salt generation from $r_k \coloneqq \mathsf{RF}_{\mathsf{salt}}(m, k)$ to $r_k \coloneqq \mathsf{RF}_{\mathsf{salt}}(m, B + 1)$ for k such that $x_k \neq \bot$. Since the view of the adversary does not change, we have $\Pr[W_9] = \Pr[W_{10}]$.

GAME $G_{10.1}$: We simulate GetLogs using SampDom. From Lemma 13, we have

$$|\Pr[W_9] - \Pr[W_{10}]| \le \sqrt{6(q_{\mathsf{H}} + 2q_{\mathsf{F}})^3(\delta + 2(1 - \gamma(1 - \beta^B)))}.$$

GAME G_{11} : We change the condition of $\mathbf{win} = \top$ from $(r, x) \neq (r_k, x_k)$ to $r \neq r_k$ in FORGE. From Lemma 14, we have $|\Pr[W_{10}] - \Pr[W_{11}]| \leq \operatorname{Adv}_{\mathsf{T}}^{\operatorname{CR}}(\mathcal{A}_{\mathsf{cr}})$.

The EUF-NMA adversary \mathcal{A}_{nma} against $HSR[T, \hat{H}]$ with oracle access to \hat{H} can simulate G_{11} since it can simulate all the oracles by using SampDom. Similar to Lemma 15, the EUF-NMA adversary outputs $(m^*, (r^*, x^*))$ that was queried in FORGE and caused win to become \top . Since $F(vk, x^*) = \hat{H}(r^*, m^*)$ holds, the EUF-NMA adversary can win its game if win $= \top$ in G_{11} . Hence, there exists an EUF-NMA adversary \mathcal{A}_{nma} such that $\Pr[W_{11}] \leq \operatorname{Adv}_{HSR}^{EUF-NMA}(\mathcal{A}_{nma})$. Since $HSR[T, \hat{H}]$ is equivalent to HSR[T, H], this completes the proof. \Box

The BU security does not require the collision-resistance of T; therefore, we can eliminate the CR assumption in the BU security as follows:

Corollary 2 (EUF-NMA + **qPRF** \Rightarrow **BU).** For any quantum BU adversary \mathcal{A}_{bu} of DHSR_B[T, H, PRF, PRF'] issuing at most q_{H} quantum queries to H, q_{S} quantum queries to SIGN, and q_{F} classical queries to FORGE, there exist an EUF-NMA adversary \mathcal{A}_{nma} of HSR[T, H] and qPRF adversaries \mathcal{A}_{prf} of PRF and \mathcal{A}'_{prf} of PRF' issuing at most Bq_{S} queries such that

$$\begin{aligned} \operatorname{Adv}_{\mathsf{DHSR}}^{\mathrm{BU}}(\mathcal{A}_{\mathsf{bu}}) &\leq \operatorname{Adv}_{\mathsf{HSR}}^{\mathrm{EUF}\operatorname{-NMA}}(\mathcal{A}_{\mathsf{nma}}) + \operatorname{Adv}_{\mathsf{PRF}}^{q\operatorname{PRF}}(\mathcal{A}_{\mathsf{prf}}) + \operatorname{Adv}_{\mathsf{PRF'}}^{q\operatorname{PRF}}(\mathcal{A'}_{\mathsf{prf}}) \\ &+ 8(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}} + 1)^{2} \left(1 - \gamma \left(1 - \beta^{B}\right)\right) \\ &+ \frac{Bq_{\mathsf{F}}}{|\mathcal{R}|} + 2(q_{\mathsf{H}} + q_{\mathsf{F}}) \sqrt{\frac{B - 1}{|\mathcal{R}|}} + 2(q_{\mathsf{H}} + q_{\mathsf{S}} + q_{\mathsf{F}}) \sqrt{\frac{2B(B - 1)}{|\mathcal{R}|}} \\ &+ \sqrt{6 \left(q_{\mathsf{H}} + 2q_{\mathsf{F}}\right)^{3} \left(\delta + 2 \left(1 - \gamma \left(1 - \beta^{B}\right)\right)\right)}, \end{aligned}$$

where T is (γ, β) -correct and δ -PS, and the running times of \mathcal{A}_{nma} , \mathcal{A}_{prf} , and \mathcal{A}'_{prf} are about that of \mathcal{A}_{po} .

Proof. In Theorem 2, we can change the winning condition in G_5 from $(r, x) \neq (r_k, x_k)$ to $r \neq r_k$ (see Line 36 in Fig. 14). Therefore, we can remove G_{11} and the collision-resistance of T is unnecessary for the BU security.

4.3 (Strong) Existential Unforgeability

We can prove $\mathsf{DHSR}_B[\mathsf{T},\mathsf{H},\mathsf{PRF},\mathsf{PRF}']$ is sEUF-CMA-secure.

Theorem 3 (EUF-NMA + **PS** + **SPR** + **PRF** \Rightarrow **sEUF-CMA).** For any quantum sEUF-CMA adversary \mathcal{A}_{cma} of DHSR_B[T, H, PRF, PRF'] issuing at most q_H quantum queries to H and q_S quantum queries to SIGN, there exist an EUF-NMA adversary \mathcal{A}_{nma} of HSR[T, H], a PS adversary \mathcal{A}_{ps} issuing at most q_S queries, SPR adversary \mathcal{A}_{spr} of T, and PRF adversaries \mathcal{A}_{prf} of PRF and \mathcal{A}'_{orf} of PRF' issuing at most Bq_S queries such that

$$\begin{aligned} \operatorname{Adv}_{\mathsf{DHSR}}^{\operatorname{SEUF-CMA}}(\mathcal{A}_{\mathsf{cma}}) &\leq \operatorname{Adv}_{\mathsf{HSR}}^{\operatorname{EUF-NMA}}(\mathcal{A}_{\mathsf{nma}}) + \operatorname{Adv}_{\mathsf{T}}^{\operatorname{PS}}(\mathcal{A}_{\mathsf{ps}}) + q_{\mathsf{S}}\operatorname{Adv}_{\mathsf{T}}^{\operatorname{SPR}}(\mathcal{A}_{\mathsf{spr}}) \\ &+ \operatorname{Adv}_{\mathsf{PRF}}^{\operatorname{PRF}}(\mathcal{A}_{\mathsf{prf}}) + \operatorname{Adv}_{\mathsf{PRF}'}^{\operatorname{PRF}}(\mathcal{A}_{\mathsf{'prf}}') + q_{\mathsf{S}}\left(1 - \gamma\left(1 - \beta^{B}\right)\right) \\ &+ \frac{3}{2}Bq_{\mathsf{S}}\sqrt{\frac{q_{\mathsf{H}} + Bq_{\mathsf{S}} + 1}{|\mathcal{R}|}} + 2(q_{\mathsf{H}} + 1)\sqrt{\frac{B - 1}{|\mathcal{R}|}}, \end{aligned}$$

where T is (γ, β) -correct and the running times of \mathcal{A}_{nma} , \mathcal{A}_{ps} , \mathcal{A}_{spr} , \mathcal{A}_{prf} , and \mathcal{A}'_{prf} are about that of \mathcal{A}_{cma} .

Proof. We use the sequence of games shown in Fig. 15.

GAME G_0 : This is the original SEUF-CMA game: $\Pr[W_0] = \operatorname{Adv}_{\mathsf{DHSR}}^{\operatorname{SEUF-CMA}}(\mathcal{A}_{\mathsf{cma}})$. GAME G_1 : We replace PRF and PRF' with random functions $\mathsf{RF}_{\mathsf{salt}}$ and $\mathsf{RF}_{\mathsf{inv}}$, respectively.

Lemma 17. There exist PRF adversaries \mathcal{A}_{prf} of PRF and \mathcal{A}'_{prf} of PRF' such that

$$|\Pr[W_0] - \Pr[W_1]| \le \mathrm{Adv}_{\mathsf{PRF}}^{\mathsf{PRF}}(\mathcal{A}_{\mathsf{prf}}) + \mathrm{Adv}_{\mathsf{PRF}'}^{\mathsf{PRF}}(\mathcal{A}'_{\mathsf{prf}}).$$

Proof. As in Lemma 7, we can replace $\mathsf{PRF}(s, \cdot)$ and $\mathsf{PRF}'(s', \cdot)$ separately. Since the PRFs are classically executed, the (classical) PRF adversaries $\mathcal{A}_{\mathsf{prf}}$ and $\mathcal{A}'_{\mathsf{prf}}$ can simulate the outputs of PRF and PRF' using the outputs of their oracles. Thus, the advantage gap due to the above transformation can be bounded by the PRF advantages.

GAME G_2 : Let σ be a database of signatures indexed by messages, where each signature $\sigma[m]$ corresponds to a message m used in generating the signature. The signing oracle SIGN returns $\sigma[m]$ if m has been queried previously ($\sigma[m] \neq \emptyset$). Since this is a conceptual change, we have $\Pr[W_1] = \Pr[W_2]$.

Storing σ ensures that, in the subsequent game hops, when randomness is generated without using the random function, the same randomness is used for the same m.

GAME G_3 : The signing oracle SIGN uniformly chooses y_k and reprograms $\mathsf{H} := \mathsf{H}^{(r_k,m)\mapsto y_k}$ for the chosen y_k . This step is crucial in the simulation of the signing oracle, as it requires generating y_k independently of (r_k,m) and treating y_k as the output of H when (r_k,m) is given as input.

GAMES G_0 - G_6	$\frac{H(r,m)}{r}$
$2 \text{ BE } = \bigoplus_{n \in \mathbb{N}} \operatorname{Eunc}(M \times [B+1] \mathcal{R}) //G_{1-}G_{2-1}$	13 return $RFH(r, m)$
	SIGN(m)
$\begin{array}{c} \mathbf{J} \mathbf{H} $	$14 \text{ if } \sigma[m] \neq \emptyset \text{ then}$ //G ₂ -G ₆
$5 (vk, sk) \leftarrow \text{Gen}(1^{\lambda})$	15 return $\sigma[m]$ // G_2 - G_6
$\begin{array}{c} 6 (\mathbf{s}, \mathbf{s}') \leftarrow \mathbf{c} \mathcal{K} \times \mathcal{K} \\ 6 (\mathbf{s}, \mathbf{s}') \leftarrow \mathbf{c} \mathcal{K} \times \mathcal{K} \\ \end{array} $	16 $(r_k, y_k, x_k) \coloneqq GetLogs(m)$
$= (m^* (m^* m^*)) \langle A^{\text{SIGN}, \mathbf{H} \rangle}(mk)$	17 if $x_k = \bot$ then
$\begin{array}{c} r & (m, (r, x)) \leftarrow \mathcal{A}_{cma} & (vk) \\ s & if & (m^* & (r^* & r^*)) \in O & then \\ \end{array} $	$18 \sigma[m] \coloneqq \bot \qquad //G_2 - G_6$
$9 return //G_0-G_{5,1}$	19 return \perp
10 if $(m^*, r^*) \in Q$ then $//G_6$	$20 \ \sigma[m] \coloneqq (r_k, x_k) \qquad //G_2 - G_6$
11 return \perp // G_6	21 $\mathcal{Q} := \mathcal{Q} \cup \{(m, (r_k, x_k))\}$ // G_2 - $G_{5.1}$
12 return $[\![F(vk, x^*) = H(r^*, m^*)]\!]$	22 $\mathfrak{G} := \mathfrak{G} \mathfrak{G} \{(m, r_k)\}$ 77 \mathfrak{G}_6 23 return (r_k, x_k)
$\frac{\text{GetLogs}(m) \text{ for } G_0}{}$	$\operatorname{GetLogs}(m)$ for G_1 - $G_{5.0}$
24 $k := 0$	37 $k := 0$
25 repeat	38 repeat
$\begin{vmatrix} 26 \\ k \coloneqq k+1 \\ DDF((k+1)) \end{vmatrix}$	$39 k \coloneqq k+1$
$\begin{array}{c c} 27 & r_k \coloneqq PKF(s, (m, k)) \\ \hline \end{array}$	40 $r_k := RF_{salt}(m, k)$
$\begin{array}{c c} 28 & y_k \coloneqq RF_{H}(r_k, m) \\ & \downarrow & (l & DDF((l & (l + 1))) \end{array} \end{array}$	$\begin{array}{c c} 41 & y_k \coloneqq RF_{H}(r_k, m) & //G_1 \text{-} G_2 \\ 42 & y_k \leftarrow \mathcal{Y} & //G_1 \text{-} G_2 \end{array}$
29 $x_k \coloneqq \operatorname{Inv}(sk, y_k; PRF(s, (m, k)))$	42 $g_k \leftarrow g_k = 1$ 43 $r_k \leftarrow \ln (sk u_k \cdot RE_k (m, k))$
30 until $x_k \neq \perp \lor k \geq B$	$43 \mathbf{x}_k := \min(3k, y_k, \min(m, n))$
31 return (r_k, y_k, x_k)	$44 RF_{H} \coloneqq RF_{H}^* = \% / \mathcal{G}_3$
Get $logs(m)$ for $G_{r,1}$ -G _c	45 until $x_k \neq \perp \forall k \geq B$
$\frac{\operatorname{Geteogs}(m)\operatorname{for}\operatorname{G}_{5,1}-\operatorname{G}_{6}}{\operatorname{Geteogs}(m)}$	46 $r_k \coloneqq KF_{salt}(m, B+1)$ // G _{5.0}
$\begin{array}{ccc} 32 & T_k &= NF_{salt}(m, D+1) \\ ss & x_k \leftarrow SamnDom(wk) \end{array}$	47 $RF_{H} \coloneqq RF_{H}^{(r_k, m) \mapsto g_k}$ // G_4 - $G_{5.0}$
$\begin{array}{l} 33 x_k \leftarrow Sampboli(v_k) \\ 34 y_k \coloneqq F(v_k \mid x_k) \end{array}$	48 return (r_k, y_k, x_k)
$ \begin{array}{c} \mathbf{y}_{\kappa} := \mathbf{r} (cn, w_{\kappa}) \\ \mathbf{p}_{\kappa} := \mathbf{p}_{\kappa} (r_{\kappa}, m) \mapsto y_{\kappa} \end{array} $	
$\begin{array}{c} 35 K\Gamma_{H} \coloneqq K\Gamma_{H}^{*} \coloneqq K\Gamma_{H}^{*} & \mathcal{K} \\ a_{H} potump \left(m u_{H} m_{H} \right) \end{array}$	
$36 \text{ return } (r_k, y_k, x_k)$	

The Security of Hash-and-Sign with Retry against Superposition Attacks 27

Fig. 15: Games for EUF-NMA \Rightarrow sEUF-CMA

Lemma 18. We have

$$|\Pr[W_2] - \Pr[W_3]| \le \frac{3}{2} Bq_{\mathsf{S}} \sqrt{\frac{q_{\mathsf{H}} + Bq_{\mathsf{S}} + 1}{|\mathcal{R}|}}$$

Proof. The AR adversary \mathcal{B}_{ar} (see Fig. 10) can simulate G_2 and G_3 , where RF_H is a random function reprogrammed in the AR game. To simulate GetLogs, given m, \mathcal{B}_{ar} submits m to its oracle REPRO and obtains random $r_k \leftarrow_{\$} \mathcal{R}$ until $(x_k \neq \bot) \lor (k \geq B)$. Note that $r_k = \mathsf{RF}_{\mathsf{salt}}(m, k)$ is uniformly distributed over \mathcal{R} in both games. Note also that \mathcal{B}_{ar} can return $\sigma[m]$ without using REPRO if the same m is queried again. Hence, \mathcal{B}_{ar} can use the oracle's output as the salts in GetLogs. If \mathcal{B}_{ar} plays AR₀, RF_H is not reprogrammed; therefore, it can simulate G_2 ; otherwise RF_H is reprogrammed for random y and \mathcal{B}_{ar} can simulates G_3 . Therefore, there exists an AR adversary \mathcal{B}_{ar} such that $|\Pr[W_2] - \Pr[W_3]| \leq \mathrm{Adv}_{\mathsf{AR}}^{\mathsf{AR}}(\mathcal{B}_{\mathsf{ar}})$. Since RF_H is reprogrammed at most Bq_{S} times, we have the bound in this lemma from Lemma 2.

GAME G_4 : We cancel the reprogramming executed for intermediate results, and RF_H is reprogrammed only for the final (r_k, y_k) (see Line 47). By eliminating

the need to simulate intermediate results, we are now ready to simulate the signing oracle.

Lemma 19. We have

$$|\Pr[W_3] - \Pr[W_4]| \le 2(q_{\mathsf{H}} + 1)\sqrt{\frac{B-1}{|\mathcal{R}|}}.$$

Proof. The reprogrammings during retries are canceled in G_4 . The random function H in G_3 is reprogrammed for each retry attempt, while H in G_4 is reprogrammed only for the final result; therefore, differences of these random functions are all in $S := \{(r,m) | \exists i \in [k-1], r = r_i\}$ for $\{(r_i, y_i, x_i)\}_{i \in [k]}$ generated inside GetLogs(m). Since the random function is reprogrammed, we use Lemma 4 (O2H with adaptive reprogramming), where we set O as SIGN. Let \mathcal{B}_{o2h} be an adversary who runs \mathcal{A}_{cma} in G_4 and finds an element in S. Choosing $i \leftarrow_{\$} [q_{\mathsf{H}}]$, \mathcal{B}_{o2h} measures the query input register of \mathcal{A}_{cma} and returns the result. \mathcal{B}_{o2h} has no information on S and $\frac{|S|}{|\mathcal{R} \times \mathcal{M}|} \leq \frac{(B-1)|\mathcal{M}|}{|\mathcal{R}||\mathcal{M}|} = \frac{B-1}{|\mathcal{R}|}$ holds. From Lemma 4, we have this lemma.

GAME $G_{5.0}$: We modify GetLogs in two steps to make it simulatable. Firstly, the value $r_k := \mathsf{RF}_{\mathsf{salt}}(m,k)$ is generated by $\mathsf{RF}_{\mathsf{salt}}(m,B+1)$ in GetLogs. Since both $\mathsf{RF}_{\mathsf{salt}}(m,k)$ and $\mathsf{RF}_{\mathsf{salt}}(m,B+1)$ are uniformly distributed and the adversary can only access outputs of $\mathsf{RF}_{\mathsf{salt}}$ via SIGN, the adversary's view remains unchanged. Consequently, we have $\Pr[W_4] = \Pr[W_{5.0}]$. GAME $G_{5.1}$: Secondly, we simulate SIGN using SampDom.

Lemma 20. Suppose that T is (γ, β) -correct. There exists a PS adversary \mathcal{A}_{ps} of T such that

$$\left|\Pr[W_{5.0}] - \Pr[W_{5.1}]\right| \le \operatorname{Adv}_{\mathsf{T}}^{\operatorname{PS}}(\mathcal{A}_{\mathsf{ps}}) + q_{\mathsf{S}}\left(1 - \gamma\left(1 - \beta^B\right)\right).$$

Proof. We consider the simulation by the PS adversary \mathcal{A}_{ps} . For the signing query, \mathcal{A}_{ps} returns $r_k := \mathsf{RF}_{\mathsf{salt}}(m, B+1)$ and x_k , which is output by its oracle SAMPLE_b. If \mathcal{A}_{ps} plays PS₁, x_k is generated by SampDom, allowing us to simulate $G_{5.1}$. If \mathcal{A}_{ps} plays PS₀, we need to account for the possibility that the number of retries exceeds B or that inversion fails ($\mathsf{F}(vk, x_k) \neq y_k$), which we define as **bad**₅. When **bad**₅ does not occur, \mathcal{A}_{ps} simulates $G_{5.1}$. Since **bad**₅ happens with a probability of at most $q_{\mathsf{S}}(1-\gamma(1-\beta^B))$, we have this lemma.

GAME G_6 : We change the condition to output \perp from $(m^*, (r^*, x^*)) \in \mathcal{Q}$ to $(m^*, r^*) \in \mathcal{Q}$. As the condition $(m^*, (r^*, x^*)) \notin \mathcal{Q}$ allows for the possibility that H may be reprogrammed on (r^*, m^*) , the new condition $(m^*, r^*) \notin \mathcal{Q}$ eliminates that possibility. Then, the EUF-NMA adversary can win its game by submitting \mathcal{A}_{cma} 's output if \mathcal{A}_{cma} wins G_6 .

Lemma 21. There exists an SPR adversary \mathcal{A}_{spr} of T such that

$$|\Pr[W_{5.1}] - \Pr[W_6]| \le q_{\mathsf{S}} \mathrm{Adv}_{\mathsf{T}}^{\mathrm{SPR}}(\mathcal{A}_{\mathsf{spr}}).$$

The Security of Hash-and-Sign with Retry against Superposition Attacks 29

$\begin{array}{c} \underline{\mathcal{A}_{nma}^{\hat{H}}(vk)} \\ 1 \mathbf{RF_{salt}} \leftarrow_{\$} Func(\mathcal{M} \times [B+1], \mathcal{R}) \\ 2 \mathcal{Q} \coloneqq \emptyset \\ 3 (m^*, (r^*, x^*)) \leftarrow \mathcal{A}_{cma}^{\mathrm{SiGN}, H\rangle}(vk) \\ 4 \mathbf{if} (m^*, r^*) \in \mathcal{Q} \text{ then} \\ 5 \text{ return } \bot \\ 6 \mathrm{return} (m^*, (r^*, x^*)) \\ \underline{H}(r, m) \end{array}$	$\frac{\text{SIGN}(m)}{12 \text{ if } \sigma[m] \neq \emptyset \text{ then}}$ $13 \text{return } \sigma[m]$ $14 (r_k, y_k, x_k) \coloneqq \text{GetLogs}(m)$ $15 \text{ if } x_k = \bot \text{ then}$ $16 \sigma[m] \coloneqq \bot$ $17 \text{return } \bot$ $18 \sigma[m] \coloneqq (r_k, x_k)$ $19 \ \mathcal{Q} \coloneqq \mathcal{Q} \cup \{(m, r_k)\}$ $20 \text{ return } (r_k, x_k)$
7 if $\sigma[m] \neq \emptyset \land \sigma[m] \neq \bot$ then 8 $ (r_k, x_k) \coloneqq \sigma[m]$ 9 if $r = r_k$ then 10 $ $ return $F(vk, x_k)$ 11 return $\hat{H}(r, m)$	$\frac{\text{GetLogs}(m)}{21 r_k := RF_{salt}(m, B+1)}$ $22 x_k := \text{SampDom}(vk)$ $23 y_k := F(vk, x_k)$ $24 \text{return} (r_k, y_k, x_k)$

Fig. 16: Simulation of the modified sEUF-CMA game by EUF-NMA adversary

Proof. $G_{5.1}$ and G_6 differ only if the adversary submits $(m^*, (r^*, x^*))$ such that $(m^*, (r^*, x^*)) \notin \mathcal{Q}$ in $G_{5.1}, (m^*, r^*) \in \mathcal{Q}$ in G_6 , and $\mathsf{F}(vk, x^*) = \mathsf{H}(r^*, m^*)$ holds in both games. That is, $r^* = r_k$ and $x^* \neq x_k$, where (r_k, x_k) is generated by SIGN (m^*) ; therefore, $\mathsf{F}(vk, x^*) = \mathsf{F}(vk, x_k) = \mathsf{H}(r_k, m^*)$ holds. Let **bad**₆ be such an event and $|\Pr[W_{5.1}] - \Pr[W_6]| \leq \Pr[\mathsf{bad}_6]$ holds.

We show a bound on \mathbf{bad}_6 using the SPR game shown in Definition 7. Note that we assume that the distribution of the challenge \hat{x} follows the one of SampDom(vk) in the SPR game. The SPR adversary \mathcal{A}_{spr} simulates G_6 by setting its challenge \hat{x} as the output of SampDom in *i*-th query to SIGN, where $i \leftarrow_{\$} [q_{\mathsf{S}}]$. When \mathbf{bad}_6 occurs for $(m^*, (r^*, x^*))$ and m^* is *i*-th query, $\mathsf{F}(vk, x^*) = \mathsf{F}(vk, \hat{x}) = \mathsf{H}(r^*, m^*)$ holds. Hence, $\mathcal{A}_{\mathsf{spr}}$ can win the SPR game by submitting x^* as a second preimage of \hat{x} . Since $\mathcal{A}_{\mathsf{spr}}$ correctly guesses *i* with $\frac{1}{q_{\mathsf{S}}}$, $\Pr[\mathbf{bad}_6] \leq q_{\mathsf{S}} \mathrm{Adv}_{\mathsf{T}}^{\mathrm{SPR}}(\mathcal{A}_{\mathsf{spr}})$ holds.

Then, we can conclude this theorem by bounding $\Pr[W_6]$.

Lemma 22. There exists an EUF-NMA adversary \mathcal{A}_{nma} of $\mathsf{HSR}[\mathsf{T},\mathsf{H}]$ such that

$$\Pr[W_6] \le \mathrm{Adv}_{\mathsf{HSR}}^{\mathrm{EUF}-\mathrm{NMA}}(\mathcal{A}_{\mathsf{nma}}).$$

Proof. The EUF-NMA adversary \mathcal{A}_{nma} with oracle access to \hat{H} can simulate G_6 as in Fig. 16. Note that \mathcal{A}_{nma} sets $H = \hat{H}$ and outputs of H and \hat{H} differ for reprogrammed points. To simulate the reprogramming in the execution of SIGN, H(r,m) outputs $F(vk, x_k)$ for $(r_k, x_k) = \sigma[m]$ if $r = r_k$ holds (see Lines 7 to 10). When \mathcal{A}_{cma} wins the game by submitting $(m^*, (r^*, x^*)), H(r^*, m^*)$ is not reprogrammed from $(m^*, r^*) \notin \mathcal{Q}$. Therefore, $F(vk, x^*) = \hat{H}(r^*, m^*)$ holds and \mathcal{A}_{nma} can win the game by submitting $(m^*, (r^*, x^*))$.

The EUF-CMA security does not require second-preimage resistance of ${\sf T}$ as follows:

Corollary 3 (EUF-NMA + **PS** + **PRF** \Rightarrow **EUF-CMA).** For any quantum EUF-CMA adversary \mathcal{A}_{cma} of $\mathsf{DHSR}_B[\mathsf{T},\mathsf{H},\mathsf{PRF},\mathsf{PRF'}]$ issuing at most q_{H} quantum queries to H and q_{S} quantum queries to SIGN , there exist an EUF-NMA adversary \mathcal{A}_{nma} of $\mathsf{HSR}[\mathsf{T},\mathsf{H}]$, a PS adversary $\mathcal{A}_{\mathsf{ps}}$ issuing at most q_{S} queries, and PRF adversaries $\mathcal{A}_{\mathsf{prf}}$ of PRF and $\mathcal{A}'_{\mathsf{prf}}$ of PRF' issuing at most Bq_{S} queries such that

$$\begin{aligned} \operatorname{Adv}_{\mathsf{DHSR}}^{\mathrm{EUF-CMA}}(\mathcal{A}_{\mathsf{cma}}) &\leq \operatorname{Adv}_{\mathsf{HSR}}^{\mathrm{EUF-NMA}}(\mathcal{A}_{\mathsf{nma}}) + \operatorname{Adv}_{\mathsf{T}}^{\mathrm{PS}}(\mathcal{A}_{\mathsf{ps}}) + \operatorname{Adv}_{\mathsf{PRF}}^{\mathrm{PRF}}(\mathcal{A}_{\mathsf{prf}}) \\ &+ \operatorname{Adv}_{\mathsf{PRF}}^{\mathrm{PRF}}(\mathcal{A}_{\mathsf{prf}}') + q_{\mathsf{S}}\left(1 - \gamma\left(1 - \beta^{B}\right)\right) \\ &+ \frac{3}{2}Bq_{\mathsf{S}}\sqrt{\frac{q_{\mathsf{H}} + Bq_{\mathsf{S}} + 1}{|\mathcal{R}|}} + 2(q_{\mathsf{H}} + 1)\sqrt{\frac{B - 1}{|\mathcal{R}|}}, \end{aligned}$$

where T is (γ, β) -correct and the running times of \mathcal{A}_{nma} , \mathcal{A}_{ps} , \mathcal{A}_{prf} , and \mathcal{A}'_{prf} are about that of \mathcal{A}_{cma} .

Proof. Since \mathcal{Q} stores only messages, if $m^* \notin \mathcal{Q}$, then $\mathsf{RF}_{\mathsf{H}}(r^*, m^*)$ is not reprogrammed in the proof of Theorem 3. Therefore, we can skip the last game G_6 and the second-preimage resistance of T is not required for EUF-CMA security. \Box

Remark 2. We compare the result with the existing proof for the original HSwR by Kosuge and Xagawa [19]. First, we improve the tightness utilizing the derandomization with bounded loop. The security bound of the original HSwR shown in [19] is as follows.

$$\begin{split} \operatorname{Adv}_{\mathsf{HSwR}}^{(\mathrm{s})\operatorname{EUF-CMA}}(\mathcal{A}_{\mathsf{cma}}) &\leq \operatorname{Adv}_{\mathsf{HSwR}}^{\operatorname{EUF-NMA}}(\mathcal{A}_{\mathsf{nma}}) + \operatorname{Adv}_{\mathsf{T}}^{\operatorname{PS}}(\mathcal{A}_{\mathsf{ps}}) \\ &+ \frac{3}{2}q_{\mathsf{S}}'\sqrt{\frac{q_{\mathsf{H}} + q_{\mathsf{S}}' + 1}{|\mathcal{R}|}} + 2(q_{\mathsf{H}} + 2)\sqrt{\frac{q_{\mathsf{S}}' - q_{\mathsf{S}}}{|\mathcal{R}|}}, \end{split}$$

where q'_{S} is a bound on the total number of retries during q_{S} signing queries $(q'_{\mathsf{S}} \leq Bq_{\mathsf{S}} \text{ holds in DHSwBR})$. The last term is replaced with $2(q_{\mathsf{H}}+1)\sqrt{\frac{B-1}{|\mathcal{R}|}}$ in Theorem 3 and Corollary 3. However, as a drawback, the PRF advantage (due to the derandomization) and the probability that no valid signature is generated in signing queries (due to the bounded loop) are added to the security bound. These terms become negligible when well-evaluated PRFs are used and *B* is set appropriately based on the failing probability β .

Second, we relax the condition of T required for the sEUF-CMA security from *injection* to second-preimage-resistance, where this relaxation is also applied to the original HSwR.

References

 Alagic, G., Majenz, C., Russell, A., Song, F.: Quantum-access-secure message authentication via blind-unforgeability. In: Canteaut, A., Ishai, Y. (eds.) EURO-CRYPT 2020, Part III. LNCS, vol. 12107, pp. 788–817. Springer, Cham (May 2020). https://doi.org/10.1007/978-3-030-45727-3_27 2, 3, 7, 8 The Security of Hash-and-Sign with Retry against Superposition Attacks

- Ambainis, A., Hamburg, M., Unruh, D.: Quantum security proofs using semiclassical oracles. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part II. LNCS, vol. 11693, pp. 269–295. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26951-7_10 6, 13, 33, 34
- Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) ACM CCS 93. pp. 62–73. ACM Press (Nov 1993). https://doi.org/10.1145/168588.168596 2, 3
- Bellare, M., Rogaway, P.: The exact security of digital signatures: How to sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 399–416. Springer, Berlin, Heidelberg (May 1996). https://doi.org/10.1007/3-540-68339-9_34_2
- Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Lee, D.H., Wang, X. (eds.) ASI-ACRYPT 2011. LNCS, vol. 7073, pp. 41–69. Springer, Berlin, Heidelberg (Dec 2011). https://doi.org/10.1007/978-3-642-25385-0_3 2, 4
- Boneh, D., Zhandry, M.: Secure signatures and chosen ciphertext security in a quantum computing world. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 361–379. Springer, Berlin, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40084-1_21 2, 3, 4, 7, 13, 14
- Chatterjee, R., Chung, K.M., Liang, X., Malavolta, G.: A note on the post-quantum security of (ring) signatures. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) PKC 2022, Part II. LNCS, vol. 13178, pp. 407–436. Springer, Cham (Mar 2022). https://doi.org/10.1007/978-3-030-97131-1_14 3, 4
- Cogliati, B., Fouque, P.A., Goubin, L., Minaud, B.: New security proofs and techniques for hash-and-sign with retry signature schemes. Cryptology ePrint Archive, Report 2024/609 (2024), https://eprint.iacr.org/2024/609 3
- Devevey, J., Fallahpour, P., Passelègue, A., Stehlé, D.: A detailed analysis of Fiat-Shamir with aborts. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part V. LNCS, vol. 14085, pp. 327–357. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38554-4_11 6, 16
- Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Berlin, Heidelberg (Aug 1987). https://doi.org/10.1007/3-540-47721-7_12_2
- Furue, H., Ikematsu, Y., Hoshino, F., Takagi, T., Yasuda, K., Miyazawa, T., Saito, T., Nagai, A.: QR-UOV. Tech. rep., National Institute of Standards and Technology (2023), available at https://csrc.nist.gov/Projects/pqc-dig-sig/ round-1-additional-signatures 3
- Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008). https://doi.org/10.1145/1374376.1374407 2, 10
- Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. 17(2), 281–308 (1988). https://doi.org/10.1137/0217017, https://doi.org/10.1137/0217017 1
- Goubin, L., Cogliati, B., Faugère, J., Fouque, P., Larrieu, R., Macario-Rat, G., Minaud, B., Patarin, J.: PROV — PRovable unbalanced Oil and Vinegar. Tech. rep., National Institute of Standards and Technology (2023), available at https: //csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures 3

- 32 H. Kosuge and K. Xagawa
- Grilo, A.B., Hövelmanns, K., Hülsing, A., Majenz, C.: Tight adaptive reprogramming in the QROM. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part I. LNCS, vol. 13090, pp. 637–667. Springer, Cham (Dec 2021). https://doi.org/10.1007/978-3-030-92062-3 22 6, 12, 13
- Hosoyamada, A., Yasuda, K.: Building quantum-one-way functions from block ciphers: Davies-Meyer and Merkle-Damgård constructions. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part I. LNCS, vol. 11272, pp. 275–304. Springer, Cham (Dec 2018). https://doi.org/10.1007/978-3-030-03326-2_10_3
- Hülsing, A., Rijneveld, J., Song, F.: Mitigating multi-target attacks in hashbased signatures. In: Cheng, C.M., Chung, K.M., Persiano, G., Yang, B.Y. (eds.) PKC 2016, Part I. LNCS, vol. 9614, pp. 387–416. Springer, Berlin, Heidelberg (Mar 2016). https://doi.org/10.1007/978-3-662-49384-7_15 12
- Kiltz, E., Lyubashevsky, V., Schaffner, C.: A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 552–586. Springer, Cham (Apr / May 2018). https://doi.org/10.1007/978-3-319-78372-7_18 12
- Kosuge, H., Xagawa, K.: Probabilistic hash-and-sign with retry in the quantum random oracle model. In: Tang, Q., Teague, V. (eds.) PKC 2024, Part I. LNCS, vol. 14601, pp. 259–288. Springer, Cham (Apr 2024). https://doi.org/10.1007/978-3-031-57718-5_9 2, 3, 4, 5, 6, 10, 11, 14, 30
- Lyubashevsky, V.: Fiat-Shamir with aborts: Applications to lattice and factoringbased signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Berlin, Heidelberg (Dec 2009). https://doi.org/10.1007/978-3-642-10366-7_35_3
- 21. NIST: Call for additional digital signature schemes for the post-quantum cryptography standardization process (Sep 2022), https://csrc.nist.gov/csrc/media/ Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf 2
- 22. NIST: Status report on the third round of the nist post-quantum cryptography standardization process (Sep 2022), https://csrc.nist.gov/publications/detail/ nistir/8413/final 2
- Sakumoto, K., Shirai, T., Hiwatari, H.: On provable security of UOV and HFE signature schemes against chosen-message attack. In: Yang, B.Y. (ed.) Post-Quantum Cryptography 4th International Workshop, PQCrypto 2011. pp. 68–82. Springer, Berlin, Heidelberg (Nov / Dec 2011). https://doi.org/10.1007/978-3-642-25405-5_5 2, 11
- Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: 35th FOCS. pp. 124–134. IEEE Computer Society Press (Nov 1994). https://doi.org/10.1109/SFCS.1994.365700 1
- Unruh, D.: Revocable quantum timed-release encryption. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 129–146. Springer, Berlin, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_8 6, 13
- Xagawa, K.: Signatures with memory-tight security in the quantum random oracle model. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part VII. LNCS, vol. 14657, pp. 30–58. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58754-2_2 2, 3, 4, 5, 6, 17, 18

The Security of Hash-and-Sign with Retry against Superposition Attacks

33

- Zhandry, M.: How to construct quantum random functions. In: 53rd FOCS. pp. 679–687. IEEE Computer Society Press (Oct 2012). https://doi.org/10.1109/FOCS.2012.37 12
- Zhandry, M.: Secure identity-based encryption in the quantum random oracle model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 758–775. Springer, Berlin, Heidelberg (Aug 2012). https://doi.org/10.1007/978-3-642-32009-5_44_14

A Proof of Lemma 4

We modify the original O2H to make it applicable in environments where the random function is being reprogrammed. Specifically, we demonstrate that O2H applies in the existence of an oracle that triggers reprogramming as in the tight adaptive reprogramming.

Lemma 4 (O2H with Adaptive Reprogramming). Let $H_0, H_1: \mathcal{X} \to \mathcal{Y}$ be functions that are reprogrammed depending on classical queries to an oracle O (H_0 and H_1 may be reprogrammed differently). Assume that $H_0(x) = H_1(x)$ for all $x \notin S$ when O is queried the same number of times with the same inputs. Let z be a random bitstring. (S, H_0, H_1, z may have arbitrary joint distribution.) Let \mathcal{A} be a quantum algorithm with q quantum queries to H_0 or H_1 and some classical queries to O. Then, there exists a quantum algorithm \mathcal{B} that, given access to the oracle H_0 and \mathcal{A} , finds an element in \mathcal{S} such that

$$\left|\Pr\left[\mathcal{A}^{|\mathsf{H}_{0}\rangle,\mathsf{O}}(z)=1\right]-\Pr\left[\mathcal{A}^{|\mathsf{H}_{1}\rangle,\mathsf{O}}(z)=1\right]\right| \leq 2q\sqrt{\Pr\left[x\leftarrow\mathcal{B}^{|\mathsf{H}_{0}\rangle,\mathsf{O},\mathcal{A}}(z):x\in\mathcal{S}_{i}\right]}$$

Proof. Lemma 4 is an adaptation of [2, Theorem 3], which consists of [2, Lemma 8] (proof assuming a pure state) and [2, Lemma 9] (proof assuming a mixed state). We only modify the former and use the latter as it is.

Let $|\psi_{H_0}^i\rangle$ and $|\psi_{H_1}^i\rangle$ be the states of $\mathcal{A}^{H_0,O}(z)$ and $\mathcal{A}^{H_1,O}(z)$ just before the (i+1)-th query, including the query input register. In [2, Lemma 8], a bound on $D_i = |||\psi_{H_0}^i\rangle - |\psi_{H_1}^i\rangle||^2$ is obtained, from which $\sqrt{D_q} = |||\psi_{H_0}^q\rangle - |\psi_{H_1}^q\rangle||$ is derived. In this proof, in addition to the adversary's state, we define states $|\phi_{H_0}^i\rangle$ and $|\phi_{H_1}^i\rangle$ that store results of past queries to O by $\mathcal{A}^{H_0,O}(z)$ and $\mathcal{A}^{H_1,O}(z)$, respectively. Let O_{H_b} be a unitary operation corresponding to the oracle queries of H_b ($b \in \{0, 1\}$) that returns $H_b(x)$ by reading the query input register along with $|\phi_{H_b}^i\rangle$. Note that H_b is reprogrammed depending on $|\phi_{H_b}^i\rangle$, and $O_{H_b} |\phi_{H_b}^i\rangle|x,y\rangle =$ $|x, y \oplus H_b(x)\rangle$ holds. Define P_S as the orthogonal projector that projects the query register onto the subspace spanned by states $|x\rangle$ such that $x \in S$. Formally, this is given by $P_S = \sum_{x \in S} |x\rangle \langle x|$. This projector P_S ensures that only those states corresponding to $x \in S$ are selected. Let $B_i = ||P_S ||\psi_{H_0}^{i-1}\rangle||^2$ be the probability of obtaining $x \in S$ by measuring query input register of $\mathcal{A}^{H_0,O}(z)$ just before the *i*-th query.

Setting $D'_{i} = \left\| \left| \phi^{i}_{\mathsf{H}_{0}} \psi^{i}_{\mathsf{H}_{0}} \right\rangle - \left| \phi^{i}_{\mathsf{H}_{1}} \psi^{i}_{\mathsf{H}_{1}} \right\rangle \right\|^{2}$ and letting U be the state transition operation between the queries to H_{b} (U includes queries to O), the following

bound is obtained.

$$\begin{split} D'_{i} &= \left\| UO_{H_{0}} \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - UO_{H_{1}} \left| \phi_{H_{1}}^{i-1} \psi_{H_{1}}^{i-1} \right\rangle \right\|^{2} \\ &= \left\| \left(O_{H_{0}} \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - O_{H_{1}} \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle \right) + \left(O_{H_{1}} \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - O_{H_{1}} \left| \phi_{H_{1}}^{i-1} \psi_{H_{1}}^{i-1} \right\rangle \right) \right\|^{2} \\ &\leq \left\| \left(O_{H_{0}} - O_{H_{1}} \right) \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| O_{H_{1}} \left(\left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \phi_{H_{1}}^{i-1} \psi_{H_{1}}^{i-1} \right\rangle \right) \right\|^{2} \\ &+ 2 \left\| \left(O_{H_{0}} - O_{H_{1}} \right) \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| O_{H_{1}} \left(\left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \phi_{H_{1}}^{i-1} \psi_{H_{1}}^{i-1} \right\rangle \right) \right\|^{2} \\ &+ 2 \left\| \left(O_{H_{0}} \left| \phi_{H_{0}}^{i-1} \right\rangle - O_{H_{1}} \left| \phi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| O_{H_{1}} \left(\left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \phi_{H_{1}}^{i-1} \psi_{H_{1}}^{i-1} \right\rangle \right) \right\|^{2} \\ &+ 2 \left\| \left(O_{H_{0}} \left| \phi_{H_{0}}^{i-1} \right\rangle - O_{H_{1}} \left| \phi_{H_{0}}^{i-1} \right\rangle \right) \left| \psi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| O_{H_{1}} \left(\left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \phi_{H_{1}}^{i-1} \psi_{H_{1}}^{i-1} \right\rangle \right) \right\|^{2} \\ &+ 2 \left\| \left(O_{H_{0}} \left| \phi_{H_{0}}^{i-1} \right\rangle - O_{H_{1}} \left| \phi_{H_{0}}^{i-1} \right\rangle \right) P_{S} \left| \psi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| O_{H_{1}} \left(\left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \phi_{H_{1}}^{i-1} \psi_{H_{1}}^{i-1} \right\rangle \right) \right\|^{2} \\ &+ 2 \left\| \left(O_{H_{0}} \left| \phi_{H_{0}}^{i-1} \right\rangle - O_{H_{1}} \left| \phi_{H_{0}}^{i-1} \right\rangle \right) P_{S} \left| \psi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| O_{H_{1}} \left(\left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \phi_{H_{1}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle \right) \right\|^{2} \\ &\leq 4 \left\| P_{S} \left| \psi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \psi_{H_{1}}^{i} \phi_{H_{1}}^{i} \right\rangle \right\|^{2} \\ &+ 4 \left\| P_{S} \left| \psi_{H_{0}}^{i-1} \right\rangle \right\|^{2} + \left\| \left| \phi_{H_{0}}^{i-1} \psi_{H_{0}}^{i-1} \right\rangle - \left| \psi_{H_{1}}^{i} \phi_{H_{1}}^{i} \right\rangle \right\|^{2} \\ &= 4B_{i} + D'_{i-1} + 4 \sqrt{B_{i} D'_{i-1}} = \left(\sqrt{D'_{i-1}} + 2 \sqrt{B_{i}} \right)^{2} \right\|^{2} \end{aligned}$$

9

Since O_{H_0} and O_{H_1} both read the same state $|\phi_{\mathsf{H}_0}^{i-1}\rangle$, it follows that $\mathsf{H}_0(x) = \mathsf{H}_1(x)$ for all $x \notin \mathcal{S}$. Therefore, $(O_{\mathsf{H}_0} |\phi_{\mathsf{H}_0}^{i-1}\rangle - O_{\mathsf{H}_1} |\phi_{\mathsf{H}_0}^{i-1}\rangle)$ only applies to $|x, y\rangle$ such that $x \in \mathcal{S}$. Consequently, even if $P_{\mathcal{S}}$ projects the query input register onto the subspace spanned by $|x\rangle$ such that $x \in \mathcal{S}$, the state after applying $(O_{\mathsf{H}_0} |\phi_{\mathsf{H}_0}^{i-1}\rangle - O_{\mathsf{H}_1} |\phi_{\mathsf{H}_0}^{i-1}\rangle)$ does not change. Therefore, we have

$$\left(O_{\mathsf{H}_{0}}\left|\phi_{\mathsf{H}_{0}}^{i-1}\right\rangle - O_{\mathsf{H}_{1}}\left|\phi_{\mathsf{H}_{0}}^{i-1}\right\rangle\right)P_{\mathcal{S}} = \left(O_{\mathsf{H}_{0}}\left|\phi_{\mathsf{H}_{0}}^{i-1}\right\rangle - O_{\mathsf{H}_{1}}\left|\phi_{\mathsf{H}_{0}}^{i-1}\right\rangle\right).$$

Since $D'_0 = 0$ and $\sqrt{D_q} \le \sqrt{D'_q}$ hold, we have

$$\sqrt{D_q} \le \sqrt{D_q'} \le 2\sum_{i \in [q]} \sqrt{B_i} \le 2q \sqrt{\sum_{i \in [q]} \frac{1}{q} B_i} = 2q \sqrt{\Pr\left[x \leftarrow \mathcal{B}^{|\mathsf{H}_0\rangle, \mathcal{A}}(z) : x \in \mathcal{S}\right]}.$$

Hence, we have $\||\psi_{\mathsf{H}_0}^q\rangle - |\psi_{\mathsf{H}_1}^q\rangle\| \leq 2q\sqrt{\Pr[x \leftarrow \mathcal{B}|_{\mathsf{H}_0}), \mathcal{A}(z) : x \in \mathcal{S}]}$. Extending this result to O2H for mixed states using the proof of [2, Lemma 9], we have this lemma.

B Variations of Hash-and-Sign

We present the original hash-and-sign and its three variations in Figs. 17 and 18.

C Quantum Preimage-Simulatability

We define the quantum preimage-simulatability as follows.

The Security of Hash-and-Sign with Retry against Superposition Attacks 35

$ \frac{\operatorname{KeyGen}(1^{\lambda})}{1 (vk, sk)} \leftarrow \operatorname{Gen}(1^{\lambda}) \\ 2 \operatorname{return} \ (vk, sk) \\ \operatorname{Sign}(sk, m) $	$ \begin{array}{c} \displaystyle \underbrace{KeyGen(1^{\lambda})}_{1 (vk, sk)} \leftarrow Gen(1^{\lambda}) \\ 2 s' \leftarrow_{\$} \mathcal{K} \\ 3 \mathbf{return} \ (vk, (sk, s')) \end{array} $
$ \begin{array}{c} \mathbf{s} y := \mathbf{H}(m) \\ 4 x \leftarrow \ln \mathbf{v}(sk, y) \\ 5 \mathbf{return} x \\ \end{array} $ Vrfv(vk, m, x)	$ \begin{array}{c} \underline{Sign((sk,s'),m)} \\ 4 \hspace{0.5cm} y \coloneqq H(m) \\ 5 \hspace{0.5cm} x \coloneqq Inv(sk,y;PRF'(s',m)) \\ 6 \hspace{0.5cm} \mathbf{return} \hspace{0.5cm} x \end{array} $
$\overbrace{6 \text{ return } \llbracket F(vk, x) = H(m) \rrbracket$	$\frac{Vrfy(vk,m,x)}{7 \; \mathbf{return} \; [\![F(vk,x) = H(m)]\!]}$

Fig. 17: Algorithms of original hash-and-sign (HS) and derandomized hash-and-sign (DHS)

$ \begin{array}{ c c }\hline {\sf KeyGen}(1^{\lambda}) \\ \hline {\bf 1} & (vk, sk) \leftarrow {\sf Gen}(1^{\lambda}) \\ {\bf 2} & {\sf return} & (vk, sk) \end{array} $	$ \begin{array}{c} \displaystyle \frac{KeyGen(1^{\lambda})}{1 (vk, sk)} \leftarrow Gen(1^{\lambda}) \\ 2 (s, s') \leftarrow_{\$} \mathcal{K} \times \mathcal{K} \end{array} $
Sign(sk,m)	3 return $(vk, (sk, s, s'))$
$\begin{array}{c} 3 r \leftarrow_{\mathbf{S}} \mathcal{R} \\ 4 y := H(r, m) \end{array}$	$\frac{\text{Sign}((sk, s, s'), m)}{A : r := \text{PRE}(s, m)}$
5 $x \leftarrow \text{Inv}(sk, y)$ 6 return (r, x)	
Vrfy(vk,m,(r,x))	7 return (r, x)
7 return $\llbracket F(vk, x) = H(r, m) \rrbracket$	$\frac{Vrfy(vk, m, (r, x))}{\mathbf{s} \; \mathbf{return} \; [\![F(vk, x) = H(r, m)]\!]}$

Fig. 18: Algorithms of probabilistic hash-and-sign (PHS) and derandomized probabilistic hash-and-sign (DPHS)

Definition 9 (Quantum Preimage Simulatablity). Let T be a TDF with SampDom. Using a game defined in Fig. 19, we define an advantage function of an adversary playing the qPS (Quantum Preimage Sampling) game against T as $\operatorname{Adv}_{T}^{qPS}(\mathcal{A}) = |\Pr[q\mathsf{PS}_{0}^{\mathcal{A}}=1] - \Pr[q\mathsf{PS}_{1}^{\mathcal{A}}=1]|$. We say T is quantum preimagesimulatable if its advantage is negligible for any efficient adversary.

In the proof of Theorem 1, we have the following lemma.

Lemma 23. Suppose that T is (γ, β) -correct. There exists a qPS adversary A_{qps} of T such that

$$|\Pr[W_{10.0}] - \Pr[W_{10.1}]| \le \mathrm{Adv}_{\mathsf{T}}^{q_{\mathrm{PS}}}(\mathcal{A}_{\mathsf{qps}}) + \sqrt{6(q_{\mathsf{H}} + 2q_{\mathsf{F}})^3(1 - \gamma(1 - \beta^B))}.$$

Proof. Before taking the bound using the qPS advantage, we introduce an intermediate game $G_{10.05}$ between $G_{10.0}$ and $G_{10.1}$, in which GetLogs obtains x_k after the unbounded loop iterations and computes $y_k := \mathsf{F}(vk, x_k)$ instead of using

$\mathbf{GAME} \ qPS_b$	$\operatorname{QSAMPLE}_0(m)$
1 RF _{inv} $\leftarrow_{\$}$ Func($\mathcal{M} \times [B], \mathcal{R}'$) 2 RF' _H $\leftarrow_{\$}$ Func($\mathcal{M} \times [B], \mathcal{Y}$) 3 RF _{sd} $\leftarrow_{\$}$ Func($\mathcal{M}, \mathcal{R}''$) 4 $(vk, sk) \leftarrow \text{Gen}(1^{\lambda})$ 5 $b^* \leftarrow \mathcal{A}^{ \text{QSAMPLE}_b\rangle}(vk)$ 6 return b^*	7 $k \coloneqq 0$ 8 repeat 9 $\mid k \coloneqq k + 1$ 10 $\mid y \coloneqq RF'_{H}(m, k)$ 11 $\mid x \coloneqq Inv(sk, y; RF_{inv}(m, k))$ 12 until $x \neq \bot$ 13 return x
	$\frac{\text{QSAMPLE}_1(m)}{14 \ x := \text{SampDom}(vk; RF_{sd}(m))$

Fig. 19: qPS game

the y_k generated during the loop. Let \mathcal{D}_m and \mathcal{D}'_m denote the distributions of $(x, y) \in \mathcal{X}' \times \mathcal{Y}$ output by $\mathsf{GetLogs}(m)$ in $G_{10.0}$ and $G_{10.05}$, respectively, where $\mathcal{X}' = \mathcal{X} \cup \{\bot\}$. The outputs of $\mathsf{GetLogs}(m)$ differ in cases where it returns $x = \bot$ or when $x \neq \bot$ but $\mathsf{F}(vk, x) \neq y$ holds in $G_{10.05}$. Thus, the statistical distance between \mathcal{D}_m and \mathcal{D}'_m is bounded as follows:

$$\sum_{\substack{(x,y)\in\mathcal{X}'\times\mathcal{Y}\\ :x=\perp\forall\mathsf{F}(vk,x)\neq y}} |\mathcal{D}_m(x,y) - \mathcal{D}'_m(x,y)| = \sum_{\substack{(x,y)\in\mathcal{X}'\times\mathcal{Y}\\ :x=\perp\forall\mathsf{F}(vk,x)\neq y}} \mathcal{D}_m(x,y)$$
$$= 1 - \sum_{x\in\mathcal{X}} \mathcal{D}_m(x,\mathsf{F}(vk,x))$$
$$\leq 1 - \gamma \left(1 - \beta^B\right)$$

Applying Lemma 5, we have

$$|\Pr[W_{10.05}] - \Pr[W_{10.1}]| \le \sqrt{6 \left(q_{\mathsf{H}} + 2q_{\mathsf{F}}\right)^3 \left(1 - \gamma \left(1 - \beta^B\right)\right)}.$$
 (4)

Then, the qPS adversary \mathcal{A}_{qps} can simulate $G_{10.05}$ and $G_{10.1}$ as in Fig. 20. The qPS adversary \mathcal{A}_{qps} simulates $G_{10.05}$ if b = 0; otherwise, it simulates $G_{10.1}$. Therefore,

$$\left|\Pr[W_{10.05}] - \Pr[W_{10.1}]\right| \le \operatorname{Adv}_{\mathsf{T}}^{q\operatorname{PS}}(\mathcal{A}_{\mathsf{qps}}) \tag{5}$$

From Eqs. (4) and (5), we have the bound in this lemma.



Fig. 20: Simulation of the modified PO game by qPS adversary