Post-Quantum Oblivious Transfer from Smooth Projective Hash Functions with Grey Zone

Slim Bettaieb\textsuperscript{1}, Loïc Bidoux\textsuperscript{2}, Olivier Blazy\textsuperscript{3}, Baptiste Cottier\textsuperscript{4}, and David Pointcheval\textsuperscript{4}

\textsuperscript{1} Worldline, France
\textsuperscript{2} Technology Innovation Institute, United Arab Emirates
\textsuperscript{3} Ecole Polytechnique, IPP, France
\textsuperscript{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 [CS02], 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 [GL03,ACP09,KV11,BBC\textsuperscript{13a}], Zero-Knowledge Proofs [JR12,BBC\textsuperscript{13b}] or Witness Encryption [DS15]. 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 [KV09], 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 $\mu$, and the values that decrypt into $\mu$. As mentioned in [BBB+21], 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 [Can01].

**Related Works.** In code-based cryptography, the first proposition was made by Persichetti in [Per13]. 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 [BBB+21]. 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 [SA21], 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 [KV09] 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 [BBDQ18]. This latter article, motivated by this issue, offers the first non-approximated SPHF based on lattices. While the two previous constructions are in the standard model, Zha et al. [ZY17] 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 [BBB+21], 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 Hash Projective 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

2.1 Oblivious Transfer

Oblivious transfer, introduced by Rabin [Rab81], 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 \( b \)).

2.2 Universal Composability

Universal Composability is a security model 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 \( \Pi \) is UC-secure, if, for all environment \( \mathcal{E} \), there exists a simulator \( \mathcal{S} \) such that the execution of the protocol \( \Pi \) 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.

2.3 Smooth Projective Hash Functions

Introduced in 2002 [CS02], 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:

- **Setup(\( \kappa \))**: Generates the parameters \( \text{param} \) from \( \kappa \), the security parameter. \( \text{param} \) includes a description of \( \mathcal{L} \), a language in \( \mathcal{X} \);
- **HashKG(\( \text{param} \))**: Generates a random hash key \( \text{hk} \);
- **ProjKG(\( \text{hk} \))**: Derives the projection key \( \text{hp} \);
- **Hash(\( \text{hk}, x \))**: Returns the hash value \( H_{\text{hk}} \in \mathcal{V} \) associated to the word \( x \);
- **ProjHash(\( \text{hp}, x, w \))**: Returns \( H_{\text{hp}} \in \mathcal{V} \) using a witness \( w \) linked to the word \( x \).
Those algorithms should ensure two requirements:

- **Correctness**: For any \( x \in \mathcal{L} \), with witness \( w \), \( H_{hk} = H_{hp} \) where, with the keys \( hk \leftarrow \text{HashKG}(\text{param}) \), \( hp \leftarrow \text{ProjKG}(hk) \), the hash values are \( H_{hk} \leftarrow \text{Hash}(hk, x) \) and \( H_{hp} \leftarrow \text{ProjHash}(hp, x, w) \), under the condition that \( \mathcal{R}(x, w) = 1 \);

- **Smoothness**: For any \( x \in \mathcal{X} \backslash \mathcal{L} \), the distributions of \((hp, H_{hk})\) and \((hp, v)\) are indistinguishable where, for the keys \( hk \leftarrow \text{HashKG}(\text{param}) \) and \( hp \leftarrow \text{ProjKG}(hk) \), \( H_{hk} \leftarrow \text{Hash}(hk, x) \), and \( v \leftarrow \mathcal{V} \);

The aforementioned definition of smoothness was introduced by Cramer and Shoup in [CS02]. Two variants of this definition have later been proposed: The first variation has been provided by Gemaro and Lindell in [GL03], 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 [KV09] 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 [CS98] to encompass a *Grey Zone*. An SPHFwGZ is defined with a tuple of algorithms:

- **Setup(\(1^\kappa\))**: Generate the parameters \( \text{param} \) from \( \kappa \), the security parameter, or an explicit random tape \((\sigma, \rho)\) in \( \mathcal{S}_0 \times \mathcal{R}_0 \). \( \text{param} \) 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 \( \mathcal{X} \);
- **HashKG(\(\text{param}\))**: Generates a random hash key \( hk \);
- **ProjKG(hk, x)**: Derives the projection key \( hp \) (it may need \( x \) as input);
- **Hash(hk, x)**: Returns the hash value \( H_{hk} \in \mathcal{V} \), where \( \mathcal{V} \) is the set of hash values, associated to the word \( x \);
- **ProjHash(hp, x, w)**: Returns \( H_{hp} \in \mathcal{V} \) 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 $\text{param}$, defining $\set L, \set L', \set X$:

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

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

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

### 3.2 Word Indistinguishability and Trapdoor

First, we assume languages $\set L$ and $\set L'$ in $\set X$ are defined according to a random tape $(\sigma, \rho)$ sampled in a set $S_0 \times R_0$ (i.e. from $\text{param} \leftarrow \text{Setup}(\sigma, \rho)$). The samplable set $S_0$ is defined together with its twin set $S_1$ such that when $\sigma \in S_1$, and $\text{param} \leftarrow \text{Setup}(\sigma, \rho)$, there exists a trapdoor $td_\sigma$ that allows to test if a given word $x \in \set X$ is in $\set L'$ or not. We then also need the following algorithms:

- $\text{WordGen}_L(\text{param})$: Samples and returns $x \xleftarrow{} \set L$, together with its witness $w$;

- $\text{WordGen}_X(\text{param})$: Samples and returns $x \xleftarrow{} \set X$;

- $\text{WordTest}(td_\sigma, x)$, using the trapdoor $td_\sigma$, tests if $x \in \set L'$.

As we assumed $\set L$ to be a hard subset of $\set X$ when $\sigma \in S_0$, we have the **Word-Indistinguishability Property**. An adversary cannot distinguish between random words in $\set L$ and random words in $\set X$, for any $\sigma \in S_0$, with more than a negligible advantage.

The string $\sigma$ can be seen as a CRS, that admits a trapdoor when sampled from $S_1$. The normal use is with $\sigma \xleftarrow{} S_0$, which needs to be efficiently samplable. When $\sigma \in S_1$, the trapdoor $td_\sigma$ must be easy to compute from $\sigma$.

### 3.3 Decomposition Intractability and Trapdoor

We also define the alternate sets $R_1$ and $R'_1$ for $R_0$. During normal use, $\rho$ is sampled from $R_0$, which needs to be efficiently samplable. When $\rho \in R_1$, and $\text{param} \leftarrow \text{Setup}(\sigma, \rho)$, there exists a trapdoor $td_\rho = (x, x', w, w')$, that must be easy to compute from $\rho$. When $\rho \in R'_1$, and $\text{param} \leftarrow \text{Setup}(\sigma, \rho)$, there exists a trapdoor $td_\rho = (x, x')$, that must be easy to compute from $\rho$. Let us define the complement algorithm, for any $\rho \in R_0 \cup R_1 \cup R'_1$:

- $\text{ComplementWord}(\text{param}, \rho, x)$: from any word $x \in \set X$, it outputs $x'$;

From this complement algorithm, we expect the following statistical properties, for any $\sigma \in S_0 \cup S_1$ but $\rho \in R_0$:

- **Complement**: for any $x \in \set X$, if $x' \leftarrow \text{ComplementWord}(\text{param}, \rho, x)$, then $x = \text{ComplementWord}(\text{param}, \rho, x')$;
– **Alternate:** for any $x \in \mathcal{L}'$, ComplementWord$(\text{param}, \rho, x) \not\in \mathcal{L}'$.

But we also need a computational assumption: the **Decomposition Intractability**, which states that no adversary can generate, with non-negligible probability, for random $(\sigma, \rho) \gets \mathcal{S}_0 \times \mathcal{R}_0$, two words $x, y \not\in \mathcal{L}'$ such that $y = \text{ComplementWord}(\text{param}, \rho, x)$, and so even with the trapdoor $\text{td}_x$.

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

Again, the string $\rho$ 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 $\rho \gets \mathcal{R}_0$, which needs to be efficiently samplable. When $\rho \in \mathcal{R}_1$ or $\rho \in \mathcal{R}_1'$, the trapdoor $\text{td}_\rho$ must be easy to compute from $\rho$.

Eventually, for the security proof to go through, we will make use of the **CRS Indistinguishability**: An adversary cannot 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 $\text{crs} = (\sigma, \rho)$.

### 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 $\text{crs} = (\sigma, \rho) \in \mathcal{S}_0 \times \mathcal{R}_0$ 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 $\Phi_{\text{SPHFwGZ}}^{\text{OT}}$.

The protocol $\Phi_{\text{SPHFwGZ}}^{\text{OT}}$ provides **Correctness**. Indeed, with the honest generation $(x, w) \leftarrow \text{WordGen}(\mathcal{L}(\text{param}))$ we have $c = (H \oplus m, h)$. Then, $m = H \oplus m \oplus \text{ProjHash}(c_1, x, w)$ if and only if $H = \text{ProjHash}(c_1, x, w)$ which is ensured due to the correctness property of the SPHFwGZ. We also need to prove the privacy. But let us proceed in the Universal Composability framework.

#### 4.2 Security Analysis

The proof requires two functionalities: as our Oblivious Transfer protocol will be proven in the CRS-hybrid model (as in [PVW08]), with the functionality $F_{\text{CRS}}$, where the two players get the same random $\text{crs}$ from the $\text{sid}$. In practice, as we assumed $\mathcal{S}_0$ and $\mathcal{R}_0$ efficiently samplable, $(\sigma, \rho)$ can be derived from $H(\text{sid})$. Then, in Figure 2, we recall the ideal functionality for a secure oblivious transfer,
Post-Quantum OT from SPHF with Grey Zone

Receiver

with input $b \in \{0, 1\}$
for $\text{sid}$ and $\text{crs} = (\sigma, \rho)$ from $\mathcal{F}_{\text{CRS}}$

Sender

with input $m_0, m_1 \in \mathcal{M}$
for $\text{sid}$ and $\text{crs} = (\sigma, \rho)$ from $\mathcal{F}_{\text{CRS}}$

\[
\begin{array}{l}
\text{param} \leftarrow \text{Setup}(\sigma, \rho) \\
(x_b, w) \leftarrow \text{WordGen}(\mathcal{L}, \text{param}) \\
x_{1-b} \leftarrow \text{ComplementWord}(\rho, x_b) \\
x_0 \leftarrow \text{ComplementWord}(\rho, x_0) \\
\text{for } i \in \{0, 1\}:
\quad h_{ki} \leftarrow \text{HashKG}(\text{param}) \\
\quad h_{pi} \leftarrow \text{ProjKG}(h_{ki}, x_i) \\
\quad H_i \leftarrow \text{Hash}(h_{ki}, x_i) \\
\quad c_i = (H_i \oplus m_i, h_{pi}) \\
\end{array}
\]

\[H' \leftarrow \text{ProjHash}(c_{b,1}, x_b, w)\]
\[m = H' \oplus c_{0,0}\]

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

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

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

Fig. 2. Functionality $\mathcal{F}_{\text{OT}}$

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 $m_b$.

**Theorem 1.** The protocol $\Phi_{\text{OT}}^{\text{SPHFwGZ}}$ UC-realizes $\mathcal{F}_{\text{OT}}$ in the $\mathcal{F}_{\text{CRS}}$-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 $\mathcal{F}_{\text{CRS}}$ samples $\text{crs}$ in $S_0 \times R_0$.

**Game $G_1$.** In this game, the simulator $\mathcal{S}$ simulates itself the sampling of $\text{crs} = (\sigma, \rho) \leftarrow S_0 \times R_0$, 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 $\text{crs} = (\sigma, \rho) \leftarrow S_0 \times R_0$, the simulator $\mathcal{S}$ samples $\text{crs} = (\sigma, \rho) \leftarrow S_1 \times R_0$, and therefore with the trapdoor $\text{td}_{\sigma}$. 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 $\text{td}_{\sigma}$ to get $t_i = \text{WordTest}(x_i, \text{td}_{\sigma})$ 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 $(\sigma, \rho) \in S_1 \times R_0$. 

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 $S$ proceeds on $m_b$ with $x_b$ and on a random message with $x_{1-b}$. Under the smoothness of the SPHFwGZ, as $x_{1-b} \in L'$, 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 $\text{crs} = (\sigma, \rho) \xleftarrow{\$} S_0 \times R_0$, the simulator $S$ samples $\text{crs} = (\sigma, \rho) \xleftarrow{\$} S_0 \times R_1$, and therefore with the trapdoor $\text{td}_\sigma = (x, x', w, w')$. This game is indistinguishable from the previous one due to the CRS indistinguishability.

Game $G_6$. In this game, the simulator $S$ respectively sets $(x_0, w_0, x_1, w_1)$ as $(x, w, x', w')$ from $\text{td}_\rho$. 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 $\text{crs} = (\sigma, \rho) \xleftarrow{\$} S_0 \times R_0$, the simulator $S$ samples $\text{crs} = (\sigma, \rho) \xleftarrow{\$} S_0 \times R_1'$, and therefore with the trapdoor $\text{td}_\rho = (x, x')$, and simulates the flows with random $m_0, m_1 \xleftarrow{\$} M$ and random $b \xleftarrow{\$} \{0, 1\}$. Under the CRS Indistinguishability and the smoothness of the SPHFwGZ, as both $x, x' \in L'$, 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 $F_{OT}$ which leads to the following simulator:

- If no participant is corrupted, one uses $\text{crs} \xleftarrow{\$} S_0 \times R_1'$, and the simulator $S$ simply uses random inputs for the sender and the receiver;
- If the receiver is corrupted, one uses $\text{crs} \xleftarrow{\$} S_1 \times R_0$, and the simulator $S$ extracts $b$ using the trapdoor $\text{td}_\sigma$, and sends $(\text{sid}, \text{receiver}, b)$ to $F_{OT}$;
- If the sender is corrupted, one uses $\text{crs} \xleftarrow{\$} S_0 \times R_1$, and the simulator $S$ extracts $m_0, m_1$ using the trapdoor $\text{td}_\rho$, and sends $(\text{sid}, \text{sender}, m_0, m_1)$ to $F_{OT}$;
- The adversary sends $(\text{sid}, \text{answer})$ 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 instanciation of our Oblivious Transfer from any (possibly Noisy) Homomorphic Encryption.

We consider a noisy encryption scheme $\Pi = \{\text{Setup}, \text{KeyGen}, \text{Encrypt}, \text{Decrypt}\}$ with possible decryption failures. $X$ as the ciphertext space of $\Pi$, whereas

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

Sets $S_0$ and $S_1$ can both be seen as public keys $pk$ generated from $\text{KeyGen}(1^n)$ except that when $\sigma \in S_1$, the secret key $sk$ is known and defines the trapdoor $\text{td}_\sigma$. Hence, $\sigma$ (which defines the public key $pk$) defines the sets $L$ and $L'$ in $X$.

On the other hand, we can define $R_0 = X$, the set of all the ciphertexts, or a superset, with uniform distribution; $R_1 = \{c_0 \otimes c_1\}$, 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 $td_\rho$ as $(c_0, c_1, r_0, r_1)$: $\mathcal{R}_0' = \{c_0 \otimes c_1\}$, 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 $td_\rho$ 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 $\Pi, L, L'$, and $\mathcal{X}$ are indistinguishable. The homomorphic property implies that $\{x \otimes x'| (x, x') \in \mathcal{X}'\} = \mathcal{X}$. As a consequence, we have indistinguishability between $\mathcal{R}_0 = \mathcal{X} = \{x \otimes x'| (x, x') \in \mathcal{X}'\}$, $\mathcal{R}_1 = \{x \otimes x'| (x, x') \in \mathcal{L}'\}$, and $\mathcal{R}_1' = \{x \otimes x'| (x, x') \in \mathcal{L}'\}$. Furthermore, as $\mathcal{S}_0 = \mathcal{S}_1$, they are perfectly indistinguishable;

- **Word Indistinguishability**: Under the semantic security of the encryption scheme $\Pi$, one cannot distinguish between $c_0 \in \mathcal{L}$, an encryption of $0$ and $c_1 \in \mathcal{X}$, an encryption of a random value.

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 $G$ of prime order $p$, the Decisional Diffie-Hellman problem consists in, given $g^a$ and $g^b$, distinguishing $g^{ab}$ from $g^{c}$, for $a, b, c \overset{\$}{\leftarrow} Z_p$.

**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 [ElG85] in a group $G = \langle g \rangle$ of prime order $p$, defined by the Setup algorithm:

- **KeyGen($1^\kappa$):** picks $\beta \overset{\$}{\leftarrow} Z_p$, and sets $pk = h = g^\beta$, $sk = \beta$.

- **Encrypt($pk = h = g^\beta, M \in G$) encrypts the message $M$ under the public key $pk$ as follows:** Pick $r \overset{\$}{\leftarrow} Z_p$; Output the ciphertext: $c = (g^r, h^r \cdot M)$;

- **Decrypt($sk, c = (c_0, c_1)$) decrypts the ciphertext $c$ using the decryption key $sk$ as follows:** $M = c_1/c_0^sk$.

**Theorem 2.** The above ElGamal encryption scheme is IND-CPA under the Decisional Diffie-Hellman assumption.
**SPHFwGZ from ElGamal Encryption.** From the above ElGamal encryption scheme $\text{EG} = (\text{Setup}, \text{KeyGen}, \text{Encrypt}, \text{Decrypt})$, in a group $\mathbb{G}$, denoted multiplicatively, of prime order $p$, with generator $g$.

We set $S_0 = S_1 = \{\sigma = h = g^{u_0}; \text{td}_\sigma \triangleq \Z_p\}$. Then, $\mathcal{R}_0$ is defined as $\mathbb{G}^2 = \{\rho = (\hat{g}, \hat{h}) \triangleq \mathbb{G}^2\}$ and $\mathcal{R}_1$ as $\{\rho = (\hat{g} = g^{u_0}, \hat{h} = h^{u_0}); \hat{g} \triangleq \mathbb{G}, \text{td}_\sigma \triangleq \Z_p\}$. The crs is set as $(\sigma, \rho)$. One can note that witnesses only exist when $\rho \in \mathcal{R}_1$, then $\text{td}_\rho = ((c_0, c_1), (c_0 \cdot \hat{g}, c_1 \cdot \hat{h}), r, r + \text{td}_\sigma)$, where $(c_0, c_1)$ is an encryption of $M = g^0$, with randomness $r$; while $\text{td}_\sigma$ 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 $\text{param} = (g, \sigma = h)$, which determines all the sets (specified by the Setup algorithm), we can define:

- $\text{hk} = \text{HashKG}(\text{param}) = (\alpha, \beta) \triangleq \Z_p^2$;
- $\text{hp} = \text{ProjKG}(\text{hk}) = g^\alpha h^\beta$;
- $H = \text{Hash}(\text{hk}, x = (u, v)) = u^\alpha v^\beta \in \mathbb{G}$;
- $H' = \text{ProjHash}(\text{hp}, x, w = r) = \text{hp}^r$, if $x = (g^r, h^r) \in \mathcal{L}$.

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 $r$, $H = u^\alpha v^\beta = (g^r h^\beta)^r = \text{hp}^r = H'$;
- **Smoothness:** When $x = (u, v) = (g^r, h^r) \notin \mathcal{L}$, then $r' = r + r''$ with $r'' \neq 0$:
  
  $H = u^\alpha v^\beta = (g^r h^\beta)^r \cdot g^{r''}\beta = \text{hp}^r \cdot g^{r''}\beta = H' \cdot g^{r''}\beta$. But $\beta$ is perfectly hidden in $\text{hp}$, and $g^{r''}\beta$ is perfectly unpredictable;
- **Decomposition Intractability:** We can note that 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 $\mathcal{L} = \mathcal{X}\backslash \mathcal{L}$. A random ciphertext $\rho$ 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 $\rho$. Hence, the decomposition intractability is statistical: the probability of existence of the decomposition is bounded by $1/p$, on $\rho$, even knowing the decryption key, and thus the trapdoor $\text{td}_\sigma$.

Note that this construction exactly corresponds to the one from [CS98].

### 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 $\lambda$. Given a basis $\mathcal{B}$ of an $n$-dimensional lattice, find a set of $n$ linearly independent vectors $v_1, \ldots, v_n \in \mathcal{L} (\mathcal{B})$ such that $\|v_i\| \leq \gamma(n) \cdot \lambda_n(\mathcal{B})$, for all $1 \leq i \leq n$. The approximation factor $\gamma$ is typically a polynomial in $n$, the non approximated version assumes $\gamma = 1$. 


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

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

Regev Encryption. Regev [Reg05] 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 $q/\sigma = \text{poly}(n)$, $\text{LWE}_{\chi, q}$ 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 [MP12] to build our public matrix $A$. Define $g_A(s, e) = As + e$, the gadget matrix $G$ as $G^t = I_n \otimes g^t$, where $g^t = [1, 2, \ldots, 2^k]$ and $k = \lceil \log q \rceil - 1$, and let $H \in \mathbb{Z}_q^{n \times n}$ be invertible. The notation $[A | B]$ is for horizontal concatenation, while $\langle A, B \rangle$ is for vertical concatenation.

Lemma 1 ([MP12, Theorems 5.1 and 5.4]). There exist two PPT algorithms $\text{TrapGen}$ and $g_{\chi, q}^{-1}$ with the following properties assuming $q \geq 2$ and $m \geq \Theta(n \log q)$:

- $\text{TrapGen}(1^n, 1^m, q)$ outputs $(T, A_0)$, where the distribution of the matrix $A_0$ is at negligible statistical distance from uniform in $\mathbb{Z}_q^{m \times n}$, and such that $TA_0 = 0$, where $s_1(T) \leq O(\sqrt{m})$ and where $s_1(T)$ is the operator norm of $T$, which is defined as $\max_{x \neq 0} ||Tx||/||x||$.\(^5\)

- Let $(T, A_0) \leftarrow \text{TrapGen}(1^n, 1^m, q)$. Let $A_H = A_0 + [0 : GH]$ for some invertible matrix $H$ called a tag. Then, we have $TA_H = GH$. Furthermore, if $x \in \mathbb{Z}_q^n$ can be written as $A_HS + e$, with $s \in \mathbb{Z}_q^n$ and $e \in \mathbb{Z}_q^n$ where $||e|| \leq B' := q/\Theta(\sqrt{m})$, then $g_{\chi, q}^{-1}(T, x, H)$ outputs $(s, e)$.

More precisely, to sample $(T, A_0)$ with $\text{TrapGen}$, we sample a uniform $\tilde{A} \in \mathbb{Z}_{q}^{\tilde{m} \times n}$ where $\tilde{m} = m - nk = \Theta(n \log q)$, and some $R \leftarrow \mathcal{D}^{nk \times \tilde{m}}$, where the distribution $\mathcal{D}$ assigns probability $1/2$ to $0$, and $1/4$ to $\pm 1$. We output $T = [-R | I_{nk}]$ along with $A_0 = [\tilde{A} ; RA]$. Then, given a tag $H$, with $A_H = A_0 + [0 : GH]$, we have: $TA_H = GH$.

We will only consider a fixed tag $H = I$, for the Micciancio-Peikert encryption [MP12]. 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.

\(^5\)The bound on $s_1(T)$ 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 $\text{Encode}(\mu \in \{0, 1\}) = \mu \cdot (0, \ldots, 0, [q/2])^t$. Note that $2 \cdot \text{Encode}(\mu) = (0, \ldots, 0, \mu)^t \mod q$, as $[q/2]$ is the inverse of 2 mod $q$, for such an odd $q$.

Let $(T, A_0) \leftarrow \text{TrapGen}(1^n, 1^m, q)$. The public encryption key is $pk = A_0$, and the secret decryption key is $sk = T$.

- $\text{Encrypt}(pk = A_0, \mu \in \{0, 1\})$ encrypts the message $\mu$ under the public key $pk$ as follows: Let $A = A_0 \oplus [0; G]$. Pick $s \in \mathbb{Z}_q^n$, $e \leftarrow \mathbb{Z}_q^m$ where $t = \sigma \sqrt{m} \cdot \omega(\sqrt{\log n})$. Restart if $\|e\| > B$, where $B = 2t\sqrt{m}$. Output the ciphertext:

$$c = As + e + \text{Encode}(\mu) \mod q \ .$$

- $\text{Decrypt}(sk = T, c \in \mathbb{Z}_q^n)$ decrypts the ciphertext $c$ using the decryption key $sk$ as follows: With $B'' = q/2\Theta(\sqrt{m})$, output

$$\begin{cases} 
\mu & \text{if } g_A^{-1}(T, 2c, I) = (2s, 2e + (0, \ldots, 0, \mu)) \\
\perp & \text{otherwise.}
\end{cases}$$

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

$$\mu' := \text{Decrypt}(T, c) \neq \perp \iff d(c - \text{Encode}(\mu'), A(A)) < B'' \ .$$

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

Theorem 3. Assume $m \geq \Theta(n \log q)$. The above scheme is IND-CPA assuming the hardness of the LWE$_{\chi,q}$ problem for $\chi = \mathbb{Z}_q$. Furthermore, this encryption scheme is homomorphic for plaintexts in $(\mathbb{Z}_2, +)$, and ciphertexts in $\mathbb{Z}_q^n$ 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 $m = n \log(q)$, $t = \sqrt{mn} \cdot \omega(\sqrt{\log(n)})$, $k = \Theta(n)$, $s \geq \Theta(\sqrt{m}) \land s/q = \text{negl}(n)$, $s = \Omega(m^k q^{2/3})$. We also set $R$ to be a probabilistic rounding function from $[0, 1]$ to $\{0, 1\}$, such that $R(x) = 1$ with probability $0.5 \cdot \cos(2\pi x)$ and 0 otherwise.

We set $S_0 = S_1 = \{\sigma = A = A_0 + [0; G]|(T, A_0) \leftarrow \text{TrapGen}(1^n, 1^m, q)\}$, $\text{td}_\sigma$ being $T$. Then, $R_0$ is defined as $\{\rho = v \in \mathbb{Z}_q^n\}$ and $R_1$ is the set composed

---

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

7 Note that the inversion algorithm $g_\chi^{-1}$ can succeed even if $\|e\| > B''/2$, depending on the randomness of the trapdoor. It is crucial to reject decryption nevertheless when $\|e\| > B''$ to ensure security.
of all the sums of two honest encryptions of 0, in other words \( \{ \rho = A(s + s') + e + e' \pmod q \mid s, s' \in \mathbb{Z}_q^*, e, e' \leftarrow D_{\mathbb{Z}_q^*} \land \|e\| \leq B \land \|e'\| \leq B \} \) with \( \text{td}_\rho = (A, \rho, A^s + e, (s, e), (s', e')) \).

With \( \mathcal{X} = \{ c \in \mathbb{Z}_q^* \} \), \( \mathcal{L} = \{ c \mid \exists s, e, c = \text{Encrypt}(A_0, 0; s, e) \} \) defined following the description above, and \( \mathcal{L'} = \{ c \in \mathcal{X} | \text{Decrypt}(T, c) \neq 0 \} \), note that \( s \) could be enough as a witness for \( c = A + e \in \mathcal{L} \), as one can check \( e = c - A \) is small enough. This defines the \text{Setup} algorithm, and we have:

**Definition 4 (Bit-SPHFwGZ over Micciancio-Peikert like Ciphertexts [BBDQ18])**. For \( k = \Theta(n) \), and picking \( s \geq \Theta(\sqrt{n}) \), and \( s = \Omega(mk^2q^{2/3}) \), we can define:

- \( \text{HashKG}(\text{param}) = \text{hk} = h \overset{\$}{\leftarrow} D_{\mathbb{Z}_{2^n}}^m \)
- \( \text{ProjKG}(\text{hk}) = \text{hp} = A'h \)
- \( \text{Hash}(\text{hk}, c) = R((\text{hk}, c)) = R((h, c)) \in \{0, 1\} \)
- \( \text{ProjHash}(\text{hp}, c, w = s) = R((\text{hp}, s)) = R((A'h, s)) \)

For a word \( c = As + e \) in the language \( \mathcal{L} \), \( (h, c) = hAs + h'e = (A'h, s) + h'e \).

And by construction \( h'e \) is small. The choice of the rounding function \( R(x) \), characterized by a coin flip where the outcome 1 is weighted by \( 0.5 \cdot \cos(\frac{\pi x}{q}) \), is such that it allows to cancel out this small noise most of the time, while providing smoothness for words outside the language (ensuring that \( R((\text{hk}, c)) \) is random when given only \( \text{hp} \)).

It was shown in [BBDQ18], 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 \( \text{Encrypt}(\text{pk}, m; r) \) should encryption a redundant version of \( m \), with an error-correcting code, \( \text{ECC}(m) \). 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 \( c \in \mathcal{X} \), for any \( \ell = \Omega(n) \) an error-correcting code \( \text{ECC} \) capable of correcting \( \ell/4 \) errors, then, we can define the SPHF as:

- \( \text{SETUP}(1^n) \): Outputs the result from \( \text{Setup}(1^n) \)
- \( \text{HASHKG}(\text{param}) \): Picks a random values \( K \leftarrow \{0, 1\}^\ell \), and \( \forall i \in [\ell] \), gets \( \text{hk}_i = \text{HashKG}(\text{param}) \), and set \( \text{HK} = (\{\text{hk}_i\}, K) \)
Another amplification is required to make full-fledged decomposition intractability.

\[ \text{HASH(HK, c): Returns } K, \text{ from HK; } \]

\[ \text{PROJHASH}(HP, c, w = s) : \text{forall } i \in [\ell], \text{ computes } H'_i = \text{ProjHash}(hp_i, c, s). \]

Then computes \( S' = (H'_i)_{i \in [\ell]}, \text{ and finally } K' = \text{ECC}^{-1}(T \oplus S'). \)

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 [Hoe63]. We can guarantee the expected properties:

- **Correctness:** When \( x = c \in \mathcal{L} \), with the above conversion, we have \( K = K' \) with overwhelming probability, thanks to the error-correcting code;

- **Smoothness:** When \( x = c \notin \mathcal{L} \), 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 \( \rho \) 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 \( \rho \), we can remark that a vector verifies this property as soon as \( d(\rho, \Lambda(A)) \) 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, \( \Pr_{\rho \in \mathbb{Z}_p^n}[\exists c, d | \rho = c + d \land \text{Decrypt}(sk, c) = \text{Decrypt}(sk, d) = 0] \leq 1/2 \). 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 \( (c_j)_{j \in [k]} \), with \( k \) parallel executions of the SPHFwGZ, 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. The acceptable language, for correctness is then:

\[ \tilde{L} = \mathcal{L}^k = \{(c_j)_{j \in [k]} | (\forall j \in [k]), \exists (s_j, e_j), c = \text{Encrypt}(A_0, 0; s_j, e_j) \} \subset X^k \]

whereas the language for the smoothness becomes:

\[ \tilde{L}' = \mathcal{L}'^k = \{(c_j)_{j \in [k]} | (\exists j \in [k]), \text{Decrypt}(T, c_j) \neq 0 \} \subset X^k \]

Then, for random \( (\rho_j)_{j \in [k]} \overset{\$}{\leftarrow} X^k \), a decomposition would be a list of pairs \( (c_j, \tilde{d}_j)_{j \in [k]} \in (X \times X)^k \) such that for all \( j, \rho_j = c_j + \tilde{d}_j \) and \( \text{Decrypt}(T, c_j) = \)
Decrypt(T, 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.

References

ACP09. Michel Abdalla, Céline Chevalier, and David Pointcheval. Smooth projective hashing for conditionally extractable commitments. In Shai Halevi, editor, CRYPTO 2009, volume 5677 of LNCS, pages 671–689. Springer, Heidelberg, August 2009.

BBB+21. Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Yann Connan, and Philippe Gaborit. A gapless code-based hash proof system based on RQC and its applications. Cryptology ePrint Archive, Report 2021/026, 2021. https://eprint.iacr.org/2021/026.

BBC+13a. Fabrice Ben Hamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud. Efficient UC-secure authenticated key-exchange for algebraic languages. In Kaoru Kurosawa and Goichiro Hanaoka, editors, PKC 2013, volume 7778 of LNCS, pages 272–291. Springer, Heidelberg, February / March 2013.

BBC+13b. Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud. New techniques for SPHF s and efficient one-round PAKE protocols. In Ran Canetti and Juan A. Garay, editors, CRYPTO 2013, Part I, volume 8042 of LNCS, pages 449–475. Springer, Heidelberg, August 2013.

BBDQ18. Fabrice Benhamouda, Olivier Blazy, Léo Ducas, and Willy Quach. Hash proof systems over lattices revisited. In Michel Abdalla and Ricardo Dahab, editors, PKC 2018, Part II, volume 10770 of LNCS, pages 644–674. Springer, Heidelberg, March 2018.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In 42nd FOCS, pages 136–145. IEEE Computer Society Press, October 2001.

CS98. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, CRYPTO’98, volume 1462 of LNCS, pages 13–25. Springer, Heidelberg, August 1998.

CS02. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, EUROCRYPT 2002, volume 2332 of LNCS, pages 45–64. Springer, Heidelberg, April / May 2002.
DS15. David Derler and Daniel Slamanig. Practical witness encryption for algebraic languages and how to reply an unknown whistle-blower. Cryptology ePrint Archive, Report 2015/1073, 2015. https://eprint.iacr.org/2015/1073.

ElG85. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, 31:469–472, 1985.

GL03. Rosario Gennaro and Yehuda Lindell. A framework for password-based authenticated key exchange. In Eli Biham, editor, EUROCRYPT 2003, volume 2656 of LNCS, pages 524–543. Springer, Heidelberg, May 2003. https://eprint.iacr.org/2003/032.ps.gz.

Hoe63. Wassily Hoeffding. Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association, 58(301):13–30, 1963.

JR12. Charanjit S. Jutla and Arnab Roy. Relatively-sound NIZKs and password-based key-exchange. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, PKC 2012, volume 7293 of LNCS, pages 485–503. Springer, Heidelberg, May 2012.

KV09. Jonathan Katz and Vinod Vaikuntanathan. Smooth projective hashing and password-based authenticated key exchange from lattices. In Mitsuru Matsui, editor, ASIACRYPT 2009, volume 5912 of LNCS, pages 636–652. Springer, Heidelberg, December 2009.

KV11. Jonathan Katz and Vinod Vaikuntanathan. Round-optimal password-based authenticated key exchange. In Yuval Ishai, editor, TCC 2011, volume 6597 of LNCS, pages 293–310. Springer, Heidelberg, March 2011.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, EUROCRYPT 2012, volume 7237 of LNCS, pages 700–718. Springer, Heidelberg, April 2012.

Per13. Edoardo Persichetti. Secure and anonymous hybrid encryption from coding theory. In Philippe Gaborit, editor, Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013, pages 174–187. Springer, Heidelberg, June 2013.

PVW08. Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, CRYPTO 2008, volume 5157 of LNCS, pages 554–571. Springer, Heidelberg, August 2008.

Rab81. Michael O. Rabin. How to exchange secrets with oblivious transfer. Technical Report TR-81, Aiken Computation Laboratory, Harvard University, 1981.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, 37th ACM STOC, pages 84–93. ACM Press, May 2005.

SA21. Masoumeh Koochak Shooshhti and Mohammad Reza Aref. Smooth projective hash function from codes and its applications. IEEE Transactions on Services Computing, pages 1–1, 2021.

ZY17. Jiang Zhang and Yu Yu. Two-round PAKE from approximate SPH and instantiations from lattices. In Tsuyoshi Takagi and Thomas Peyrin, editors, ASIACRYPT 2017, Part III, volume 10620 of LNCS, pages 37–67. Springer, Heidelberg, December 2017.