A Security Proof of the GLP Signature Scheme

Thanh Xuan Khuc[1(✉)], Minh Kim Bui[2], and Hien Chu[3]

[1] Institute of Cryptography Science and Technology, 21 Truc Khe, Lang Ha, Dong Da, Hanoi, Vietnam
khucxuanthanh@gmail.com

[2] Ho Chi Minh University of Science, Vietnam National University, 223 Nguyen Van Cu, District 5, Ho Chi Minh City, Vietnam
kmath93@gmail.com

[3] Ho Chi Minh City University of Education, 280 An Duong Vuong, District 5, Ho Chi Minh City, Vietnam
hienchu.1610@gmail.com

Abstract. In 2012, Tim Güneysu, et al. proposed the GLP signature scheme, a practical and efficient post-quantum signature scheme. It is built on the modification of Vadim Lyubashevsky's idea of constructing previous signature schemes. It has a significantly smaller signature and key size than prior signature scheme. The design of the GLP is a foundation to construct newer signature schemes such as Bai-Galbraith, Dilithium. However, Tim Güneysu has only given the description of the GLP signature scheme that has not yet given a detailed security proof for this scheme. Therefore, in this paper, we will present a full security proof for the GLP signature scheme. Specifically, we show that the GLP signature scheme is EU-CMA secure in the random oracle model.

Keywords: Latticed-based signature · R-SIS problem · The GLP signature scheme · Post-quantum cryptography

1 Introduction

The security of currently popular signature schemes is based on hard problems in number theory, such as integer factorization or discrete logarithm problem. More than 20 years ago, Peter Shor proposed an efficient algorithm to solve these hard arithmetic problems on a quantum computer [Sho99]. Therefore, as soon as quantum computers achieve sufficient computational power, widely-used signature schemes will be insecure. This urges researchers to search for new cryptography primitives resistant to quantum computing-based attacks. Due to the recent innovative development of quantum computers, post-quantum cryptography becomes more and more crucial. Among the candidates for post-quantum cryptography, lattice-based cryptography is the most potential candidates.

The last few years have witnessed surprising development of lattice-based cryptography, especially constructions of new signatures schemes. At present, lattice-based signatures have become more practical and competitive with classical signatures related to efficiency and security level. Among the methods of constructing signatures, Fiat-Shamir technique is attracting the attention of the cryptography community for its effectiveness and practicality. The first Fiat-Shamir lattice-based signature was introduced by Lyubashevsky [Lyu08] in 2008. This scheme based on the hardness of the shortest vector problem. Later, several improved schemes was proposed in [Lyu12, DDLL13, BG14, DLL+17]. One of them, the Dilithium signature scheme, has been submitted to NIST to standardize as a post-quantum signature.

The GLP signature scheme [GLP12][1] is constructed by adapting the idea from [Lyu12] and [Lyu08]. Concretely, GLP follows the idea that ephemeral values in signing algorithm are chosen randomly from a uniform distribution over a set the same as in [Lyu08], instead of from a Gaussian distribution in [Lyu12]. Moreover, the way how to select a valid signature of GLP is the same as the proposed way in [Lyu08] (in [Lyu08] the signatures is simply checked to be in some interval or not, while [Lyu12] uses rejection sampling to output a valid signature). However, due to the usage of optimized parameters to reduce the signature size and key size, the security proof of GLP was founded on the proof in [Lyu12]. Key size and signature size of GLP is significantly smaller than those of other signature schemes in [Lyu12, Lyu08]. Besides, the construction idea of GLP is a foundation to construct later improved scheme such as [BG14, DLL+17].

Our contribution: In [GLP12], the authors described the GLP signature scheme but did not prove its security in details (concretely, [GLP12] provided a guideline to prove based on [Lyu12]). Therefore, our main contribution is to present a full security proof for the GLP signature scheme based on previous technique in [Lyu08, Lyu12]. We show that the GLP signature scheme is EU-CMA secure in the random oracle model under assuming the hardness of the worst-case lattice problems.

Organization of the paper: In Sect. 2, we recall the definitions of SIS problem and its hardness on ideal lattices. The description of GLP is recalled in Sect. 3. In Sect. 4, we present a security proof of the GLP signature scheme in the random oracle model. Finally, the conclusion of this paper is described in Sect. 5.

2 Preliminaries

Throughout the paper, we will assume that $n = 2^\alpha$ where α is a positive integer, $p \equiv 1 \bmod 2n$ and $R^{p^n} = \mathbb{Z}_p[x]/\langle x^n + 1\rangle$. Each element of R^{p^n} can be presented as a polynomial of which the degree is at most $n-1$ and the coefficients are in $\left[-\frac{p-1}{2}, \frac{p-1}{2}\right]$. We let $R_k^{p^n}$ be a subset of the ring R^{p^n} in which the elements of $R_k^{p^n}$ can be presented as polynomials with degree at most $n-1$ and the

[1] A later scheme version is given in [GLP15]. But, it still does not contain a full security proof.

coefficients in $[-k, k]$, where $|k| < \frac{p-1}{2}$. For any set S, let $s \leftarrow_{\$} S$ mean that s is chosen uniformly at random from S. We denote $U(0,1)$ to be the uniform distribution on $(0,1)$. Let $\mathbf{s} = s_0 + s_1x + \ldots + s_{n-1}x^{n-1}$ be a polynomial, we denote $\|\mathbf{s}\|_\infty = \max_{i=0}^{n-1} |s_i|$.

Definition 1 (R-SIS$_{p,n,\gamma,\beta}$ problem, [Lyu12]). *Given the polynomials* $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_\gamma$ *are chosen uniformly random from* R^{p^n}*, find the polynomials* $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_\gamma \neq 0$ *in* $R^{p^n}_\beta$ *such that* $\mathbf{a}_1\mathbf{s}_1 + \mathbf{a}_2\mathbf{s}_2 + \ldots + \mathbf{a}_\gamma\mathbf{s}_\gamma = 0$.

Definition 2 (R-SIS$_{p,n,\gamma,k}$ distribution, [Lyu12]). *The R-SIS$_{p,n,\gamma,k}$ distribution is sampled by choosing* $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_\gamma$ *uniformly random from* R^{p^n}*, the polynomials* $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_\gamma$ *uniformly random from* $R^{p^n}_k$ *and outputting* $(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_\gamma, \mathbf{t} = \mathbf{a}_1\mathbf{s}_1 + \mathbf{a}_2\mathbf{s}_2 + \ldots + \mathbf{a}_\gamma\mathbf{s}_\gamma)$.

Definition 3 (R-SIS$_{p,n,\gamma,k}$ decision problem, [Lyu12]). *Given* $(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_\gamma, \mathbf{t})$*, distinguish whether* $(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_\gamma, \mathbf{t})$ *are generated from the R-SIS$_{p,n,\gamma,k}$ distribution or chosen uniformly random from* $(\underbrace{R^{p^n} \times \ldots . \times R^{p^n}}_{\gamma}, R^{p^n})$.

The R-SIS$_{p,n,\gamma,\beta}$ problem is consider as a hard problem. Solving the problem R-SIS$_{p,n,\gamma,\beta}$ have the hardness as solving worst-case lattice problems in ideal lattices [LM06].

Definition 4 (DCK$_{p,n}$ problem, [GLP12]). *Distinguish between the uniform distribution over* $R^{p^n} \times R^{p^n}$ *and the distribution* $(\mathbf{a}, \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2)$*, where* $\mathbf{s}_i$ *are uniformly random in* $R^{p^n}_1$ *and* $\mathbf{a}$ *is uniformly random in* R^{p^n}.

As in [GLP12], the DCK$_{p,n}$ problem is also considered as a hard problem. The following lemma indicates a reduction from the DCK$_{p,n}$ problem to the R-SIS$_{p,n,2,3\alpha+1}$ decision problem. In other words, if one can solve the R-SIS$_{p,n,2,3\alpha+1}$ decision problem, then one can solve the DCK$_{p,n}$ problem.

Lemma 1 [Lyu12]**.** *Let α be a non-negative integer such that* $GCD(2\alpha+1, p) = 1$*, then there exists a polynomial-time reduction from the* DCK$_{p,n}$ *problem to the* R-SIS$_{p,n,2,3\alpha+1}$ *decision problem.*

Proof. See [Lyu12], Lemma 3.6, p. 7. □

The following lemma shows that, with suitable parameters, if one can solve the R-SIS$_{p,n,2,\beta}$ problem then one can solve the DCK$_{p,n}$ problem.

Lemma 2. [2] *If* $8\beta n \leq p$ *then there is a polynomial-time randomized reduction from the* DCK$_{p,n}$ *problem to the* R-SIS$_{p,n,2,\beta}$ *problem.*

[2] This lemma is stated based on Lemma 3.7 in [Lyu12]. Namely, we give a reduction for the hard problems on the ideal lattices instead of the lattice in $\mathbb{R}^n$ as in Lemma 3.7.

Proof. Given an instance $(\mathbf{a}, \mathbf{t})$ of the $\mathrm{DCK}_{p,n}$ problem. Using the $\text{R-SIS}_{p,n,2,\beta}$ oracle on $(\mathbf{a}, 1)$ to find $\mathbf{s}'_1, \mathbf{s}'_2 \in \mathcal{R}_\beta^{p^n}$ such that $\mathbf{a}\,\mathbf{s}'_1 + \mathbf{s}'{}_2 = 0$. If $\mathbf{t} = \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2$ then

$$\mathbf{s}'_1\,\mathbf{t} = \mathbf{s}'_1\,(\mathbf{a}\mathbf{s}_1 + \mathbf{s}_2) = \mathbf{a}\,\mathbf{s}'_1\,\mathbf{s}_1 + \mathbf{s}'_1\,\mathbf{s}_2 = -\,\mathbf{s}'_2\,\mathbf{s}_1 + \mathbf{s}'_1\,\mathbf{s}_2.$$

Since $\|\mathbf{s}_1\|_\infty, \|\mathbf{s}_2\|_\infty \le 1$, $\|\mathbf{s}'_1\|_\infty, \|\mathbf{s}'_2\|_\infty \le \beta$, and [[Lyu08], Lemma 2.8], we have

$$\|\mathbf{s}'_2\,\mathbf{s}_1\|_\infty \le n\|\mathbf{s}'_2\|_\infty\|\mathbf{s}_1\|_\infty \le n\beta$$
$$\|\mathbf{s}'_1\,\mathbf{s}_2\|_\infty \le n\|\mathbf{s}'_1\|_\infty\|\mathbf{s}_2\|_\infty \le n\beta$$

Hence $\|\mathbf{s}'_1\,\mathbf{t}\|_\infty = \|-\,\mathbf{s}'_2\,\mathbf{s}_1 + \mathbf{s}'_1\,\mathbf{s}_2\|_\infty \le 2n\beta \le \frac{p}{4}$. On the other hand, if $\mathbf{t}$ is uniformly random R^{p^n} then $\|\mathbf{s}'_1\,\mathbf{t}\|_\infty$ will also be uniformly random in $\mathbb{Z}_p$. Hence, in order to solve the $\mathrm{DCK}_{p,n}$ problem, distinguisher simply looks at the value $\|\mathbf{s}'_1\,\mathbf{t}\|_\infty$. If $\|\mathbf{s}'_1\,\mathbf{t}\|_\infty \le \frac{p}{4}$ then he says that $(\mathbf{a}, \mathbf{t})$ is an instance of the $\mathrm{DCK}_{p,n}$ problem. Otherwise, he says that $(\mathbf{a}, \mathbf{t})$ is chosen uniformly from $R^{p^n} \times R^{p^n}$. In the case $(\mathbf{a}, \mathbf{t})$ is an instance of the $\mathrm{DCK}_{p,n}$ problem, the distinguisher will be correct. However, in the case of the uniform distribution $(\mathbf{a}, \mathbf{t})$ is uniformly on $R^{p^n} \times R^{p^n}$, he will make an error with probability $\frac{1}{2}$ (because $\mathbf{t}$ is chosen uniformly from R^{p^n}, the probability that $\|\mathbf{s}'_1\,\mathbf{t}\|_\infty \le \frac{p}{4}$ is $\frac{1}{2}$). □

A signature scheme includes three algorithms: Keygen, Sign and Verify. Keygen takes as input a security parameter and outputs a public key pk and secret key sk. Sign takes as input a message μ and a secret key sk, outputs a signature Σ. Verify takes as input a message μ, signature Σ and public key pk, output valid (1) or invalid (0). We require that, with the non-negligible probability, $\mathsf{Verify}\,(\mu, \Sigma, pk) = 1$.

The standard security notion for signatures is existential unforgeability under chosen message attacks (EU-CMA). Consider the following game of challenger $\mathcal{C}$ and forger $\mathcal{F}$. Firstly, the challenger generates a key pair (sk, pk) and sends pk to $\mathcal{F}$. The forger takes as input public key pk. Besides, the forger can make the polynomial queries for signatures on messages $\mu_1, \ldots, \mu_Q$ of its choice. For the i-th query, the challenger answers (μ_i, Σ_i). The output of the forger is (μ^*, Σ^*). It wins the game if $\mathsf{Verify}\,(\mu^*, \Sigma^*, pk) = 1$, and $\mu^* \ne \mu_i$ for any $i = 1, \ldots, Q$. The signature scheme is called EU-CMA secure if there is no polynomial-time $\mathcal{F}$ whose success probability in the above game is non-negligible.

3 The GLP Signature Scheme

The signature scheme has three main algorithms: key generation, signing and verifying. In particular, the signing and verifying algorithm can access to the random oracle H is defined in [GLP12] as follows: $H : \{0,1\}^* \to \{\mathbf{v} : \mathbf{v} \in R_1^{p^n}, \sum_{i=0}^{n-1} |v_i| = 32\}$, where $\mathbf{v} = v_0 + v_1 x + \ldots + v_{n-1}x^{n-1} \in R_1^{p^n}$.

Key Generation Algorithm

Input: Parameters (n, p).
Output: Secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and public key $(\mathbf{a}, \mathbf{t})$.

```
1. Choose secret key $\mathbf{s}_1, \mathbf{s}_2 \leftarrow_{\$} R_1^{p^n}$.
2. Choose $\mathbf{a} \leftarrow_{\$} R^{p^n}$.
3. Compute $\mathbf{t} \leftarrow \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2$.
4. Return secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and public key $(\mathbf{a}, \mathbf{t})$.
```

Signing Algorithm

Input: Parameters (n, p, k), message μ, secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and $\mathbf{a}$.
Output: A signature for message μ.

```
1. Choose $\mathbf{y}_1, \mathbf{y}_2 \leftarrow_{\$} R_k^{p^n}$.
2. Compute $\mathbf{c} \leftarrow H(\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2, \mu)$.
3. Compute $\mathbf{z}_1 \leftarrow \mathbf{s}_1\mathbf{c} + \mathbf{y}_1, \mathbf{z}_2 \leftarrow \mathbf{s}_2\mathbf{c} + \mathbf{y}_2$.
4. If $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$ then
5.    Output $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$
6. Else return to Step 1.
```

Verifying Algorithm

Input: The parameters (n, p, k), public key $(\mathbf{a}, \mathbf{t})$, message μ and signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$.
Output: The signature is valid or invalid.

```
1. If $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$ and $\mathbf{c} = H(\mathbf{a}\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu)$ then
2.    The signature is valid.
3. Else
4.    The signature is invalid.
```

The secret keys are random polynomials $\mathbf{s}_1, \mathbf{s}_2 \leftarrow_{\$} R_1^{p^n}$. The public key is $(\mathbf{a}, \mathbf{t})$, where $\mathbf{a} \leftarrow_{\$} R^{p^n}$ and $\mathbf{t} = \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2 \in R^{p^n}$.

To sign message μ, firstly, the signer chooses random polynomials $\mathbf{y}_1, \mathbf{y}_2 \in R_k^{p^n}$. Then the signer compute $\mathbf{c} = H(\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2, \mu)$ and $\mathbf{z}_1 = \mathbf{s}_1\mathbf{c} + \mathbf{y}_1, \mathbf{z}_2 = \mathbf{s}_2\mathbf{c} + \mathbf{y}_2$. Before outputting the signature for the message μ, the signer will check if $\mathbf{z}_1, \mathbf{z}_2$ has belong to $R_{k-32}^{p^n}$ or not. If $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$ then signer will output $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$ as the signature for μ. Otherwise, the signer regenerate $\mathbf{y}_1, \mathbf{y}_2$ and recalculate the signature. The signer will perform the signing algorithm until it can give $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$. With the parameters chosen as in Table 1, Lemma 3 will show that the average signer needs to make approximately 7 times to generate a valid signature.

To verify the signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$, the verifier simply checks that $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$ and that $\mathbf{c} = H(\mathbf{a}\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu)$. If $\mathbf{z}_1, \mathbf{z}_2$ are generated from the signing algorithm then we have $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$ and

$$\begin{aligned} H\left(\mathbf{a}\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu\right) &= H\left(\mathbf{a}\left(\mathbf{s}_1\mathbf{c} + \mathbf{y}_1\right) + \left(\mathbf{s}_2\mathbf{c} + \mathbf{y}_2\right) - \left(\mathbf{a}\mathbf{s}_1 + \mathbf{s}_2\right)\mathbf{c}, \mu\right) \\ &= H\left(\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2, \mu\right) = \mathbf{c} \end{aligned}$$

Table 1. The GLP signature scheme parameters

		I	II
1.	n	512	1024
2.	p	8383489	16760833
3.	k	2^{14}	2^{15}
4.	The average number of executions to generate a valid signature	7	7
5.	Signature size (bit) $\approx 2n \log(2(k-32)+1)+n$	15869	33789
6.	Secret key size (bit) $\approx 2n \log(3)$	1623	3246
7.	Public key size $\approx n \log p$	11775	24574

The Table 1 [GLP12] shows the parameters n, p, k, the average number of executions to generate a valid signature, signature size and keys size used in the GLP signature scheme.

The parameter k controls the trade-off between the security and the run time of the scheme. The smaller k gets, the more secure the scheme becomes and the shorter the signatures get but the time to sign will increase. The authors of the implementation of [GLP12] suggest $k = 2^{14}, n = 512$ and $p = 8383489$ for ≈80 bits of security and $k = 2^{15}, n = 1024$ and $p = 16760833$ for >256 bits of security.

The size of the signature is the number of bits used to represent $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$. Since $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$, $\mathbf{z}_1, \mathbf{z}_2$ can be represented by $2n \log(2(k-32)+1)$. And $\mathbf{c}$ can be represented by n bits. Therefore, the signature size can be represented by $2n \log(2(k-32)+1)+n$ bits.

Since $\mathbf{s}_1, \mathbf{s}_2 \in R_1^{p^n}$, the size of secret key can be represented by $2n \log(3)$. The public key consists of two polynomials $(\mathbf{a}, \mathbf{t})$, where $\mathbf{a}$ is shared by all users (therefore, can be viewed as part of the signature scheme) and $\mathbf{t}$ is the individual component of each user. Therefore, the public key size with each user is just a component $\mathbf{t} \in R^{p^n}$ and can be represented by $n \log p$.

Lemma 3. [3] *For any* $\mathbf{w} \in R^{p^n}$ *such that* $\|\mathbf{w}\|_\infty \leq 32$,

$$\mathsf{Pr}\left[\mathbf{w} + \mathbf{y} \in R_{k-32}^{p^n} : \mathbf{y} \leftarrow_{\$} R_k^{p^n}\right] = \left(1 - \frac{64}{2k+1}\right)^n.$$

Proof. Let some $\mathbf{w} \in R^{p^n}$ such that $\|\mathbf{w}\|_\infty \leq 32$ and consider $\mathbf{w}$ as a vector of dimension n with coefficients w_j (for $1 \leq j \leq n$) having absolute value at most 32. Then the sum $\mathbf{w} + \mathbf{y}$ will belong to $R_{k-32}^{p^n}$ if for every coefficient w_i the corresponding coefficient of $\mathbf{y}$ (denoted y_j) is in the range

$$[-(k-32) - w_j, k - 32 - w_j] \tag{1}$$

Since every coefficient y_j is generated randomly in the range $[-k, k]$ and $|w_j| \leq 32$, the range (1) is contained in the range of possible coefficient y_j of $\mathbf{y}$. The

[3] This lemma is stated based on Lemma 6.1 in [Lyu08].

probability that y_j is in the range (1) is $\frac{2(k-32)+1}{2k+1} = 1 - \frac{64}{2k+1}$. Therefore,

$$\mathsf{Pr}\left[\mathbf{w}+\mathbf{y} \in R_{k-32}^{p^n} : \mathbf{y} \leftarrow_{\$} R_k^{p^n}\right] = \left(1 - \frac{64}{2k+1}\right)^n .$$

□

Since $\|\mathbf{s}_1\|_\infty \leq 1, \|\mathbf{s}_2\|_\infty \leq 1$ and $\|\mathbf{c}\|_1 \leq 32$, we have $\|\mathbf{s}_1\mathbf{c}\|_\infty \leq 32$ and $\|\mathbf{s}_2\mathbf{c}\|_\infty \leq 32$. Hence, we have

$$\mathsf{Pr}\left[\mathbf{s}_1\mathbf{c}+\mathbf{y}_1 \in R_{k-32}^{p^n} : \mathbf{y}_1 \leftarrow_{\$} R_k^{p^n}\right] = \left(1 - \frac{64}{2k+1}\right)^n$$

and

$$\mathsf{Pr}\left[\mathbf{s}_2\mathbf{c}+\mathbf{y}_2 \in R_{k-32}^{p^n} : \mathbf{y}_2 \leftarrow_{\$} R_k^{p^n}\right] = \left(1 - \frac{64}{2k+1}\right)^n$$

Moreover, $\mathbf{z}_1 = \mathbf{s}_1\mathbf{c}+\mathbf{y}_1$ and $\mathbf{z}_2 = \mathbf{s}_2\mathbf{c}+\mathbf{y}_2$, the probability that $\mathbf{z}_1, \mathbf{z}_2$ belong to $R_{k-32}^{p^n}$ is

$$\mathsf{Pr}\left[\mathbf{z}_1 \in R_{k-32}^{p^n} \wedge \mathbf{z}_1 \in R_{k-32}^{p^n} : \mathbf{y}_1, \mathbf{y}_2 \leftarrow_{\$} R_k^{p^n}\right] = \left(1 - \frac{64}{2k+1}\right)^{2n} .$$

We can see that if k is too small then the probability that $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$ is very low. The below table show the relationship between n, k and the average number of executions to generate a valid signature (Table 2).

Table 2. The relationship between n, k and the average number of executions to generate a valid signature

n	k	The average number of executions to generate a valid signature
512	2^{14}	7
512	2^{15}	3
512	2^{16}	2
1024	2^{14}	55
1024	2^{15}	7
1024	2^{16}	3
1024	2^{17}	2

4 Security Proof of the GLP Signature Scheme

In this section, we show that the GLP signature scheme is EU-CMA secure in the random oracle model, assuming the hardness of the DCK$_{p,n}$ problem and the

R-SIS$_{q,n,2,\beta}$ problem. In [GLP12], it's a bit confusing when saying that the security of the GLP signature scheme is based only on the hardness of the DCK$_{p,n}$ problem. Because [GLP12] assumes that there is a polynomial-time reduction from the DCK$_{p,n}$ problem to the R-SIS$_{q,n,2,\beta}$ problem as Lemma 2. However, we can easily see that if the parameters n, p, k chosen as in [GLP12] (Table 1) then the condition $8\beta n \leq p$ in Lemma 2 is not satisfied. In other words, there is no polynomial-time reduction from the DCK$_{p,n}$ problem to the R-SIS$_{q,n,2,\beta}$ problem with the parameters chosen as in [GLP12] by using Lemma 2.

Firstly, we see that if a adversary can derive the secret key $(\mathbf{s}_1, \mathbf{s}_2)$ from the public key $(\mathbf{a}, \mathbf{t})$ then he can be used to solve DCK$_{p,n}$ problem. Therefore, the security of the GLP signature scheme must be based on DCK$_{p,n}$ problem.

The following theorem shows the security of the GLP signature scheme also need to be based on R-SIS$_{q,n,2,\beta}$ problem. Assume, for contradiction, that there exists a polynomial-time forger $\mathcal{F}$ breaking the EU-CMA security of the signature scheme with non-negligible advantage. Then, we can use $\mathcal{F}$ to solve the R-SIS$_{q,n,2,\beta}$ problem.

Theorem 1. [4] *If there is a polynomial-time adversary who can produce a valid signature with success probability δ after making s queries to the signing oracle and h queries to the random oracle H, then there exists a polynomial-time algorithm to solve* R-SIS$_{p,n,2,\beta}$ *problem, where* $\beta = (2(k-32) + 64k')$, $k' = 3\left\lceil \frac{2^{\frac{100}{n}-1}\sqrt{p}-1}{3} \right\rceil + 1$, *with probability at least* $\approx \frac{\delta^2}{2(h+s)}$.

Proof. This theorem is proven through Lemmas 5 and 6 using the hybrid arguments. To be more precise, Lemma 5 shows that the actual signing algorithm and the actual key generation algorithm can be replaced by the key generation algorithm Hybrid 3 and the signing algorithm Hybrid 3 (those are obtained from the actual key generation algorithm and the actual signing algorithm. The advantage of the adversary in distinguishing the actual algorithms and the Hybrid 3 algorithms is negligible). Therefore, if the adversary is able to produce a valid signature with probability δ for the actual key generation algorithm and actual signing algorithm, then he can also produce a valid signature with probability δ for the key generation algorithm Hybrid 3 and signing algorithm Hybrid 3.

In Lemma 6, we show that if an adversary is able to produce a valid signature with probability δ for the key generation algorithm Hybrid 3 and signing algorithm Hybrid 3, then we can use that signature to solve the R-SIS$_{p,n,2,\beta}$ problem, where $\beta = (2(k-32) + 64k')$, $k' = 3\left\lceil \frac{2^{\frac{100}{n}-1}\sqrt{p}-1}{3} \right\rceil + 1$, with success probability at least

$$\left(\frac{1}{2} - 2^{-200}\right)\left(\delta - 2^{-200}\right)\left(\frac{\delta - 2^{-200}}{h+s} - 2^{-200}\right) \approx \frac{\delta^2}{2(h+s)}.$$

□

[4] This theorem is stated based on Theorem 5.1 in [Lyu12]. Namely, we provide an additional algorithms Hybrid 3 to prove the security of the GLP signature scheme.

The Hybrid algorithms are described as follows.

Key Generation Algorithm Hybrid 1 and Hybrid 2

Input: Parameters (n, p).
Output: Secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and public key $(\mathbf{a}, \mathbf{t})$.

```
1. Choose secret key $\mathbf{s}_1, \mathbf{s}_2 \leftarrow_{\$} R_1^{p^n}$.
2. Choose $\mathbf{a} \leftarrow_{\$} R^{p^n}$.
3. Compute $\mathbf{t} \leftarrow \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2$.
4. Return secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and public key $(\mathbf{a}, \mathbf{t})$.
```

Signing Algorithm Hybrid 1

Input: Parameters (n, p, k), message μ, secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and $\mathbf{a}$.
Output: A signature for μ.

```
1. Choose $\mathbf{y}_1, \mathbf{y}_2 \leftarrow_{\$} R_k^{p^n}$.
2. Choose $\mathbf{c} \leftarrow_{\$} \{\mathbf{v} \in R_1^{p^n} : \sum_{i=1}^{n} |v_i| = 32\}$
3. Compute $\mathbf{z}_1 \leftarrow \mathbf{s}_1\mathbf{c} + \mathbf{y}_1, \mathbf{z}_2 \leftarrow \mathbf{s}_2\mathbf{c} + \mathbf{y}_2$.
4. If $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$ then
5.    Output a signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$
6.    Program[5] $\mathbf{c} = H(\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu)$
7. Else return to Step 1.
```

Signing Algorithm Hybrid 2 and Hybrid 3

Input: Parameters (n, p, k), message μ, secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and $\mathbf{a}$
Output: A signature for μ.

```
1. Choose $\mathbf{c} \leftarrow_{\$} \{\mathbf{v} \in R_1^{p^n} : \sum_{i=1}^{n} |v_i| = 32\}$
2. Choose $\mathbf{z}_1 \leftarrow_{\$} R_{k-32}^{p^n}, \mathbf{z}_2 \leftarrow_{\$} R_{k-32}^{p^n}$.
3. Choose $u \leftarrow_{\$} U(0, 1)$
4. Set $M \leftarrow \left(1 - \frac{64}{2k+1}\right)^{2n}$
5. If $u \leq M$ then
6.    Output a signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$
7.    Program $\mathbf{c} = H(\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu)$
8. Else return to Step 1.
```

[5] When it is queried, the oracle H is programmed to return a random $\mathbf{c} \in \{\mathbf{v} \in R_1^{p^n} : \sum_{i=1}^{n} |v_i| = 32\}$ without checking whether that value has been used before.

Key Generation Algorithm Hybrid 3

Input: Parameters (n, p, k'), where $k' = 3\alpha + 1, \alpha = \left\lceil \frac{2^{\frac{100}{n}-1}\sqrt{p}-1}{3} \right\rceil$.

Output: Secret key $(\mathbf{s}_1, \mathbf{s}_2)$ and public key $(\mathbf{a}, \mathbf{t})$.

1. Choose secret key $\mathbf{s}_1, \mathbf{s}_2 \leftarrow_{\$} R_{k'}^{p^n}$.
2. Choose $\mathbf{a} \leftarrow_{\$} R^{p^n}$.
3. Compute $\mathbf{t} \leftarrow \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2$.
4. Return secret ket $(\mathbf{s}_1, \mathbf{s}_2)$ and public key $(\mathbf{a}, \mathbf{t})$.

The following lemma shows that, if $(\mathbf{s}_1, \mathbf{s}_2)$ is randomly chosen from $R_{k'}^{p^n} \times R_{k'}^{p^n}$ in which k' is reasonably chosen, then with high probability, there exists $(\mathbf{s}'_1, \mathbf{s}'_2) \in R_{k'}^{p^n} \times R_{k'}^{p^n}$ different from $(\mathbf{s}_1, \mathbf{s}_2)$ satisfying $\mathbf{a}\mathbf{s}_1 + \mathbf{s}_2 = \mathbf{a}\mathbf{s}'_1 + \mathbf{s}'_2$.

Lemma 4. [6] *For any* $\mathbf{a} \in R^{p^n}$ *and* $(\mathbf{s}_1, \mathbf{s}_2)$ *randomly chosen from* $R_{k'}^{p^n} \times R_{k'}^{p^n}$, *in which* $k' \geq 2^{\frac{100}{n}-1}\sqrt{p}$. *Then, with probability at least* $1 - 2^{-200}$, *there exists* $(\mathbf{s}'_1, \mathbf{s}'_2)$ *different from* $(\mathbf{s}_1, \mathbf{s}_2)$ *satisfying* $\mathbf{a}\mathbf{s}_1 + \mathbf{s}_2 = \mathbf{a}\mathbf{s}'_1 + \mathbf{s}'_2$.

Proof. For any $\mathbf{a} \in R^{p^n}$, consider the map $f_{\mathbf{a}}$ defined as follows:

$$\begin{aligned} f_{\mathbf{a}} : R_{k'}^{p^n} \times R_{k'}^{p^n} &\to R^{p^n} \\ (\mathbf{s}_1, \mathbf{s}_2) &\mapsto \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2 \end{aligned}$$

We see that the cardinality of R^{p^n} is $\left|R^{p^n}\right| = p^n$ and the cardinality of $R_{k'}^{p^n} \times R_{k'}^{p^n}$ is $(2k'+1)^{2n}$. Therefore, the probability of choosing $(\mathbf{s}_1, \mathbf{s}_2) \in R_{k'}^{p^n} \times R_{k'}^{p^n}$ such that there are no collisions is at least

$$\frac{p^n}{(2k'+1)^{2n}} \leq \frac{p^n}{\left(2^{\frac{100}{n}}\sqrt{p}+1\right)^{2n}} < \frac{p^n}{2^{200}p^n} = \frac{1}{2^{200}}$$

In other words, with probability at least $1 - 2^{-200}$, there exists $(\mathbf{s}'_1, \mathbf{s}'_2)$ different from $(\mathbf{s}_1, \mathbf{s}_2)$ such that $\mathbf{a}\mathbf{s}_1 + \mathbf{s}_2 = \mathbf{a}\mathbf{s}'_1 + \mathbf{s}'_2$. □

Lemma 5. *Let $\mathcal{D}$ be a distinguisher who can querry to the random oracle H. Moreover, $\mathcal{D}$ can query the actual key generation algorithm, the actual signing algorithm, the key generation algorithm Hybrid 3 and the signing algorithm Hybrid 3. If $\mathcal{D}$, after making h queries to the random oracle H and s queries to the signing algorithm (either actual or Hybrid 3), then the advantage of $\mathcal{D}$ in distinguishing the actual key generation algorithm and the actual signing algorithm with the key generation algorithm Hybrid 3 and the signing algorithm Hybrid 3 is less than $s\,(s-1+2h)\,2^{-(n+1)}$.*

Proof. We will use the hybrid arguments to prove the Lemma. Specifically, we will use three Hybrid key generation and signing algorithms (the verification

[6] This lemma is stated based on Lemma 5.2 in [Lyu12].

algorithms are the same to the actual one, so we do not mention them here). The key generation algorithm Hybrid 1 and Hybrid 2 are similar to the actual key generation algorithm while the key generation Hybrid 3 is different from the actual one in that $(\mathbf{s}_1, \mathbf{s}_2)$ is uniformly chosen from $R_{k'}^{p^n} \times R_{k'}^{p^n}$ instead of $R_1^{p^n} \times R_1^{p^n}$. The signing algorithm Hybrid 1 is different from the actual one in that $\mathbf{c}$ is randomly chosen from $\{\mathbf{v} \in R_1^{p^n} : \sum_{i=0}^{n-1} |v_i| = 32\}$ instead of being computed by $\mathbf{c} = H(\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu)$. The signing algorithm Hybrid 2 is different from the Hybrid 1 in that in the Hybrid 2, $\mathbf{z}_1, \mathbf{z}_2$ are uniformly chosen from $R_{k-32}^{p^n}$ instead of being computed as $\mathbf{z}_1 = \mathbf{s}_1\mathbf{c} + \mathbf{y}_1, \mathbf{z}_2 = \mathbf{s}_2\mathbf{c} + \mathbf{y}_2$. The signing algorithm Hybrid 3 is similar to the Hybrid 2 one. We will show that with a suitable choice of parameters, the advantage of the distinguisher $\mathcal{D}$ in distinguishing the Hybrid algorithms is negligible.

First of all, we show that the distinguisher $\mathcal{D}$ has advantage less than $s(s - 1 + 2h)2^{-(n+1)}$ in distinguishing the actual signing algorithm with the Hybrid 1 one. The only difference between those two algorithms is that in the Hybrid 1 algorithm, the output of the random oracle H is randomly chosen from $\{\mathbf{v} \in R_1^{p^n} : \sum_{i=0}^{n-1} |v_i| = 32\}$ and then is programmed as the answer of $H(\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu) = H(\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2, \mu)$ without checking whether the hash value $(\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2, \mu)$ has been queried to H or not. Therefore, in the signing algorithm Hybrid 1, there may be the case that with the same input, two queries to H may produce two different outputs, while with the actual algorithm one gets the same output. And this is the only point that the distinguisher can distinguish between the actual signing algorithm with the Hybrid 1 algorithm. In other words, the advantage of the distinguisher in distinguishing these two algorithms is the probability that the signing algorithm Hybrid 1 produces two outputs with the same query to H.

Since $\mathcal{D}$ makes h queries to H and s queries to the signing algorithm, there are at most $s + h$ values $(\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2, \mu)$ established. Now, we show that for every call to the signing algorithm Hybrid 1, the probability of generating $\mathbf{y}_1, \mathbf{y}_2$ such that $\mathbf{a}_1\,\mathbf{y}_1 + \mathbf{y}_2$ is equal to the previous queried value is less than 2^{-n}. Given $\mathbf{t}$ arbitrarily in R^{p^n}, we have

$$\begin{aligned}
\mathsf{Pr}\left[\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2 = \mathbf{t}; \mathbf{y}_i \leftarrow_{\$} R_k^{p^n}\right] &= \mathsf{Pr}\left[\mathbf{y}_2 = (\mathbf{t} - \mathbf{a}\,\mathbf{y}_1); \mathbf{y}_i \leftarrow_{\$} R_k^{p^n}\right] \\
&\leq \max_{\mathbf{t}' \in R^{p^n}} \mathsf{Pr}\left[\mathbf{y}_2 = \mathbf{t}'; \mathbf{y}_2 \leftarrow_{\$} R_k^{p^n}\right] \\
&\leq \left(\frac{2k+1}{p}\right)^n
\end{aligned}$$

By hypothesis $k \ll p$, then

$$\mathsf{Pr}\left[\mathbf{a}\,\mathbf{y}_1 + \mathbf{y}_2 = \mathbf{t}; \mathbf{y}_1 \leftarrow_{\$} R_k^{p^n}, \mathbf{y}_2 \leftarrow_{\$} R_k^{p^n}\right] < 2^{-n}.$$

We see that the previous queried value can be generated in the random oracle H or in signing algorithm Hybrid 1. Hence, the probability of getting a collision after making s queries is at

$$\left(\binom{s}{2} + sh\right) 2^{-n} = s(s-1+2h)2^{-(n+1)}$$

Now, we need to show that outputs of the signing algorithms Hybrid 1 and Hybrid 2 are indistinguishable. Indeed, the element $\mathbf{c}$ in both algorithms are computed in a similar way. The main difference is the way $\mathbf{z}_1, \mathbf{z}_2$ are computed. In the Hybrid 1 algorithm, $\mathbf{z}_1 = \mathbf{s}_1\mathbf{c} + \mathbf{y}_1, \mathbf{z}_2 = \mathbf{s}_2\mathbf{c} + \mathbf{y}_2$ and $\mathbf{z}_1, \mathbf{z}_2$ are output only if $\mathbf{z}_1, \mathbf{z}_2 \in R_{k-32}^{p^n}$. By Lemma 3, the probability that the Hybrid 1 algorithm produces a valid signature is $\left(1 - \frac{64}{2k+1}\right)^{2n}$, whereas the Hybrid 2 chooses $\mathbf{z}_1, \mathbf{z}_2$ uniformly from $R_{k-32}^{p^n}$ and the probability of producing a valid signature is

$$\mathsf{Pr}\left[u \leq \left(1 - \frac{64}{2k+1}\right)^{2n} : u \leftarrow_{\$} U(0,1)\right] = \left(1 - \frac{64}{2k+1}\right)^{2n}.$$

Therefore, $\mathcal{D}$ cannot distinguish whether $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$ is outputted by the Hybrid 1 algorithm or the Hybrid 2 algorithm.

Next, we show that $\mathcal{D}$ cannot distinguish the public keys generated by the key generation algorithm Hybrid 3 and the Hybrid 2 algorithm. Indeed, by the hardness of $\mathrm{DCK}_{p,n}$, the distinguisher $\mathcal{D}$ cannot distinguish between the outputs of the Hybrid 2 algorithm and the uniform distribution on $R^{p^n} \times R^{p^n}$. By Lemma 1, there is a reduction from solving $\mathrm{DCK}_{p,n}$ to solving the $\text{R-SIS}_{p,n,2,(3\alpha+1)}$ decision problem. Hence $\mathcal{D}$ cannot distinguish between the outputs the Hybrid 3 algorithm with the uniform distribution on $R^{p^n} \times R^{p^n}$ (if it does then it is possible to solve the $\text{R-SIS}_{p,n,2,(3\alpha+1)}$ decision problem, from which one can solve the $\mathrm{DCK}_{p,n}$ problem). Therefore, $\mathcal{D}$ cannot distinguish between the outputs by the Hybrid 3 algorithm and the Hybrid 2 algorithm.

As a conclusion, we see that the actual key generation algorithm and the actual signing algorithm can be replaced by the corresponding Hybrid 3 algorithms. The advantage of the distinguisher in distinguishing between the actual algorithms and the Hybrid 3 algorithms is less than $s(s-1+2h)2^{-(n+1)}$. $\square$

Lemma 6. *Assume that there exists a polynomial-time forger $\mathcal{F}$ who can produce a valid signature with success probability δ after making at most s queries to the signing algorithm Hybrid 3 and at most h queries to the random oracle H. Then there exists a polynomial-time algorithm to efficiently solve the* $\text{R-SIS}_{p,n,2,\beta}$ *problem, in which $\beta = (2(k-32) + 64k')$, $k' = 3\left\lceil \frac{2^{\frac{100}{n}-1}\sqrt{p}-1}{3} \right\rceil + 1$ with success probability at least*

$$\left(\frac{1}{2} - 2^{-200}\right)\left(\delta - 2^{-200}\right)\left(\frac{\delta - 2^{-200}}{h+s} - 2^{-200}\right).$$

Proof. Denote by $D_H := \{\mathbf{v} \in R_1^{p^n} : \sum_{i=0}^{n-1} |v_i| = 32\}$ the range of the random oracle H. Given $\mathbf{a} \in R^{p^n}$, choose $\mathbf{s}_1, \mathbf{s}_2 \leftarrow_{\$} R_{k'}^{p^n}$ to be the secret key and compute

$\mathbf{t} = \mathbf{a}_1\mathbf{s}_1 + \mathbf{s}_2$. The public key is $(\mathbf{a}, \mathbf{t})$. Set $t = h + s$ to be the upper bound of the total number of times that H is queried or programmed by $\mathcal{F}$. One query to H can be made by the forger or H can be programmed by the signing algorithm Hybrid 3 when the forger requests a signature of some message. Next we choose a random coins ϕ for the forger, a random coins ψ for the signer (using Hybrid 3) and choose $\mathbf{r}_1, \ldots, \mathbf{r}_t \leftarrow_{\$} D_H$ to be the answers of the random oracle.

Now, we consider a sub-algorithm $\mathcal{A}$ with input $(\mathbf{a}, \mathbf{t}, \phi, \psi, \mathbf{r}_1, \ldots, \mathbf{r}_t)$. The algorithm $\mathcal{A}$ will run $\mathcal{F}$ by supplying to $\mathcal{F}$ the public key $(\mathbf{a}, \mathbf{t})$ and the random coins ϕ as input. Whenever $\mathcal{F}$ requests a signature of a message, $\mathcal{A}$ runs the signing algorithm Hybrid 3 and uses the random coins ψ to generate a signature. During the signing process, the random oracle H is programmed and answers the first value in $(\mathbf{r}_1, \ldots, \mathbf{r}_t)$ which has not been used before. Here $\mathcal{A}$ will save a table consisting of all queries to H, and when a same query is requested again, the random oracle will output the same answer. The forger can also make queries to the random oracle. In this case, the answer is done in the same way. Whenever $\mathcal{F}$ stops and produces a forgery (with probability δ), the algorithm $\mathcal{A}$ will take the output of $\mathcal{F}$ as its output.

With probability δ, the forger $\mathcal{F}$ will produce a message μ and a signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$ satisfying $\mathbf{z}_1, \mathbf{z}_2 \in R^{p^n}_{k-32}$ and $\mathbf{c} = H((\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}), \mu)$. Note that if the random oracle was not queried or programmed with input $\mathbf{w} = (\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c})$ then the probability that $\mathcal{F}$ produces $\mathbf{c}$ with $\mathbf{c} = H(\mathbf{w}, \mu)$ is $1/|D_H|$. Hence, with probability $1 - 1/|D_H|$, $\mathbf{c}$ has to be one of the values $\mathbf{r}_i$ with $1 \leq i \leq t$. Thus the probability that $\mathcal{F}$ succeeds in a forgery and $\mathbf{c}$ is one of the values $\mathbf{r}_i$ (with $1 \leq i \leq t$) is at least $\delta - 1/|D_H|$. Indeed, let A be the event that $\mathcal{F}$ succeeds in a forgery and B be the event that $\mathbf{c}$ is one of the values $\mathbf{r}_i$. Then we have

$$\begin{aligned}\mathsf{Pr}[A \cap B] &= \mathsf{Pr}[A] + \mathsf{Pr}[B] - \mathsf{Pr}[A \cup B] \\ &\geq \delta + 1 - \frac{1}{|D_H|} - 1 = \delta - \frac{1}{|D_H|}.\end{aligned}$$

Assume that j is the index such that $\mathbf{c} = \mathbf{r}_j$, with $1 \leq j \leq t$. Then there are two possibilities as follows:

– Either $\mathbf{r}_j$ is an answer of a query of $\mathcal{F}$ to the random oracle,
– or $\mathbf{r}_j$ is programmed in the signing process.

In the second case, assume that, when signs a message μ', the signer programs the random oracle as $H((\mathbf{a}\mathbf{z}'_1 + \mathbf{z}'_2 - \mathbf{t}\mathbf{c}), \mu') = \mathbf{c}$. If the forger produces a valid signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$ for μ then $\mu \neq \mu'$ or $(\mathbf{z}_1, \mathbf{z}_2) \neq (\mathbf{z}'_1, \mathbf{z}'_2)$ (or both are different). Because if $\mu = \mu'$ and $(\mathbf{z}_1, \mathbf{z}_2) = (\mathbf{z}'_1, \mathbf{z}'_2)$, the adversary just outputs a message with a signature that he already saw. If $\mu \neq \mu'$ then we have a collision for H, since

$$H((\mathbf{a}\mathbf{z}'_1 + \mathbf{z}'_2 - \mathbf{t}\mathbf{c}), \mu') = \mathbf{c} = H((\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}), \mu).$$

If $\mu = \mu'$ then $(\mathbf{z}_1, \mathbf{z}_2) \neq (\mathbf{z}'_1, \mathbf{z}'_2)$ and

$$H((\mathbf{a}\mathbf{z}'_1 + \mathbf{z}'_2 - \mathbf{t}\mathbf{c}), \mu) = \mathbf{c} = H((\mathbf{a}\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}), \mu).$$

Thus if $\mathbf{a}\mathbf{z}'_1 + \mathbf{z}'_2 - \mathbf{t}\mathbf{c} \neq \mathbf{a}_1\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}$ then we find a collision for H. On the other hand, if $\mathbf{a}\mathbf{z}'_1 + \mathbf{z}'_2 - \mathbf{t}\mathbf{c} = \mathbf{a}_1\,\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}$ then

$$\mathbf{a}\,(\mathbf{z}_1 - \mathbf{z}'_1) + (\mathbf{z}_2 - \mathbf{z}'_2) = 0.$$

Since $\|\mathbf{z}_1\|_\infty, \|\mathbf{z}'_2\|_\infty \leq k - 32$, one has $\|\mathbf{z}_1 - \mathbf{z}'_1\|_\infty, \|\mathbf{z}_2 - \mathbf{z}'_2\|_\infty \leq 2(k-32)$. Therefore we can solve the R-SIS$_{p,n,2,\beta}$ problem where $\beta = 2(k-32)$ with success probability at least $\delta - 1/|D_H|$.

Now we come back to the first case, i.e., $\mathbf{r}_j$ is an answer when $\mathcal{F}$ makes a query to the random oracle. In this case, first we save the signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{r}_j)$ of the forger $\mathcal{F}$ for μ. Then, we generate new random elements $\mathbf{r}'_j, \ldots, \mathbf{r}'_t \leftarrow_\$ D_H$. Next we run again the algorithm $\mathcal{A}$ with input $(\mathbf{a}, \mathbf{t}, \phi, \psi, \mathbf{r}_1, \ldots, \mathbf{r}_{j-1}, \mathbf{r}'_j, \ldots, \mathbf{r}'_t)$. By the General Forking Lemma [[BN06], Lemma 1], the probability that $\mathbf{r}'_j \neq \mathbf{r}_j$ and the forger uses $\mathbf{r}'_j$ as an answer for a query to the random oracle is at least

$$\left(\delta - \frac{1}{|D_H|}\right)\left(\frac{\delta - 1/|D_H|}{t} - \frac{1}{|D_H|}\right),$$

and so with the above probability, $\mathcal{F}$ produces a signature $\mathbf{z}'_1, \mathbf{z}'_2, \mathbf{r}'_j$ for μ and

$$\mathbf{a}\,\mathbf{z}_1 + \mathbf{a}\,\mathbf{z}_2 - \mathbf{t}\mathbf{c} = \mathbf{a}\mathbf{z}'_1 + \mathbf{a}\mathbf{z}'_2 - \mathbf{t}\mathbf{c}' \tag{2}$$

where $\mathbf{c} = \mathbf{r}_j$ and $\mathbf{c}' = \mathbf{r}'_j$. Plug $\mathbf{t} = \mathbf{a}_1\mathbf{s}_1 + \mathbf{s}_2$ into (2), we obtain

$$\mathbf{a}\,(\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1) + (\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2) = 0. \tag{3}$$

Since $\|\mathbf{z}_1\|_\infty, \|\mathbf{z}'_1\|_\infty, \|\mathbf{z}_2\|_\infty, \|\mathbf{z}'_2\|_\infty \leq k - 32$ and $\|\mathbf{s}_1\mathbf{c}\|_\infty, \|\mathbf{s}_1\mathbf{c}'\|_\infty, \|\mathbf{s}_2\mathbf{c}\|_\infty, \|\mathbf{s}_2\mathbf{c}'\|_\infty \leq 32k'$, one gets

$$\|\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1\|_\infty, \|\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2\|_\infty \leq (2(k-32) + 64k').$$

Set $\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 = \mathbf{u}_1$ and $\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2 = \mathbf{u}_2$. If $\mathbf{u}_1, \mathbf{u}_2 \neq 0$ then we can solve the R-SIS$_{p,n,2,\beta}$ problem with $\beta = (2(k-32) + 64k')$. Therefore, it suffices to show that $\mathbf{u}_1, \mathbf{u}_2 \neq 0$ with probability at least $\frac{1}{2} - 2^{-200}$. By Lemma 4, with probability at least $1 - 2^{-200}$, there exists $(\mathbf{s}'_1, \mathbf{s}'_2) \neq (\mathbf{s}_1, \mathbf{s}_2) \in R_{k'}^{p^n}$ such that $\mathbf{a}\mathbf{s}'_1 + \mathbf{s}'_2 = \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2$. If $0 = \mathbf{u}_1 = \mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1$ then $\mathbf{z}_1 - \mathbf{c}\mathbf{s}'_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}'_1 \neq 0$. Indeed, assume that

$$\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 = 0 \text{ and } \mathbf{z}_1 - \mathbf{c}\mathbf{s}'_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}'_1 = 0$$

Then $\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 = \mathbf{z}_1 - \mathbf{c}\mathbf{s}'_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}'_1$. Hence

$$(\mathbf{c} - \mathbf{c}')\,(\mathbf{s}_1 - \mathbf{s}'_1) = 0. \tag{4}$$

Because $\|\mathbf{c}\|_1, \|\mathbf{c}'\|_1 \leq 32$ and $\|\mathbf{s}_1\|_\infty, \|\mathbf{s}'_1\|_\infty \leq k'$, we have $\|(\mathbf{c} - \mathbf{c}')\,(\mathbf{s}_1 - \mathbf{s}'_1)\|_\infty \leq 256k'$. By the choice of parameters, $256k' < p$. Thus if $(\mathbf{c} - \mathbf{c}')\,(\mathbf{s}_1 - \mathbf{s}'_1) = 0$ over $R^{p^n} = \mathbb{Z}_p[x]/\langle x^n + 1\rangle$ then $(\mathbf{c} - \mathbf{c}')\,(\mathbf{s}_1 - \mathbf{s}'_1) = 0$ over $\mathbb{Z}[x]/\langle x^n + 1\rangle$. Since n is a power of 2, $x^n + 1$ is irreducible in $\mathbb{Z}[x]$. Hence $\mathbb{Z}[x]/\langle x^n + 1\rangle$ is an integral domain. Then, $(\mathbf{c} - \mathbf{c}')\,(\mathbf{s}_1 - \mathbf{s}'_1) = 0$ which implies $\mathbf{c} - \mathbf{c}' = 0$ or

$\mathbf{s}_1 - \mathbf{s}'_1 = 0$. So $\mathbf{c} \neq \mathbf{c}'$ and hence $\mathbf{s}_1 = \mathbf{s}'_1$ (which contradicts to $\mathbf{s}_1 \neq \mathbf{s}'_1$ above). Similarly, if $0 = \mathbf{u}_2 = \mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2$ then $\mathbf{z}_2 - \mathbf{c}\mathbf{s}'_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}'_2 \neq 0$. On the other hand, at the beginning, we do not know whether $\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 = 0$ and $\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2 = 0$ or $\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 \neq 0$ and $\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2 \neq 0$. Therefore, the probability that $\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 \neq 0$ and $\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2 \neq 0$ is exactly the probability that the secret key is $(\mathbf{s}'_1, \mathbf{s}'_2)$. Denote by E the event that there exists a related key $(\mathbf{s}'_1, \mathbf{s}'_2)$ and by F the event that the secret key is $(\mathbf{s}'_1, \mathbf{s}'_2)$. Then the probability that $\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 \neq 0$ and $\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2 \neq 0$ is

$$\begin{aligned}\mathsf{Pr}[E \cap F] &= \mathsf{Pr}[E] + \mathsf{Pr}[F] - \mathsf{Pr}[E \cup F] \\ &\geq 1 - 2^{-200} + \frac{1}{2} - 1 = \frac{1}{2} - 2^{-200}.\end{aligned}$$

Therefore, the probability that $\mathbf{z}_1 - \mathbf{c}\mathbf{s}_1 - \mathbf{z}'_1 + \mathbf{c}'\mathbf{s}_1 \neq 0$ and $\mathbf{z}_2 - \mathbf{c}\mathbf{s}_2 - \mathbf{z}'_2 + \mathbf{c}'\mathbf{s}_2 \neq 0$ is at least $\frac{1}{2} - 2^{-200}$. Moreover, since $\mathcal{A}$ does not use the secret keys as input and does not use those secret keys for signing, the forger cannot know which secret key used in signing is $(\mathbf{s}_1, \mathbf{s}_2)$ or $(\mathbf{s}'_1, \mathbf{s}'_2)$.

Hence in case $\mathbf{r}_j$ is an answer for a query of $\mathcal{F}$ to the random oracle, we can solve the R-SIS$_{p,n,2,\beta}$ problem where $\beta = (2(k-32) + 64k')$ with probability at least

$$\left(\frac{1}{2} - 2^{-200}\right)\left(\delta - \frac{1}{|D_H|}\right)\left(\frac{\delta - 1/|D_H|}{h+s} - \frac{1}{|D_H|}\right).$$

Since $\beta = 2(k-32)$ in the first case is less than $\beta = (2(k-32) + 64k')$ in the later case and the success probability of solving the R-SIS$_{p,n,2,\beta}$ problem in the first case is greater than the success probability of solving the R-SIS$_{p,n,2,\beta}$ problem in the second case, the success probability will be the small one. Therefore, if there exists a polynomial-time forger $\mathcal{F}$ with success probability δ after making s queries to the signing algorithm Hybrid 3 and h queries to the random oracle H, then there exists a polynomial-time algorithm to solve the R-SIS$_{p,n,2,\beta}$ problem, where $\beta = (2(k-32) + 64k')$, with success probability at least

$$\left(\frac{1}{2} - 2^{-200}\right)\left(\delta - 2^{-200}\right)\left(\frac{\delta - 2^{-200}}{h+s} - 2^{-200}\right).$$

Since $|D_H| = 2^{32}\binom{n}{32}$. For $n = 512$, we have $|D_H| \approx 2^{200}$ and for $n = 1024$, we have $|D_H| \approx 2^{233}$. □

5 Conclusion

In this paper, we present a full security proof for the GLP signature scheme. Concretely, we show that the GLP signature scheme is EU-CMA secure in the random oracle model, assuming the hardness of the DCK$_{p,n}$ problem and the R-SIS$_{q,n,2,\beta}$ problem. Based on the statements and arguments in [Lyu12, Lyu08], we gave the Lemmas 2, 4, Lemmas 5, 6 and Theorem 1. The optimal version of the GLP digital signature scheme can be proved in the same way as this paper.

Acknowledgements. The authors are grateful to Duong Hoang Dung and Trieu Quang Phong for helpful comments and discussions on drafts of this paper.

References

[BG14] Bai, S., Galbraith, S.D.: An improved compression technique for signatures based on learning with errors. In: Benaloh, J. (ed.) CT-RSA 2014. LNCS, vol. 8366, pp. 28–47. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04852-9_2

[BN06] Bellare, M., Neven, G.: New multi-signatures and a general forking lemma. Full version of this paper (2006). http://www.cs.ucsd.edu/users/mihir

[DDLL13] Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 40–56. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_3

[DLL+17] Ducas, L., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium: digital signatures from module lattices. Technical report, Cryptology ePrint Archive, Report 2017/633 (2017)

[GLP12] Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: Practical lattice-based cryptography: a signature scheme for embedded systems. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 530–547. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33027-8_31

[GLP15] Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: Lattice-based signatures: optimization and implementation on reconfigurable hardware. IEEE Trans. Comput. **64**(7), 1954–1967 (2015)

[LM06] Lyubashevsky, V., Micciancio, D.: Generalized compact Knapsacks are collision resistant. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 144–155. Springer, Heidelberg (2006). https://doi.org/10.1007/11787006_13

[Lyu08] Lyubashevsky, V.: Towards Practical Lattice-Based Cryptography. University of California, San Diego (2008)

[Lyu12] Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_43

[Sho99] Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Rev. **41**(2), 303–332 (1999)