Post-Quantum and UC-secure Oblivious Transfer from SPHF with Grey Zone

Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Baptiste Cottier, David Pointcheval

Post-Quantum and UC-secure Oblivious Transfer from SPHF with Grey Zone

Slim Bettaieb[1], Loïc Bidoux[2], Olivier Blazy[3], Baptiste Cottier[4], and David Pointcheval[4]

[1] Worldline, France
[2] Technology Innovation Institute, United Arab Emirates
[3] Ecole Polytechnique, IPP, France
[4] DIENS, CNRS, ENS/PSL, Inria, Paris, France

Abstract. Oblivious Transfer (OT) is a major primitive for secure multiparty computation. Indeed, combined with symmetric primitives along with garbled circuits, it allows any secure function evaluation between two parties. In this paper, we propose a new approach to build OT protocols. Interestingly, our new paradigm features a security analysis in the Universal Composability (UC) framework and may be instantiated from post-quantum primitives. In order to do so, we define a new primitive named Smooth Projective Hash Function with Grey Zone (SPHFwGZ) which can be seen as a relaxation of the classical Smooth Projective Hash Functions, with a subset of the words for which one cannot claim correctness nor smoothness: the grey zone. As a concrete application, we provide two instantiations of SPHFwGZ respectively based on the Diffie-Hellman and the Learning With Errors (LWE) problems. Hence, we propose a quantum-resistant OT protocol with UC-security in the random oracle model.

1 Introduction

Smooth Projective Hash Function (SPHF), or Hash Proof System as introduced by Cramer and Shoup in [11], is a cryptographic primitive initially designed to provide IND-CCA encryption schemes. Over the years, SPHFs have been used for many applications such as Password-Authenticated Key Exchange [14,1,18,2], Zero-Knowledge Proofs [16,3] or Witness Encryption [12]. Since their introduction, SPHFs have been developed over classical hard problems such as discrete logarithm or factorization. However, post-quantum cryptography does not seem to be as easily compliant with SPHF. In [17], Katz *et al.* introduced *Approximate Smooth Projective Hash Functions*. The correctness property of an SPHF claims that the hash value and the projective hash value are required to be equal on words in an NP-language, when knowing a witness, while the smoothness property expects them to be independent when no witness exists. Approximate SPHF uses an approximate correctness, that allows those values to be close, relatively to a given distance. Furthermore, languages relying on code-based or lattice-based ciphertexts result in a gap between the set of valid ciphertexts of a given value μ, and the values that decrypt into μ. As mentioned in [5], an adversary could maliciously generate one of those ciphertexts and open the door for practical attacks. The presence of this gap can also be problematic when expecting to work in the Universal Composability framework [9].

Related Works. In this section, we focus on SPHF-related previous constructions. In code-based cryptography, the first proposition was made by Persichetti in [21]. The SPHF proposed there uses a weaker smoothness definition, called universality. Strictly speaking, this is not a drawback as we can transform an SPHF with universality property to a word-dependent SPHF with smoothness property. However, the main issue with this candidate is that the proof is done on random keys, rather than the whole keys. This has for consequence that an adversary can exploit some well-chosen keys resulting in a failure of the proof. A second construction was designed in [5]. As said before, when working with lattices and codes, languages based on ciphertexts present a grey zone. In this work, Bettaieb *et al.* withdraw this gap using a zero-knowledge proof asserting if two different ciphertexts of the same message are valid, reducing the SPHF on the set of valid ciphertexts, resulting in the first *gapless* post-quantum SPHF. A solution based on codes is also given in [24], but their solution offers an Approximate SPHF with computational smoothness, while real SPHF expects statistical/perfect smoothness. In lattice-based cryptography, the first construction was given in [17] where Katz *et al.* introduced the notion of Approximate SPHF. Their language not being exactly defined as the valid LWE-ciphertexts, decoding procedure was expensive, as detailed in [4]. This latter article, motivated by this issue, offers the first non-approximated SPHF based on lattices later used with the framework from [6] in [7] to build a Post-quantum UC-secure Oblivious transfer. Their construction, in the standard model, is UC-secure against adaptive corruptions but lacks of efficiency. While the two previous constructions of SPHF are in the standard model, Zha *et al.* [25] propose a SPHF requiring access to a random oracle. Indeed, their language relies on simulation-sound non-interactive zero-knowledge proofs, that we are not able to construct efficiently without random oracles.

Contribution. As mentioned above, a gap appears when working with cryptography based on lattices or codes. Rather than withdraw this gap as done in [5], we focus on the requirements needed in order to tame this gap, with an additional notion of *Decomposition Intractability* when trying to exploit this gap. Therefore, we introduce *Smooth Projective Hash Functions with Grey Zone* (SPHFwGZ) as an SPHF with the *Decomposition Intractability* property: we will require a language $\mathcal{L}$, hard to decide, as for any non-trivial SPHF, but also with additional intractability for finding two *complementary* words in $\mathcal{L}$ or the gap. As an application of SPHFwGZ, we show that one can design an Oblivious Transfer from any SPHF with Grey Zone on languages of ciphertexts for homomorphic encryption, where the security relies on the semantic security.

We provide two concrete instantiations of SPHFwGZ: the first one relies on the Diffie-Hellman Problem and the ElGamal cryptosystem. As no decryption failure occurs with the ElGamal cryptosystem, the grey zone is empty and the decomposition intractability is obvious. One can note that the resulting SPH-FwGZ is *de facto* an SPHF. The idea behind this instantiation is, on the one hand, to familiarise the reader with our construction, and on the other hand, to point out the fact that the construction is also available from any classical SPHF. A second instantiation is based on lattices and more precisely from the *Learning with Errors* problem. This allows to underline the genericity of our framework.

2 Preliminaries

Oblivious Transfer. Oblivious transfer, introduced by Rabin [22], involves a sender with input two messages m_0, m_1 and a receiver with input a selection bit b so that the latter receives m_b and nothing else, while the former does not learn anything. It provides sender-privacy (no information leakage about m_{1-b}) and receiver-privacy (no information leakage about b).

Universal Composability. Universal Composability is a security model introduced by Canetti [9] taking into account the whole environment (i.e. all exterior interactions) of the execution. Concretely, if a protocol is proven to be universally composable (or *UC-secure*), it can be used concurrently with other protocols without compromising the global protocol security. Proving universally composable security is done thanks to the real world / ideal world paradigm. In the ideal world, we consider an access to a trusted third party. A protocol Π is *UC-secure*, if, for all environment $\mathcal{E}$, there exists a simulator $\mathcal{S}$ such that the execution of the protocol Π with adversary $\mathcal{A}$ in the real world, is indistinguishable with the execution of the functionality $\mathcal{F}$ with simulator $\mathcal{S}$ in the ideal world.

Smooth Projective Hash Functions. Introduced in 2002 [11], Smooth Projective Hash Functions (SPHF), also known as Hash Proof System (HPS), initially aim to build the first public key encryption scheme secure against chosen ciphertext attacks. Nowadays, SPHF are mainly used for Honest Verifier Zero Knowledge Proofs or Witness Encryption. Such functions work on NP-languages $\mathcal{L} \subset \mathcal{X}$, defined by a binary relation $\mathcal{R}$ such that for any word $x \in \mathcal{X}$, $x \in \mathcal{L}$ if and only if there exists a witness w such that $\mathcal{R}(x, w) = 1$. Then, an SPHF defined on $\mathcal{L} \subset \mathcal{X}$ with values in $\mathcal{V}$ is defined by five algorithms:

- $\mathsf{Setup}(1^\kappa)$: Generates the parameters param from κ, the security parameter where param includes a description of $\mathcal{L}$, a language in $\mathcal{X}$;
- $\mathsf{HashKG}(\mathsf{param})$: Generates a random hash key hk;
- $\mathsf{ProjKG}(\mathsf{hk})$: Derives the projection key hp;
- $\mathsf{Hash}(\mathsf{hk}, x)$: Returns the hash value $H_{\mathsf{hk}} \in \mathcal{V}$ associated to the word x;
- $\mathsf{ProjHash}(\mathsf{hp}, x, w)$: Returns $H_{\mathsf{hp}} \in \mathcal{V}$ using a witness w linked to the word x.

Those algorithms should ensure two requirements:
- **Correctness:** For any $x \in \mathcal{L}$, with witness w, $H_{\mathsf{hk}} = H_{\mathsf{hp}}$ under the condition that $\mathcal{R}(x, w) = 1$;

– **Smoothness:** For any $x \in \mathcal{X} \backslash \mathcal{L}$, the distributions of $(\mathsf{hp}, H_{\mathsf{hk}})$ and $(\mathsf{hp}, v \leftarrow \mathcal{V})$ are indistinguishable.

The aforementioned definition of smoothness was introduced by Cramer and Shoup in [11]. Two variants of this definition have later been proposed: The first variation has been provided by Gennaro and Lindell in [14], leading to the notion of GL-SPHF. The only difference with the definition of Cramer and Shoup (recalled above) is that the projection key hp may depend on the word w of the language. The second variant, introduced by Katz and Vaikuntanathan in [17] considers the ability for an attacker to maliciously generate the word w after seeing the projection key hp. In KV-SPHF, the projection depends only on the hashing key and ensures the smoothness even if the word w is chosen after having seen the projection key. GL-SPHF will be enough for our applications, with word-dependent projection keys, as the word will be known beforehand.

3 Smooth Projective Hash Functions with Grey Zone

Our first contribution is the formalization of Smooth Projective Hash Functions with a Grey Zone (SPHFwGZ) which is a relaxation of the classical SPHF in which one cannot claim correctness nor smoothness for a subset of the words. Later, we will provide a quantum-resistant SPHFwGZ based on lattices. With this new definition, we will have two disjoint languages $\mathcal{L}, \mathcal{L}' \subset \mathcal{X}$ that will not necessarily partition the superset $\mathcal{X}$: the remaining subset $\mathcal{X} \backslash (\mathcal{L} \cup \mathcal{L}')$ will be the grey zone.

3.1 Basic Definitions

Let us describe our relaxation of *Smooth Projective Hash Function* from [10] to encompass a *Grey Zone*. An SPHFwGZ is defined with a tuple of algorithms:

– $\mathsf{Setup}(1^\kappa)$: Generate the parameters param from κ, the security parameter, or an explicit random tape (σ, ρ) in $\mathcal{S}_0 \times \mathcal{R}_0$. param includes a description of $\mathcal{L}, \mathcal{L}', \mathcal{X}$, where $\mathcal{L} \cup \mathcal{L}' \subset \mathcal{X}$ and $\mathcal{L} \cap \mathcal{L}' = \emptyset$, and $\mathcal{L}$ is a language hard to decide in $\mathcal{X}$;
– $\mathsf{HashKG}(\mathsf{param})$: Generates a random hash key hk;
– $\mathsf{ProjKG}(\mathsf{hk}, x)$: Derives the projection key hp (it may need x as input);
– $\mathsf{Hash}(\mathsf{hk}, x)$: Returns the hash value $H_{\mathsf{hk}} \in \mathcal{V}$, where $\mathcal{V}$ is the set of hash values, associated to the word x;
– $\mathsf{ProjHash}(\mathsf{hp}, x, w)$: Returns $H_{\mathsf{hp}} \in \mathcal{V}$ using a witness w linked to the word x.

As the classical SPHF, our SPHFwGZ verifies the following statistical properties, for any setup execution that provides param, defining $\mathcal{L}, \mathcal{L}', \mathcal{X}$:

– **Correctness:** For any $x \in \mathcal{L}$, $H_{\mathsf{hk}} = H_{\mathsf{hp}}$, where $\mathsf{hk} \leftarrow \mathsf{HashKG}(\mathsf{param})$, $\mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, x)$, $H_{\mathsf{hk}} \leftarrow \mathsf{Hash}(\mathsf{hk}, x)$, and $H_{\mathsf{hp}} \leftarrow \mathsf{ProjHash}(\mathsf{hp}, x, w)$ for the witness w of $x \in \mathcal{L}$;

– **Smoothness:** For any $x \in \mathcal{L}'$, the distributions of $(\mathsf{hp}, H_{\mathsf{hk}})$ and (hp, v) are indistinguishable, where $\mathsf{hk} \leftarrow \mathsf{HashKG}(\mathsf{param})$, $\mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, x)$, $H_{\mathsf{hk}} \leftarrow \mathsf{Hash}(\mathsf{hk}, x)$, and $v \stackrel{\$}{\leftarrow} \mathcal{V}$;

The algorithms and properties described above are the basic algorithms for SPH-FwGZ. For a later use, we need to define several additional properties.

3.2 Word Indistinguishability and Trapdoor

First, we assume languages $\mathcal{L}$ and $\mathcal{L}'$ in $\mathcal{X}$ are defined according to a random tape (σ, ρ) sampled in a set $\mathcal{S}_0 \times \mathcal{R}_0$ (i.e. from $\mathsf{param} \leftarrow \mathsf{Setup}(\sigma, \rho)$). The samplable set $\mathcal{S}_0$ is defined together with its twin set $\mathcal{S}_1$ such that when $\sigma \in \mathcal{S}_1$, and $\mathsf{param} \leftarrow \mathsf{Setup}(\sigma, \rho)$, there exists a trapdoor td_σ that allows to test if a given word $x \in \mathcal{X}$ is in $\mathcal{L}'$ or not. We then also need the following algorithms:

– $\mathsf{WordGen}\mathcal{L}(\mathsf{param})$: Samples and returns $x \stackrel{\$}{\leftarrow} \mathcal{L}$, together with its witness w;
– $\mathsf{WordTest}(\mathsf{td}_\sigma, x)$, using the trapdoor td_σ, tests if $x \in \mathcal{L}'$.

As we assumed $\mathcal{L}$ to be a hard subset of $\mathcal{X}$ when $\sigma \in \mathcal{S}_0$, we have the **Word-Indistinguishability Property**: An adversary can not distinguish between random words in $\mathcal{L}$ and random words in $\mathcal{X}$, for any $\sigma \in \mathcal{S}_0$, with more than a negligible advantage.

The string σ can be seen as a CRS, that admits a trapdoor when sampled from $\mathcal{S}_1$. The normal use is with $\sigma \stackrel{\$}{\leftarrow} \mathcal{S}_0$, which needs to be efficiently samplable. When $\sigma \in \mathcal{S}_1$, the trapdoor td_σ must be easy to compute from σ.

3.3 Decomposition Intractability and Trapdoor

We also define the alternate sets $\mathcal{R}_1$ and $\mathcal{R}'_1$ for $\mathcal{R}_0$. During normal use, ρ is sampled from $\mathcal{R}_0$, which needs to be efficiently samplable. When $\rho \in \mathcal{R}_1$, and $\mathsf{param} \leftarrow \mathsf{Setup}(\sigma, \rho)$, there exists a trapdoor $\mathsf{td}_\rho = (x, x', w, w')$, that must be easy to compute from ρ. When $\rho \in \mathcal{R}'_1$, and $\mathsf{param} \leftarrow \mathsf{Setup}(\sigma, \rho)$, there exists a trapdoor $\mathsf{td}_\rho = (x, x')$, that must be easy to compute from ρ. Let us define the complement algorithm, for any $\rho \in \mathcal{R}_0 \cup \mathcal{R}_1 \cup \mathcal{R}'_1$:

– $\mathsf{ComplementWord}(\mathsf{param}, \rho, x)$: from any word $x \in \mathcal{X}$, it outputs x';

From this complement algorithm, we expect the following statistical property, for any $\sigma \in \mathcal{S}_0 \cup \mathcal{S}_1$ but $\rho \in \mathcal{R}_0$:

– **Complement:** for any $x \in \mathcal{X}$, if $x' \leftarrow \mathsf{ComplementWord}(\mathsf{param}, \rho, x)$, then $x = \mathsf{ComplementWord}(\mathsf{param}, \rho, x')$;

But we also need a computational assumption: the **Decomposition Intractability**, which states that no adversary can generate, with non-negligible probability, for random $(\sigma, \rho) \stackrel{\$}{\leftarrow} \mathcal{S}_1 \times \mathcal{R}_0$, two words $x, y \notin \mathcal{L}'$ such that $y = \mathsf{ComplementWord}(\mathsf{param}, \rho, x)$, and so even with the trapdoor td_σ.

On the other hand, when $\rho \in \mathcal{R}_1$, the trapdoor $\mathsf{td}_\rho = (x, x', w, w')$ satisfies x and x' are uniformly random in $\mathcal{L}$ with witnesses w, w', and $x' =$

$\mathsf{ComplementWord}(\mathsf{param}, \rho, x)$. And when $\rho \in \mathcal{R}'_1$, the trapdoor $\mathsf{td}_\rho = (x, x')$ satisfies x and x' are uniformly random in $\mathcal{L}'$, and $x' = \mathsf{ComplementWord}(\mathsf{param}, \rho, x)$.

Again, the string ρ can be seen as a CRS, that admits a trapdoor when sampled from $\mathcal{R}_1$ or $\mathcal{R}'_1$. The normal use is with $\rho \xleftarrow{\$} \mathcal{R}_0$, which needs to be efficiently samplable. When $\rho \in \mathcal{R}_1$ or $\rho \in \mathcal{R}'_1$, the trapdoor td_ρ must be easy to compute from ρ.

Eventually, for the security proof to go through, we will make use of the **CRS Indistinguishability**: An adversary can not distinguish between $\mathcal{R}_0$, $\mathcal{R}_1$ and $\mathcal{R}'_1$, and between $\mathcal{S}_0$ and $\mathcal{S}_1$, with more than a negligible advantage.

Note that we independently consider the choices between $\mathcal{S}_0$ and $\mathcal{S}_1$ and between $\mathcal{R}_0$, $\mathcal{R}_1$ and $\mathcal{R}'_1$, but the latter choice could depend on the former choice. So the global CRS is the pair $\mathsf{crs} = (\sigma, \rho)$.

4 Oblivious Transfer from SPHFwGZ

In this section we first present our construction of Oblivious Transfers based on Smooth Projective Hash Functions with Grey Zone, and then provide a security proof of our Oblivious Transfer in the Universal Composability framework

4.1 Construction of Oblivious Transfer

Our Oblivious Transfer uses a $\mathsf{crs} = (\sigma, \rho) \in \mathcal{S}_0 \times \mathcal{R}_0$ as defined above, where we assume $\mathcal{S}_0 \times \mathcal{R}_0 \approx \mathcal{S}_1 \times \mathcal{R}_0 \approx \mathcal{S}_0 \times \mathcal{R}_1 \approx \mathcal{S}_0 \times \mathcal{R}'_1$. We describe in Figure 1 the OT protocol $\Phi_{\mathsf{OT}}^{\mathsf{SPHFwGZ}}$.

Receiver with input $b \in \{0,1\}$ for $\mathtt{sid}$ and $\mathsf{crs} = (\sigma, \rho)$ from $\mathcal{F}_{\mathsf{CRS}}$		Sender with input $m_0, m_1 \in \mathcal{M}$ for $\mathtt{sid}$ and $\mathsf{crs} = (\sigma, \rho)$ from $\mathcal{F}_{\mathsf{CRS}}$
$\mathsf{param} \leftarrow \mathsf{Setup}(\sigma, \rho)$		$\mathsf{param} \leftarrow \mathsf{Setup}(\sigma, \rho)$
$(x_b, w) \leftarrow \mathsf{WordGen}\mathcal{L}(\mathsf{param})$		
$x_{1-b} \leftarrow \mathsf{ComplementWord}(\rho, x_b)$	$\xrightarrow{x_0}$	$x_1 \leftarrow \mathsf{ComplementWord}(\rho, x_0)$
		for i in $\{0,1\}$:
		$\mathsf{hk}_i \leftarrow \mathsf{HashKG}(\mathsf{param})$
		$\mathsf{hp}_i \leftarrow \mathsf{ProjKG}(\mathsf{hk}_i, x_i)$
		$H_i \leftarrow \mathsf{Hash}(\mathsf{hk}_i, x_i)$
$H' \leftarrow \mathsf{ProjHash}(c_{b,1}, x_b, w)$	$\xleftarrow{(c_0, c_1)}$	$c_i = (H_i \oplus m_i, \mathsf{hp}_i)$
$m = H' \oplus c_{b,0}$		

Fig. 1. General description of the protocol $\Phi_{\mathsf{OT}}^{\mathsf{SPHFwGZ}}$

The protocol $\Phi_{\mathsf{OT}}^{\mathsf{SPHFwGZ}}$ provides **Correctness**. Indeed, with the honest generation $(x, w) \leftarrow \mathsf{WordGen}\mathcal{L}(\mathsf{param})$ we have $c = (H \oplus m, \mathsf{hp})$. Then, $m = H \oplus m \oplus \mathsf{ProjHash}(c_1, x, w)$ if and only if $H = \mathsf{ProjHash}(c_1, x, w)$ which is ensured due to the correctness property of the $\mathsf{SPHFwGZ}$. Moreover, the **Complement** property ensures the value x_1 computed by the sender is always the same as the value x_1 computed by the receiver.

We now prove the privacy in the Universal Composability framework.

4.2 Security Analysis

Our Oblivious Transfer protocol will be proven in the CRS-hybrid model (as in [20]), with the functionality $\mathcal{F}_{\mathsf{CRS}}$, where the two players get the same random crs from the $\mathtt{sid}$. In practice, as we assumed $\mathcal{S}_0$ and $\mathcal{R}_0$ efficiently samplable, (σ, ρ) can be derived from $\mathcal{H}(\mathtt{sid})$. As no rewind is required, the proof remains valid in case the CRS is generated using quantum-accessible random oracles [8]. Then, we recall below the ideal functionality $\mathcal{F}_{\mathsf{OT}}$ for a secure oblivious transfer, where there are two first messages from the sender with (m_0, m_1) and from the receiver with b, to initialize the process, and the final request message by the sender that decides when the receiver can get m_b:

$\mathcal{F}_{\mathsf{OT}}$ interacts with a sender S and a receiver R:

- Upon receiving a message ($\mathtt{sid}$,$\mathtt{sender}$, m_0, m_1) from S, store ($\mathtt{sid}, m_0, m_1$);
- Upon receiving a message ($\mathtt{sid}$,$\mathtt{receiver}$, b) from R, store ($\mathtt{sid}, b$);
- Upon receiving a message ($\mathtt{sid}$,$\mathtt{answer}$) from the adversary, check if both records ($\mathtt{sid}, m_0, m_1$) and ($\mathtt{sid}, b$) exist for $\mathtt{sid}$. If yes, send ($\mathtt{sid}$,m_b) to R, and $\mathtt{sid}$ to the adversary and halt. If not, send nothing but continue running.

Ideal Functionality $\mathcal{F}_{\mathsf{OT}}$

Theorem 1. *The protocol $\Phi_{\mathsf{OT}}^{\mathsf{SPHFwGZ}}$ UC-realizes $\mathcal{F}_{\mathsf{OT}}$ in the $\mathcal{F}_{\mathsf{CRS}}$-hybrid model in the static-corruption setting, from any SPHFwGZ.*

We stress that we consider static corruptions only, where the corrupted players are known when each protocol execution starts.

Game G_0. This is the real game, where $\mathcal{F}_{\mathsf{CRS}}$ samples crs in $\mathcal{S}_0 \times \mathcal{R}_0$.

Game G_1. In this game, the simulator $\mathcal{S}$ simulates itself the sampling of $\mathsf{crs} = (\sigma, \rho) \stackrel{\$}{\leftarrow} \mathcal{S}_0 \times \mathcal{R}_0$, and generates correctly every flow from the honest players, as they would do themselves, knowing the inputs (m_0, m_1) and b sent by the environment to the sender and the receiver, respectively.

Game G_2. In this game, we deal with **corrupted receivers**. Instead of sampling $\mathsf{crs} = (\sigma, \rho) \stackrel{\$}{\leftarrow} \mathcal{S}_0 \times \mathcal{R}_0$, the simulator $\mathcal{S}$ samples $\mathsf{crs} = (\sigma, \rho) \stackrel{\$}{\leftarrow} \mathcal{S}_1 \times \mathcal{R}_0$, and therefore with the trapdoor td_σ. This game is indistinguishable from the previous one due to the *CRS Indistinguishability*.

Game G_3. In this game, the simulator $\mathcal{S}$ uses the trapdoor td_σ to get $t_i = \mathsf{WordTest}(x_i, \mathsf{td}_\sigma)$ for $i \in \{0, 1\}$. If $t_0 = t_1 = 0$ (none of the words are in $\mathcal{L}'$), $\mathcal{S}$ aborts. This game is indistinguishable from the previous one, under the *Decomposition Intractability*, as $(\sigma, \rho) \in \mathcal{S}_1 \times \mathcal{R}_0$.

Game G_4. If $t_0 = t_1 = 0$, we still abort. If $t_0 = t_1 = 1$ we set $b = 0$, otherwise, we set b such that $t_b = 0$. Next, the simulator $\mathcal{S}$ proceeds on m_b with x_b and on a random message with x_{1-b}. Under the smoothness of the SPHFwGZ, as $x_{1-b} \in \mathcal{L}'$, and the *One-Time Pad Semantic Security*, this game is statistically indistinguishable from the previous one.

Game G_5. In this game, we deal with **corrupted senders**. Instead of sampling $\mathsf{crs} = (\sigma, \rho) \stackrel{\$}{\leftarrow} \mathcal{S}_0 \times \mathcal{R}_0$, the simulator $\mathcal{S}$ samples $\mathsf{crs} = (\sigma, \rho) \stackrel{\$}{\leftarrow} \mathcal{S}_0 \times \mathcal{R}_1$, and therefore with the trapdoor $\mathsf{td}_\rho = (x, x', w, w')$. This game is indistinguishable from the previous one due to the *CRS indistinguishability*.

Game G_6. In this game, the simulator $\mathcal{S}$ respectively sets (x_0, w_0, x_1, w_1) as (x, w, x', w') from td_ρ. It can then retrieve both m_0 and m_1. This game is indistinguishable from the previous one due to the *Word Indistinguishability*, and the uniform distribution of the trapdoor.

Game G_7. We now deal with **honest players**. Instead of sampling $\mathsf{crs} = (\sigma, \rho) \xleftarrow{\$} \mathcal{S}_0 \times \mathcal{R}_0$, the simulator $\mathcal{S}$ samples $\mathsf{crs} = (\sigma, \rho) \xleftarrow{\$} \mathcal{S}_0 \times \mathcal{R}'_1$, and therefore with the trapdoor $\mathsf{td}_\rho = (x, x')$, and simulates the flows with random $m_0, m_1 \xleftarrow{\$} \mathcal{M}$ and random $b \xleftarrow{\$} \{0, 1\}$. Under the *CRS Indistinguishability* and the smoothness of the $\mathsf{SPHFwGZ}$, as both $x, x' \in \mathcal{L}'$, coupled with the *One-Time Pad Semantic Security*, this game is indistinguishable from the previous one.

Game G_8. This is the ideal game We can now make use of the functionality $\mathcal{F}_{\mathsf{OT}}$ which leads to the following simulator:

- If no participant is corrupted, one uses $\mathsf{crs} \xleftarrow{\$} \mathcal{S}_0 \times \mathcal{R}'_1$, and the simulator $\mathcal{S}$ simply uses random inputs for the sender and the receiver;
- If the receiver is corrupted, one uses $\mathsf{crs} \xleftarrow{\$} \mathcal{S}_1 \times \mathcal{R}_0$, and the simulator $\mathcal{S}$ extracts b using the trapdoor td_σ, and sends $(\mathtt{sid}, \mathsf{receiver}, b)$ to $\mathcal{F}_{\mathsf{OT}}$;
- If the sender is corrupted, one uses $\mathsf{crs} \xleftarrow{\$} \mathcal{S}_0 \times \mathcal{R}_1$, and the simulator $\mathcal{S}$ extracts m_0, m_1 using the trapdoor td_ρ, and sends $(\mathtt{sid}, \mathsf{sender}, m_0, m_1)$ to $\mathcal{F}_{\mathsf{OT}}$;
- The adversary sends $(\mathtt{sid}, \mathsf{answer})$ when it decides to deliver the result to the receiver.

4.3 Noisy Homomorphic Encryption Setup

We now define a general setup leading to an instantiation of our Oblivious Transfer from many (possibly with decryption failure and possibly amplified, as shown with our lattice-based instantiation) Homomorphic Encryption with group law $*$ on plaintexts and $\circledast$ on the ciphertexts.

We consider an encryption scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with possible decryption failures. Thus, we set $\mathcal{X}$ as the ciphertext space of Π, whereas

$$\mathcal{L} = \{\mathsf{Encrypt}(\mathsf{pk}, 0; r)\} \subset \mathcal{X} \text{ and } \mathcal{L}' = \{c \in \mathcal{X}, \mathsf{Decrypt}(\mathsf{sk}, c) \neq 0\} \subset \mathcal{X}.$$

Sets $\mathcal{S}_0$ and $\mathcal{S}_1$ can both be seen as public keys pk generated from $\mathsf{KeyGen}(1^\kappa)$ except that when $\sigma \in \mathcal{S}_1$, the secret key sk is known and defines the trapdoor td_σ. Hence, σ (which defines the public key pk) defines the sets $\mathcal{L}$ and $\mathcal{L}'$ in $\mathcal{X}$. On the other hand, we can define $\mathcal{R}_0 = \mathcal{X}$, the set of all the ciphertexts, or a superset, with uniform distribution; $\mathcal{R}_1 = \{c_0 \circledast c_1\}$, for two ciphertexts c_0, c_1 in $\mathcal{L}$, following the distribution of the encryption algorithm, on plaintext 0, and according the distribution of the randomness r_0, r_1, which allows to define the trapdoor td_ρ as (c_0, c_1, r_0, r_1); $\mathcal{R}'_1 = \{c_0 \circledast c_1\}$, for two ciphertexts c_0, c_1 in $\mathcal{L}'$, following the distribution of the encryption algorithm, on non-zero plaintexts, which allows to define the trapdoor td_ρ as (c_0, c_1). The setup defined above verifies both basic assumptions required to make the Oblivious Transfer Universally Composable:

- *CRS Indistinguishability*: Under the *semantic security* of the encryption scheme Π, $\mathcal{L}$, $\mathcal{L}'$, and $\mathcal{X}$ are indistinguishable. The homomorphic property implies that $\{x \circledast x' | (x, x') \in \mathcal{X}^2\} = \mathcal{X}$. As a consequence, we have indistinguishability between $\mathcal{R}_0 = \mathcal{X} = \{x \circledast x' | (x, x') \in \mathcal{X}^2\}$, $\mathcal{R}_1 = \{x \circledast x' | (x, x') \in \mathcal{L}^2\}$, and $\mathcal{R}'_1 = \{x \circledast x' | (x, x') \in \mathcal{L}'^2\}$. Furthermore, as $\mathcal{S}_0 = \mathcal{S}_1$, they are perfectly indistinguishable;
- *Word Indistinguishability*: Under the *semantic security* of the encryption scheme Π, one can not distinguish between $c_0 \in \mathcal{L}$, an encryption of 0 and $c_1 \in \mathcal{X}$, an encryption of a random value.
- **Complement:** for any $x \in \mathcal{X}$, $x' \leftarrow \mathsf{ComplementWord}(\mathsf{param}, \rho, x) = \rho \circledast x^{-1}$, hence $\mathsf{ComplementWord}(\mathsf{param}, \rho, x') = \rho \circledast (\rho \circledast x^{-1})^{-1} = x$;

Additional properties will depend on concrete instantiations.

5 Concrete Instantiations of SPHFwGZ

We now provide two concrete instantiations of SPHFwGZ based on the Diffie-Hellman and Learning With Errors problems. As both constructions rely on an Homomorphic Encryption scheme, we can already consider the basic properties shown in Section 4.3.

5.1 Instantiation from the Diffie-Hellman Problem

In this section, we focus on elliptic curve based cryptography, using the Decisional Diffie-Hellman assumption in a prime-order group.

Definition 1 (Decisional Diffie-Hellman (DDH)). *In a group $\mathbb{G}$ of prime order p, the Decisional Diffie-Hellman problem consists in, given g^a and g^b, distinguishing g^{ab} from g^c, for $a, b, c \stackrel{\$}{\leftarrow} \mathbb{Z}_p$.*

The Decisional Diffie-Hellman assumption states that the aforementioned Decision Diffie-Hellman problem is hard to solve, with non-negligible advantage in polynomial time.

ElGamal Encryption. As expected above, we need an IND-CPA (a.k.a. with semantic security) encryption scheme, with homomorphism. We use the ElGamal encryption scheme [13] in a group $\mathbb{G} = \langle g \rangle$ of prime order p, defined by the Setup algorithm:

- $\mathsf{KeyGen}(1^\kappa)$: picks $\beta \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, and sets $\mathsf{pk} = h = g^\beta$, $\mathsf{sk} = \beta$.
- $\mathsf{Encrypt}(\mathsf{pk} = h = g^\beta,\ M \in \mathbb{G})$ encrypts the message M under the public key pk as follows: Pick $r \stackrel{\$}{\leftarrow} \mathbb{Z}_p$; Output the ciphertext: $c = (g^r, h^r \cdot M)$;
- $\mathsf{Decrypt}(\mathsf{sk}, c = (c_0, c_1))$ decrypts the ciphertext c using the decryption key sk as follows: $M = c_1 / c_0^{\mathsf{sk}}$.

Theorem 2. *The above ElGamal encryption scheme is IND-CPA under the Decisional Diffie-Hellman assumption.*

SPHFwGZ from ElGamal Encryption[5] . From the IND-CPA ElGamal encryption scheme $\mathsf{EG} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$, in a group $\mathbb{G}$, denoted multiplicatively, of prime order p, with generator g.

We set $\mathcal{S}_0 = \mathcal{S}_1 = \{\sigma = h = g^{\mathsf{td}_\sigma}; \mathsf{td}_\sigma \stackrel{\$}{\leftarrow} \mathbb{Z}_p\}$. Then, $\mathcal{R}_0$ is defined as $\mathbb{G}^2 = \{\rho = (\hat{g}, \hat{h}) \stackrel{\$}{\leftarrow} \mathbb{G}^2\}$, $\mathcal{R}_1$ as $\{\rho = (\hat{g} \leftarrow g^{r_0} \cdot g^{r_1}, \hat{h} \leftarrow h^{r_0} \cdot h^{r_1}); (r_0, r_1) \leftarrow\!\!\$\, \mathbb{Z}_p\}$ and $\mathcal{R}'_1$ as $\{c_0 \circledast c_1; (c_0, c_1) \in \mathbb{G}^{2\times 2}\}$. The crs is set as (σ, ρ). One can note that witnesses only exist when $\rho \in \mathcal{R}_1$ or $\rho \in \mathcal{R}'_1$, then $\mathsf{td}_\rho = (c_0 = (g^{r_0}, h^{r_0})$, $c_1 = (g^{r_1}, h^{r_1}), r_0, r_1)$ or $\mathsf{td}_\rho = (c_0, c_1)$ respectively. One can note c_0 and c_1 are encryptions of $M = g^0$, with respective randomness r_0 and r_1. Moreover, while td_σ always exists, it is not necessarily known.

From the above generic construction, we have $\mathcal{X} = \{(g^r, h^r \cdot M), M \in \mathbb{G}\} = \mathbb{G}^2$ and $\mathcal{L} = \{(g^r, h^r)\}$, which are indistinguishable under the Decisional Diffie-Hellman assumption. With $\mathsf{param} = (g, \sigma = h)$, which determines all the sets (specified by the Setup algorithm), we can define:

- $\mathsf{hk} = \mathsf{HashKG}(\mathsf{param}) = (\alpha, \beta) \stackrel{\$}{\leftarrow} \mathbb{Z}_p^2$;
- $\mathsf{hp} = \mathsf{ProjKG}(\mathsf{hk}) = g^\alpha h^\beta$;
- $H = \mathsf{Hash}(\mathsf{hk}, x = (u, v)) = u^\alpha v^\beta \in \mathbb{G}$;
- $H' = \mathsf{ProjHash}(\mathsf{hp}, x, w = r) = \mathsf{hp}^r$, if $x = (g^r, h^r) \in \mathcal{L}$.
- $x' = (u', v') = \mathsf{ComplementWord}(\rho = (\hat{g}, \hat{h}), x = (u, v)) = (\hat{g} \cdot u^{-1}, \hat{h} \cdot v^{-1})$

This is a word-independent SPHFwGZ. And we can show the expected properties:

- **Correctness:** When $x = (u, v) = (g^r, h^r) \in \mathcal{L}$, with witness r, $H = u^\alpha v^\beta = (g^\alpha h^\beta)^r = \mathsf{hp}^r = H'$;
- **Smoothness:** When $x = (u, v) = (g^r, h^{r'}) \notin \mathcal{L}$, then $r' = r + r''$ with $r'' \neq 0$: $H = u^\alpha v^\beta = (g^\alpha h^\beta)^r \times g^{r''\beta} = \mathsf{hp}^r \times g^{r''\beta} = H' \times g^{r''\beta}$. But β is perfectly hidden in hp, and $g^{r''\beta}$ is perfectly unpredictable;
- **Decomposition Intractability:** In ElGamal encryption there is no decryption failure: all the ciphertexts can be covered by the encryption algorithm, and the decryption perfectly inverts the encryption process. So we have $\mathcal{L}' = \mathcal{X} \backslash \mathcal{L}$. A random ciphertext ρ encrypts an $M \neq 1$ with overwhelming probability. Then, when it encrypts $M \neq 1$, from the homomorphic property, this is impossible to have two encryptions of 1 whose product is ρ. Hence, the *decomposition intractability* is statistical: the probability of existence of the decomposition is bounded by $1/p$, on ρ, even knowing the decryption key, and thus the trapdoor td_σ.

5.2 Instantiation from the Learning With Errors Problem

In this section, we focus on lattice-based cryptography. We are going to show how to instantiate the various required components from LWE:

Definition 2 (Shortest Independent Vectors Problem (SIVP$_\gamma$)). *The approximation version SIVP$_\gamma$ is the approximation version of SIVP with factor λ. Given a basis $\boldsymbol{B}$ of an n-dimensional lattice, find a set of n linearly*

[5] Note that this construction exactly corresponds to the one from [10]

independent vectors $v_1, \ldots, v_n \in \mathcal{L}(\boldsymbol{B})$ *such that* $\|v_i\| \leq \gamma(n) \cdot \lambda_n(\boldsymbol{B})$. *for all* $1 \leq i \leq n$. *The approximation factor* γ *is typically a polynomial in* n*, the non approximated version assumes* $\gamma = 1$.

Definition 3 (Learning With Errors (LWE)). *Let* $q \geq 2$, *and* χ *be a distribution over* $\mathbb{Z}$. *The Learning With Errors problem* $LWE_{\chi,q}$ *consists in, given a polynomial number of samples, distinguishing the two following distributions:*

- $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$, *where* $\mathbf{a}$ *is uniform in* $\mathbb{Z}_q^n$, $e \leftarrow \chi$, *and* $\mathbf{s} \in \mathbb{Z}_q^n$ *is a fixed secret chosen uniformly, and where* $\langle \mathbf{a}, \mathbf{s} \rangle$ *denotes the standard inner product.*
- $(\mathbf{a}, b)$, *where* $\mathbf{a}$ *is uniform in* $\mathbb{Z}_q^n$, *and* b *is uniform in* $\mathbb{Z}_q$.

Regev Encryption. Regev [23] showed that for $\chi = D_{\mathbb{Z},\sigma}$, a Gaussian centered distribution in $\mathbb{Z}$ for any standard deviation $\sigma \geq 2\sqrt{n}$, and q such that $q/\sigma = \mathsf{poly}(n)$, $\mathrm{LWE}_{\chi,q}$ is at least as hard as solving worst-case SIVP for polynomial approximation factors, which is assumed to be hard to solve, even for quantum computers.

Trapdoor for LWE. Throughout this paper, we will use the trapdoors introduced in [19] to build our public matrix $\mathbf{A}$. Define $g_{\mathbf{A}}(\mathbf{s}, \mathbf{e}) = \mathbf{A}\mathbf{s} + \mathbf{e}$, the gadget matrix $\mathbf{G}$ as $\mathbf{G}^t = \mathbf{I}_n \otimes \mathbf{g}^t$, where $\mathbf{g}^t = [1, 2, \ldots, 2^k]$ and $k = \lceil \log q \rceil - 1$, and let $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$ be invertible. The notation $[\mathbf{A} \,|\, \mathbf{B}]$ is for horizontal concatenation, while $[\mathbf{A} \,;\, \mathbf{B}]$ is for vertical concatenation.

Lemma 1 ([19, Theorems 5.1 and 5.4]). *There exist two PPT algorithms* $\mathsf{TrapGen}$ *and* $g^{-1}_{(\cdot)}$ *with the following properties assuming* $q \geq 2$ *and* $m \geq \Theta(n \log q)$*:*

- $\mathsf{TrapGen}(1^n, 1^m, q)$ *outputs* $(\mathbf{T}, \mathbf{A}_0)$, *where the distribution of the matrix* $\mathbf{A}_0$ *is at negligible statistical distance from uniform in* $\mathbb{Z}_q^{m \times n}$, *and such that* $\mathbf{T}\mathbf{A}_0 = \mathbf{0}$, *where* $s_1(\mathbf{T}) \leq O(\sqrt{m})$ *and where* $s_1(\mathbf{T})$ *is the operator norm of* $\mathbf{T}$, *which is defined as* $\max_{\mathbf{x} \neq 0} \|\mathbf{T}\mathbf{x}\| / \|\mathbf{x}\|$.[6]
- *Let* $(\mathbf{T}, \mathbf{A}_0) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$. *Let* $\mathbf{A}_{\mathbf{H}} = \mathbf{A}_0 + [\mathbf{0} \,;\, \mathbf{G}\mathbf{H}]$ *for some invertible matrix* $\mathbf{H}$ *called a* tag. *Then, we have* $\mathbf{T}\mathbf{A}_{\mathbf{H}} = \mathbf{G}\mathbf{H}$. *Furthermore, if* $\mathbf{x} \in \mathbb{Z}_q^m$ *can be written as* $\mathbf{A}_{\mathbf{H}}\mathbf{s} + \mathbf{e}$, *with* $\mathbf{s} \in \mathbb{Z}_q^n$ *and* $\mathbf{e} \in \mathbb{Z}_q^m$ *where* $\|\mathbf{e}\| \leq B' := q/\Theta(\sqrt{m})$, *then* $g^{-1}_{\mathbf{A}_{\mathbf{H}}}(\mathbf{T}, \mathbf{x}, \mathbf{H})$ *outputs* $(\mathbf{s}, \mathbf{e})$.

More precisely, to sample $(\mathbf{T}, \mathbf{A}_0)$ with $\mathsf{TrapGen}$, we sample a uniform $\bar{\mathbf{A}} \in \mathbb{Z}_q^{\bar{m} \times n}$ where $\bar{m} = m - nk = \Theta(n \log q)$, and some $\mathbf{R} \leftarrow \mathcal{D}^{nk \times \bar{m}}$, where the distribution $\mathcal{D}^{nk \times \bar{m}}$ assigns probability $1/2$ to 0, and $1/4$ to ± 1. We output $\mathbf{T} = [-\mathbf{R} \,|\, \mathbf{I}_{nk}]$ along with $\mathbf{A}_0 = [\bar{\mathbf{A}} \,;\, \mathbf{R}\bar{\mathbf{A}}]$. Then, given a tag $\mathbf{H}$, with $\mathbf{A}_{\mathbf{H}} = \mathbf{A}_0 + [\mathbf{0} \,;\, \mathbf{G}\mathbf{H}]$, we have: $\mathbf{T}\mathbf{A}_{\mathbf{H}} = \mathbf{G}\mathbf{H}$.

We will only consider a fixed tag $\mathbf{H} = \mathbf{I}$, for the Micciancio-Peikert encryption [19]. Our construction only requires CPA encryption so we don't need several tags, but we need to be able to reject improperly computed ciphertexts, and the gadget matrix is here, to allow this extra control during the decryption.

[6] The bound on $\mathbf{s}_1(\mathbf{T})$ holds except with probability at most 2^{-n} in the original construction, but we assume the algorithm restarts if it does not hold.

LWE Encryption à la Micciancio-Peikert. For this scheme, we assume q to be an odd prime. We set an encoding function for messages $\mathsf{Encode}(\mu \in \{0,1\}) = \mu \cdot (0, \ldots 0, \lceil q/2 \rceil)^t$. Note that $2 \cdot \mathsf{Encode}(\mu) = (0, \ldots, 0, \mu)^t \bmod q$, as $\lceil q/2 \rceil$ is the inverse of 2 mod q, for such an odd q.

Let $(\mathbf{T}, \mathbf{A}_0) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$. The public encryption key is $\mathsf{pk} = \mathbf{A}_0$, and the secret decryption key is $\mathsf{sk} = \mathbf{T}$.

- $\mathsf{Encrypt}(\mathsf{pk} = \mathbf{A}_0,\ \mu \in \{0,1\})$ encrypts the message μ under the public key pk as follows: Let $\mathbf{A} = \mathbf{A}_0 + [\mathbf{0}\,;\,\mathbf{G}]$. Pick $\mathbf{s} \in \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},t}^m$ where $t = \sigma\sqrt{m} \cdot \omega(\sqrt{\log n})$. Restart if $\|\mathbf{e}\| > B$, where $B \coloneqq 2t\sqrt{m}$.[7] Output the ciphertext:
$$\mathbf{c} = \mathbf{A}\mathbf{s} + \mathbf{e} + \mathsf{Encode}(\mu) \bmod q\ .$$
- $\mathsf{Decrypt}(\mathsf{sk} = \mathbf{T},\ \mathbf{c} \in \mathbb{Z}_q^m)$ decrypts the ciphertext $\mathbf{c}$ using the decryption key sk as follows: With $B'' \coloneqq q/2\Theta(\sqrt{m})$, output
$$\begin{cases} \mu & \text{if } g_{\mathbf{A}}^{-1}(\mathbf{T}, 2\mathbf{c}, \mathbf{I}) = (2\mathbf{s}, 2\mathbf{e} + (0, \ldots, 0, \mu)) \\ & \qquad \text{where } \mathbf{s} \in \mathbb{Z}_q^n, \mathbf{e} \in \mathbb{Z}^m \text{ and } \|\mathbf{e}\| \le B'' \\ \perp & \text{otherwise.}^{8} \end{cases} ,$$

Noting $\Lambda(A) = \{\mathbf{A}\mathbf{s} | \mathbf{s} \in \mathbb{Z}_q^n\}$, honestly generated ciphertext $\mathbf{c}$ are such that $d(\mathbf{c} - \mathsf{Encode}(\mu), \Lambda(\mathbf{A})) \le B$, while the decryption procedure is guaranteed not to return μ as soon as $d(\mathbf{c} - \mathsf{Encode}(\mu), \Lambda(\mathbf{A})) > B''$. From the decryption procedure, we have:

$$\mu' \coloneqq \mathsf{Decrypt}(\mathbf{T}, \mathbf{c}) \neq \perp \iff d(\mathbf{c} - \mathsf{Encode}(\mu'), \Lambda(\mathbf{A})) < B''\ .$$

Suppose that $m \ge \Theta(n \log q)$. The scheme is correct as long as $B \le B''$, or equivalently $2\sigma m^{3/2} \cdot \omega(\sqrt{\log n}) \le q$.

Theorem 3. *Assume $m \ge \Theta(n \log q)$. The above scheme is IND-CPA assuming the hardness of the $LWE_{\chi,q}$ problem for $\chi = D_{\mathbb{Z},\sigma}$.*

Furthermore, this encryption scheme is homomorphic for plaintexts in $(\mathbb{Z}_2, +)$, and ciphertexts in $\mathbb{Z}_q^m$ with component-wise addition.

Bit-SPHFwGZ from LWE Encryption Scheme. We consider, an LWE encryption scheme defined with a superpolynomial modulus. More precisely, we set $m = n\log(q), t = \sqrt{mn}.\omega(\sqrt{\log(n)})$, $k = \Theta(n), s \ge \Theta(\sqrt{n}) \wedge s/q = \mathsf{negl}(n), s = \Omega(mk^2q^{2/3})$. We also set R to be a *probabilistic* rounding function from $[0,1]$ to $\{0,1\}$, such that $R(x) = 1$ with probability $0.5 \cdot \cos(\frac{2\pi x}{q})$ and 0 otherwise.

We set $\mathcal{S}_0 = \mathcal{S}_1 = \{\sigma = \mathbf{A} = \mathbf{A}_0 + [\mathbf{0}\,;\,\mathbf{G}] | (\mathbf{T}, \mathbf{A}_0) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)\}$, td_σ being $\mathbf{T}$. Then, $\mathcal{R}_0$ is defined as $\{\rho = \mathbf{v} \in \mathbb{Z}_q^m\}$ and $\mathcal{R}_1$ is the set composed

[7] This happens only with exponentially small probability $2^{-\Theta(n)}$.

[8] Note that the inversion algorithm $g_{(\cdot)}^{-1}$ can succeed even if $\|\mathbf{e}\| > B''/2$, depending on the randomness of the trapdoor. It is crucial to reject decryption nevertheless when $\|\mathbf{e}\| > B''$ to ensure security.

of all the sums of two honest encryptions of 0, in other words $\{\rho = \mathbf{A}(\mathbf{s} + \mathbf{s}') + \mathbf{e} + \mathbf{e}' \bmod q \ |\mathbf{s}, \mathbf{s}' \in \mathbb{Z}_q^n, \mathbf{e}, \mathbf{e}' \leftarrow D_{\mathbb{Z},t}^m \wedge \|\mathbf{e}\| \leq B \wedge \|\mathbf{e}'\| \leq B\}$ with $\mathsf{td}_\rho = (\mathbf{As} + \mathbf{e}, \mathbf{As}' + \mathbf{e}', (\mathbf{s}, \mathbf{e}), (\mathbf{s}', \mathbf{e}'))$.

With $\mathcal{X}_{bit} = \{\mathbf{c} \stackrel{\$}{\leftarrow} \mathbb{Z}_q^m\}$, $\mathcal{L}_{bit} = \{\mathbf{c} | \exists \mathbf{s}, \mathbf{e}, \mathbf{c} = \mathsf{Encrypt}(\mathbf{A}_0, 0; \mathbf{s}, \mathbf{e})\}$ defined following the description above, and $\mathcal{L}'_{bit} = \{\mathbf{c} \in \mathcal{X}_{bit} | \mathsf{Decrypt}(\mathbf{T}, \mathbf{c}) \neq 0\}$. Hence $\mathcal{R}'_1 = \{\mathbf{c}_1 + \mathbf{c}_2; (\mathbf{c}_1, \mathbf{c}_2) \in {\mathcal{L}'_{bit}}^2\}$ with $\mathsf{td}_\rho = (\mathbf{c}_1, \mathbf{c}_2)$. Note that $\mathbf{s}$ could be enough as a witness for $\mathbf{c} = \mathbf{As} + \mathbf{e} \in \mathcal{L}_{bit}$, as one can check $\mathbf{e} = \mathbf{c} - \mathbf{As}$ is small enough. This defines the Setup algorithm, and we have:

Definition 4 (Bit-SPHFwGZ over Micciancio-Peikert like Ciphertexts [4]). *For $k = \Theta(n)$, and picking $s \geq \Theta(\sqrt{n})$, and $s = \Omega(mk^2 q^{2/3})$, we can define:*

- $\mathit{HashKG}(\mathsf{param}) = \mathit{hk} = \mathbf{h} \stackrel{\$}{\leftarrow} D_{\mathbb{Z},s}^m$
- $\mathit{ProjKG}(\mathit{hk}) = \mathit{hp} = \mathbf{A}^t \mathbf{h}$
- $\mathit{Hash}(\mathit{hk}, \mathbf{c}) = R(\langle \mathit{hk}, \mathbf{c} \rangle) = R(\langle \mathbf{h}, \mathbf{c} \rangle) \in \{0, 1\}$
- $\mathit{ProjHash}(\mathit{hp}, \mathbf{c}, w = \mathbf{s}) = R(\langle \mathit{hp}, \mathbf{s} \rangle) = R(\langle \mathbf{A}^t \mathbf{h}, \mathbf{s} \rangle)$

For a word $\mathbf{c} = \mathbf{As} + \mathbf{e}$ in the language $\mathcal{L}_{bit}$, $\langle \mathbf{h}, \mathbf{c} \rangle = \mathbf{h}^t \mathbf{As} + \mathbf{h}^t \mathbf{e} = \langle \mathbf{A}^t \mathbf{h}, \mathbf{s} \rangle + \mathbf{h}^t \mathbf{e}$. And by construction $\mathbf{h}^t \mathbf{e}$ is small. The choice of the rounding function $R(x)$, characterized by a coin flip where the outcome 1 is weighted by $0.5 \cdot \cos(\frac{2\pi x}{q})$, is such that it allows canceling out this small noise most of the time, while providing smoothness for words outside the language (ensuring that $R(\langle \mathsf{hk}, \mathbf{c} \rangle)$ is random when given only hp)

It was shown in [4], that for this choice of random function, such bit-SPHFwGZ achieves negligible-universality, thanks to the rounding function, but $(3/4 + o(1))$-correctness for the chosen set of parameters.

Full-Fledged SPHFwGZ from LWE. The previous construction has limitations as it is neither perfectly correct, nor smooth, we need to apply a transformation to reach those goals. This transformation is explained below, first informally, then in more details:

- It is a bit-function meaning the final hash value lives in $\{0, 1\}$, while one needs a larger mask. To solve this issue, one has to run it in parallel a linear number of times, to have an output string long enough.
- The correctness is imperfect. The output bit only matches with probability $3/4 + o(1)$. As such, applications running $\mathsf{Encrypt}(\mathsf{pk}, m; r)$ should encryption a redundant version of m, with an error-correcting code, $\mathsf{ECC}(m)$. Such transformation makes the SPHF word-dependent (i.e. the projection key is dependent on the user/receiver input), however in our scenario, such a word-dependent function is enough.

More formally, given a word $\mathbf{c} \in \mathcal{X}_{bit}$, for any $\ell = \Omega(n)$ an error-correcting code ECC capable of correcting $\ell/4$ errors, then, we can define the SPHF as:

- $\mathsf{SETUP}(1^\kappa)$: Outputs the result from $\mathsf{Setup}(1^\kappa)$
- $\mathsf{HASHKG}(\mathsf{param})$: Picks a random values $K \leftarrow \{0, 1\}^\kappa$, and $\forall i \in [\ell]$, gets $\mathsf{hk}_i = \mathsf{HashKG}(\mathsf{param})$, and set $\mathsf{HK} = (\{\mathsf{hk}_i\}, K)$;

- $\mathsf{ProjKG}(\mathsf{HK}, \mathbf{c})$: $\forall i \in [\ell]$, gets $\mathsf{hp}_i = \mathsf{ProjKG}(\mathsf{hk}_i), H_i = \mathsf{Hash}(\mathsf{hk}_i, \mathbf{c})$. It then computes $T = \mathsf{ECC}(K) \oplus S$ where $S = (H_i)_{i \in [\ell]}$, and outputs $\mathsf{HP} = ((\mathsf{hp}_i)_{i \in [\ell]}, T)$;
- $\mathsf{HASH}(\mathsf{HK}, \mathbf{c})$: Returns K, from HK;
- $\mathsf{PROJHASH}(\mathsf{HP}, \mathbf{c}, w = \mathbf{s})$: $\forall i \in [\ell]$, computes $H'_i = \mathsf{ProjHash}(\mathsf{hp}_i, \mathbf{c}, \mathbf{s})$. Then computes $S' = (H'_i)_{i \in [\ell]}$, and finally $K' = \mathsf{ECC}^{-1}(T \oplus S')$.

Such transformation allows to achieve *smoothness* which can be proven with an hybrid argument, handling intermediate distributions where the first H_i values are random. The *correctness* is simply inherited from the correcting-code capacity, while the number of errors to be corrected can be estimated thanks to the Hoeffding's bound [15]. We can guarantee the expected properties:

- **Correctness:** When $x = \mathbf{c} \in \mathcal{L}_{bit}$, with the above conversion, we have $K = K'$ with overwhelming probability, thanks to the error-correcting code;
- **Smoothness:** When $x = \mathbf{c} \notin \mathcal{L}_{bit}$, then the value K is random from an adversary point of view, as the parallelization technique allows to transform the negligible-universality to a classical smoothness (at the cost of a word-dependent SPHF);
- **Half Decomposition Intractability:** A random vector ρ should not be split into two ciphertexts that could be decrypted to 0, or at least not too often. We first deal with *half* decomposition intractability, when at most half of the random vectors can be split. To get a lower-bound on the number of vectors like such ρ, we can remark that a vector verifies this property as soon as $d(\rho, \Lambda(\mathbf{A}))$ is greater than 2 times the decryption bound.
  This is the reason, why we took a conservative value $B'' = B'/2$ in the encryption compared to classical Micciancio-Peikert encryption. By halving the decryption radius, we ensured that adding two elements that still decrypt within this bound will fall on classically decryptable ciphertexts. As such, at least half the elements cannot be reached (those that classically decrypted to 1). Hence, $\Pr_{\rho \in \mathbb{Z}_q^m}[\exists \mathbf{c}, \mathbf{d} | \rho = \mathbf{c} + \mathbf{d} \land \mathsf{Decrypt}(\mathsf{sk}, \mathbf{c}) = \mathsf{Decrypt}(\mathsf{sk}, \mathbf{d}) = 0] \leq 1/2$. This is a statistical bound, that holds even when knowing the decryption key.

Another amplification is required to make *full-fledged decomposition intractability*, by working on ciphertexts $(\mathbf{c}_j)_{j \in [k]}$, with k parallel executions of the SPH-FwGZ, with a final XOR of all the outputs, so that the smoothness for one word is enough to get the smoothness for the vector of words, but the correctness on all the words leads to the global correctness. With $\mathcal{X} = (\mathcal{X}_{bit})^k$, the acceptable languages, for correctness and smoothness respectively are then:

$$\mathcal{L} = (\mathcal{L}_{bit})^k = \{(\mathbf{c}_j)_{j \in [k]} | (\forall j \in [k]), \exists (\mathbf{s}_j, \mathbf{e}_j), \mathbf{c}_j = \mathsf{Encrypt}(\mathbf{A}_0, 0; \mathbf{s}_j, \mathbf{e_j})\} \subset \mathcal{X}$$
$$\mathcal{L}' = (\mathcal{L}'_{bit})^k = \{(\mathbf{c}_j)_{j \in [k]} | (\exists j \in [k]), \mathsf{Decrypt}(\mathbf{T}, \mathbf{c}_j) \neq 0\} \subset \mathcal{X}$$

Then, for random $(\rho_j)_{j \in [k]} \xleftarrow{\$} \mathcal{X}$, a decomposition would be a list of pairs $(\mathbf{c}_j, \mathbf{d}_j)_{j \in [k]} \in (\mathcal{X} \times \mathcal{X})$ such that for all j, $\rho_j = \mathbf{c}_j + \mathbf{d}_j$ and $\mathsf{Decrypt}(\mathbf{T}, \mathbf{c}_j) = \mathsf{Decrypt}(\mathbf{T}, \mathbf{d}_j) = 0$, which only exists with probability less than $1/2^k$. We thus have achieved all the security properties required for our applications.

6 Conclusion

In this paper, we introduced *Smooth Projective Hash Functions with Grey Zone*, that generalize SPHF to language subjected to gaps, thanks to the *Decomposition Intractability* property. This is enough to get Oblivious Transfer proven secure in the Universally Composable model. As such a primitive can be obtained from the LWE problem, we can then obtain a UC-secure post-quantum Oblivious Transfer.

Acknowledgments

This work was supported in part by the French ANR Project Crypto4Graph-AI.

References

1. Abdalla, M., Chevalier, C., Pointcheval, D.: Smooth projective hashing for conditionally extractable commitments. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 671–689. Springer, Heidelberg (Aug 2009). doi:10.1007/978-3-642-03356-8_39
2. Ben Hamouda, F., Blazy, O., Chevalier, C., Pointcheval, D., Vergnaud, D.: Efficient UC-secure authenticated key-exchange for algebraic languages. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 272–291. Springer, Heidelberg (Feb / Mar 2013). doi:10.1007/978-3-642-36362-7_18
3. Benhamouda, F., Blazy, O., Chevalier, C., Pointcheval, D., Vergnaud, D.: New techniques for SPHFs and efficient one-round PAKE protocols. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 449–475. Springer, Heidelberg (Aug 2013). doi:10.1007/978-3-642-40041-4_25
4. Benhamouda, F., Blazy, O., Ducas, L., Quach, W.: Hash proof systems over lattices revisited. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part II. LNCS, vol. 10770, pp. 644–674. Springer, Heidelberg (Mar 2018). doi:10.1007/978-3-319-76581-5_22
5. Bettaieb, S., Bidoux, L., Blazy, O., Connan, Y., Gaborit, P.: A gapless code-based hash proof system based on rqc and its applications. Designs, Codes and Cryptography **90**(12), 3011–3044 (2022). doi:10.1007/s10623-022-01075-7, `https://doi.org/10.1007/s10623-022-01075-7`
6. Blazy, O., Chevalier, C.: Generic construction of uc-secure oblivious transfer. In: International Conference on Applied Cryptography and Network Security. pp. 65–86. Springer (2015)
7. Blazy, O., Chevalier, C., Vu, Q.H.: Post-quantum uc-secure oblivious transfer in the standard model with adaptive corruptions. In: Proceedings of the 14th International Conference on Availability, Reliability and Security. ARES '19, Association for Computing Machinery, New York, NY, USA (2019). doi:10.1145/3339252.3339280, `https://doi.org/10.1145/3339252.3339280`
8. 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.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 41–69. Springer, Heidelberg (Dec 2011). doi:10.1007/978-3-642-25385-0_3

9. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd FOCS. pp. 136–145. IEEE Computer Society Press (Oct 2001). doi:10.1109/SFCS.2001.959888
10. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (Aug 1998). doi:10.1007/BFb0055717
11. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (Apr / May 2002). doi:10.1007/3-540-46035-7_4
12. Derler, D., Slamanig, D.: Practical witness encryption for algebraic languages and how to reply an unknown whistleblower. Cryptology ePrint Archive, Report 2015/1073 (2015), `https://eprint.iacr.org/2015/1073`
13. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory **31**, 469–472 (1985)
14. Gennaro, R., Lindell, Y.: A framework for password-based authenticated key exchange. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 524–543. Springer, Heidelberg (May 2003). doi:10.1007/3-540-39200-9_33, `https://eprint.iacr.org/2003/032.ps.gz`
15. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association **58**(301), 13–30 (1963). doi:10.1080/01621459.1963.10500830
16. Jutla, C.S., Roy, A.: Relatively-sound NIZKs and password-based key-exchange. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 485–503. Springer, Heidelberg (May 2012). doi:10.1007/978-3-642-30057-8_29
17. Katz, J., Vaikuntanathan, V.: Smooth projective hashing and password-based authenticated key exchange from lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 636–652. Springer, Heidelberg (Dec 2009). doi:10.1007/978-3-642-10366-7_37
18. Katz, J., Vaikuntanathan, V.: Round-optimal password-based authenticated key exchange. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 293–310. Springer, Heidelberg (Mar 2011). doi:10.1007/978-3-642-19571-6_18
19. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (Apr 2012). doi:10.1007/978-3-642-29011-4_41
20. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (Aug 2008). doi:10.1007/978-3-540-85174-5_31
21. Persichetti, E.: Secure and anonymous hybrid encryption from coding theory. In: Gaborit, P. (ed.) Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013. pp. 174–187. Springer, Heidelberg (Jun 2013). doi:10.1007/978-3-642-38616-9_12
22. Rabin, M.O.: How to exchange secrets with oblivious transfer. Technical Report TR-81, Aiken Computation Laboratory, Harvard University (1981)
23. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC. pp. 84–93. ACM Press (May 2005). doi:10.1145/1060590.1060603
24. Shooshtari, M.K., Aref, M.R.: Smooth projective hash function from codes and its applications. IEEE Transactions on Services Computing pp. 1–1 (2021). doi:10.1109/TSC.2021.3100323

25. Zhang, J., Yu, Y.: Two-round PAKE from approximate SPH and instantiations from lattices. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part III. LNCS, vol. 10626, pp. 37–67. Springer, Heidelberg (Dec 2017). doi:10.1007/978-3-319-70700-6_2