Structure-Preserving Smooth Projective Hashing

Olivier Blazy[1](✉) and Céline Chevalier[2]

[1] Université de Limoges, XLim, Limoges, France
olivier.blazy@unilim.fr
[2] CRED, Université Panthéon-Assas, Paris, France

Abstract. Smooth projective hashing has proven to be an extremely useful primitive, in particular when used in conjunction with commitments to provide implicit decommitment. This has lead to applications proven secure in the UC framework, even in presence of an adversary which can do adaptive corruptions, like for example Password Authenticated Key Exchange (PAKE), and 1-out-of-m Oblivious Transfer (OT). However such solutions still lack in efficiency, since they heavily scale on the underlying message length.

Structure-preserving cryptography aims at providing elegant and efficient schemes based on classical assumptions and standard group operations on group elements. Recent trend focuses on constructions of structure-preserving signatures, which require message, signature and verification keys to lie in the base group, while the verification equations only consist of pairing-product equations. Classical constructions of Smooth Projective Hash Function suffer from the same limitation as classical signatures: at least one part of the computation (messages for signature, witnesses for SPHF) is a scalar.

In this work, we introduce and instantiate the concept of Structure-Preserving Smooth Projective Hash Function, and give as applications more efficient instantiations for one-round PAKE and three-round OT, and information retrieval thanks to Anonymous Credentials, all UC-secure against adaptive adversaries.

Keywords: Smooth projective hash functions · Structure preserving · Oblivious transfer · Password authenticated key exchange · UC Framework · Credentials

1 Introduction

Smooth Projective Hash Functions (SPHF) were introduced by Cramer and Shoup [30] as a means to design chosen-ciphertext-secure public-key encryption schemes. These hash functions are defined such as their value can be computed in two different ways if the input belongs to a particular subset (the *language*), either using a private hashing key or a public projection key along with a private witness ensuring that the input belongs to the language.

J.H. Cheon and T. Takagi (Eds.): ASIACRYPT 2016, Part II, LNCS 10032, pp. 339–369, 2016.
DOI: 10.1007/978-3-662-53890-6_12

In addition to providing a more intuitive abstraction for their original public-key encryption scheme in [29], the notion of SPHF also enables new efficient instantiations of their scheme under different complexity assumptions such as DLin, or more generally $k - \mathsf{MDDH}$. Due to its usefulness, the notion of SPHF was later extended to several interactive contexts. One of the most classical applications is to combine them with commitments in order to provide *implicit* decommitments.

Commitment schemes have become a central tool used in cryptographic protocols. These two-party primitives (between a committer and a receiver) are divided into two phases. First, in the *commit* phase, the committer gives the receiver an analogue of a sealed envelope containing a value m, while later in the *opening* phase, the committer reveals m in such a way that the receiver can verify whether it was indeed m that was contained in the envelope. In many applications, for example password-based authenticated key-exchange, in which the committed value is a password, one wants the opening to be implicit, which means that the committer does not really open its commitment, but rather convinces the receiver that it actually committed to the value it pretended to.

An additional difficulty arises when one wants to prove the protocols in the universal composability framework proposed in [22]. Skipping the details, when the protocol uses commitments, this usually forces those commitments to be simultaneously *extractable* (meaning that a simulator can recover the committed value m thanks to a trapdoor) and *equivocable* (meaning that a simulator can open a commitment to a value m' different from the committed value m thanks to a trapdoor), which is quite a difficult goal to achieve.

Using SPHF with commitments to achieve an implicit decommitment, the language is usually defined on group elements, with projection keys being group elements, and witnesses being scalars. While in several applications, this has already lead to efficient constructions, the fact that witnesses have to be scalars (and in particular in case of commitments, the randomness used to commit) leads to drastic restrictions when trying to build protocols secure against adaptive corruptions in the UC framework.

This is the classical paradigm of protocol design, where generic primitives used in a modular approach lead to a simple design but quite inefficient constructions, while when trying to move to ad-hoc constructions, the conceptual simplicity is lost and even though efficiency might be gained, a proper security proof gets trickier. Following the same kind of reasoning, [5] introduced the concept of structure-preserving signatures in order to take the best of both worlds. There has been an ongoing series of work surrounding this notion, for instance [3,4,6–8]. This has shown that structure-preserving cryptography indeed provides the tools needed to have simultaneously simple and efficient protocols.

1.1 Related Work

Smooth Projective Hash Functions (SPHF) were introduced by Cramer and Shoup [30] and have been widely used since then, for instance for password-authenticated key exchange (PAKE) [2,14,35,43,44], or oblivious transfer (OT)

[1,28,41], and a classification was introduced separating SPHF into three main kinds, KV-SPHF,CS-SPHF,GL-SPHF depending on how the projection keys are generated and when, the former allowing one-round protocols, while the latter have more efficient communication costs (see Sect. 2.2).

Password-Authenticated Key Exchange (PAKE) protocols were proposed in 1992 by Bellovin and Merritt [12] where authentication is done using a simple password, possibly drawn from a small entropy space subject to exhaustive search. Since then, many schemes have been proposed and studied. SPHF have been extensively used, starting with the work of Gennaro and Lindell [35] which generalized an earlier construction by Katz, Ostrovsky, and Yung [42], and followed by several other works [2,24]. More recently, a variant of SPHF proposed by Katz and Vaikuntanathan even allowed the construction of one-round PAKE schemes [14,44]. The most efficient PAKE scheme so far (using completely different techniques) is the recent Asiacrypt paper [40].

The first ideal functionality for PAKE protocols in the UC framework [22,25] was proposed by Canetti *et al.* [24], who showed how a simple variant of the Gennaro-Lindell methodology [35] could lead to a secure protocol. Though quite efficient, their protocol was not known to be secure against adaptive adversaries, that are capable of corrupting players at any time, and learn their internal states. The first ones to propose an adaptively secure PAKE in the UC framework were Barak *et al.* [10] using general techniques from multi-party computation. Though conceptually simple, their solution results in quite inefficient schemes.

Recent adaptively secure PAKE were proposed by Abdalla *et al.* [1,2], following the Gennaro-Lindell methodology with variation of the Canetti-Fischlin commitment [23]. However their communication size is growing in the size of the passwords, which is leaking information about an upper-bound on the password used in each exchange.

Oblivious Transfer (OT) was introduced in 1981 by Rabin [51] as a way to allow a receiver to get exactly one out of k messages sent by another party, the sender. In these schemes, the receiver should be oblivious to the other values, and the sender should be oblivious to which value was received. Since then, several instantiations and optimizations of such protocols have appeared in the literature, including proposals in the UC framework [26,48].

More recently, new instantiations have been proposed, trying to reach round-optimality [38], or low communication costs [50]. The 1-out-of-2 OT scheme by Choi *et al.* [28] based on the DDH assumption seems to be the most efficient one among those that are secure against adaptive corruptions in the CRS model with erasures. But it does not scale to 1-out-of-m OT, for $m > 2$. [1,17] proposed a generic construction of 1-out-of-m OT secure against adaptive corruptions in the CRS model, however the commitment was still growing in the logarithm of the database length. While this is not so much a security issue for OT as this length is supposed to be fixed at the start of the protocol, this is however a weak spot for the efficiency of the final construction.

1.2 Our Contributions

Similarly to structure-preserving signatures requiring the message, the signature, and the public keys to be group elements, we propose in this paper the notion of structure-preserving Smooth Projective Hash Functions (SP-SPHF), where both words, witnesses and projection keys are group elements, and hash and projective hash computations are doable with simple pairing-product equations in the context of bilinear groups.

This allows, for example, to build Smooth Projective Hash Functions that implicitly demonstrate the knowledge of a Groth Sahai Proof (serving as a witness).

We show how to transform every previously known pairing-less construction of SPHF to fit this methodology, and then propose several applications in which storing a group element as a witness allows to avoid the drastic restrictions that arise when building protocols secure against adaptive corruptions in the UC framework with a scalar as witness. Asking the witness to be a group element enables us to gain more freedom in the simulation (the discrete logarithm of this element and / or real extraction from a commitment). For instance, the simulator can always commit honestly to a random message, since it only needs to modify its witness in the equivocation phase. Furthermore, it allows to avoid bit-per-bit construction. Such design carries similarity with the publicly verifiable MACs from [45], where the pairing operation allows to relax the verification procedure.

A work from Jutla and Roy has appeared on eprint [39] considering a parallel between QA-NIZK and SPHF: Independently from ours, they define a transformation from one to another. Their transformation can then be extended to view QA-NIZK as a special case of SP-SPHF, and so be encompassed by our framework.

As an example, we show that the UC-commitment from [34] (while not fitting with the methodology of traditional SPHF from [1]), is compatible with SP-SPHF and can be used to build UC protocols. As a side contribution, we first generalize this commitment from DLin to the $k-\mathsf{MDDH}$ assumption from [33]. The combination of this commitment and the associated SP-SPHF then enables us to give three interesting applications.

Adaptively secure 1-out-of-m Oblivious Transfer. First, we provide a construction of a three-round UC-secure 1-out-of-m OT. Assuming reliable erasures and a single global CRS, we show in Sect. 5 that our instantiation is UC-secure against adaptive adversaries. Besides having a lesser number of rounds than most recent existing OT schemes with similar security levels, our resulting protocol also has a better communication complexity than the best known solutions so far [1,28] (see Table 1 for a comparison). For ease of readability, we emphasize in this table the SXDH communication cost[1], which is simply k-MDDH for $k = 1$. Our protocol is "nearly optimal" in the sense that it is still linear in the number of lines m, but the constant in front of m is 1.

[1] Our OT and PAKE protocols are described in k-MDDH but one directly obtains the SXDH versions by simply letting $k = 1$ in the commitment presented in Sect. 4.2 (see the paper full version [15] for details).

Table 1. Comparison with existing UC-secure OT schemes

	Flow	Communication Complexity	Assumption	1-out-of
[28]	4	$26\ \mathbb{G} + 7\ \mathbb{Z}_p$	DDH	2
[1]	3	$(m + 8\log m) \times \mathbb{G}_1 + \log m \times \mathbb{G}_2 + 1 \times \mathbb{Z}_p$	SXDH	m
This paper	3	$(k+3) \times \mathbb{G}_1 + (2 + (3+k)m + k(k+1))$ $\times \mathbb{G}_2 + m \times \mathbb{Z}_p$	$k - \mathsf{MDDH}$	m
This paper	3	$4 \times \mathbb{G}_1 + 12 \times \mathbb{G}_2 + 2 \times \mathbb{Z}_p$	SXDH	m

One-round adaptively secure PAKE. Then, we provide an instantiation of a one-round UC-secure PAKE under any $k - \mathsf{MDDH}$ assumption. Once again, we show in Sect. 6 that the UC-security holds against adaptive adversaries, assuming reliable erasures and a single global CRS. Contrarily to most existing one-round adaptively secure PAKE, we show that our scheme enjoys a much better communication complexity while not leaking information about the length of the password used (see Table 2 for a comparison, in particular for the SXDH version). Only [40] achieves a slightly better complexity as ours, but only for SXDH, while ours easily extends to $k - \mathsf{MDDH}$. Furthermore, our construction is an extension to SP-SPHF of well-known classical constructions based on SPHF, which makes it simpler to understand. We omit [17] from the following table, as its contribution is to widen the construction to non-pairing based hypotheses.

Anonymous Credential-Based Message Transmission. Typical credential use involves three main parties. Users need to interact with some authorities to obtain their credentials (assumed to be a set of attributes validated / signed), and then prove to a server that a subpart of their attributes verifies an expect policy. We present a constant-size, round-optimal protocol that allow to use a Credential to retrieve a message without revealing the Anonymous Credentials in a UC secure way, by simply building on the technique proposed earlier in the paper.

Table 2. Comparison with existing UC-secure PAKE schemes where $|\text{password}| = m$

	Adaptive	One-round	Communication complexity	Assumption
[2]	yes	no	$2 \times (2m + 22m\mathfrak{K}) \times \mathbb{G} + \text{OTS}$	DDH
[44]	no	yes	$\approx 2 \times 70 \times \mathbb{G}$	DLIN
[14]	no	yes	$2 \times 6 \times \mathbb{G}_1 + 2 \times 5 \times \mathbb{G}_2$	SXDH
[1]	yes	yes	$2 \times 10m \times \mathbb{G}_1 + 2 \times m \times \mathbb{G}_2$	SXDH
[40]	yes	yes	$4 \times \mathbb{G}_1 + 4 \times \mathbb{G}_2$	SXDH
this paper	yes	yes	$2 \times (k+3) \times \mathbb{G}_1 + 2 \times (k + 3$ $+ k(k+1)) \times \mathbb{G}_2$	k-MDDH
this paper	yes	yes	$2 \times 4 \times \mathbb{G}_1 + 2 \times 5 \times \mathbb{G}_2$	SXDH

2 Definitions

2.1 Notations

If $\boldsymbol{x} \in \mathcal{S}^n$, then $|\boldsymbol{x}|$ denotes the length n of the vector, and by default vectors are assumed to be column vectors. Further, $x \overset{\$}{\leftarrow} \mathcal{S}$ denotes the process of sampling an element x from the set $\mathcal{S}$ uniformly at random.

2.2 Primitives

Encryption. An encryption scheme $\mathcal{C}$ is described through four algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$, defined formally in Appendix A.1.

Commitments. We refer the reader to [1] for formal definitions and results but we give here an informal overview to help the unfamiliar reader with the following. A *non-interactive labelled commitment scheme* $\mathcal{C}$ is defined by three algorithms:

- $\mathsf{SetupCom}(1^{\mathfrak{K}})$ takes as input the security parameter $\mathfrak{K}$ and outputs the global parameters, passed through the CRS ρ to all other algorithms;
- $\mathsf{Com}^{\ell}(x)$ takes as input a label ℓ and a message x, and outputs a pair (C, δ), where C is the commitment of x for the label ℓ, and δ is the corresponding opening data (a.k.a. decommitment information). This is a probabilistic algorithm.
- $\mathsf{VerCom}^{\ell}(C, x, \delta)$ takes as input a commitment C, a label ℓ, a message x, and the opening data δ and outputs 1 (true) if δ is a valid opening data for C, x and ℓ. It always outputs 0 (false) on $x = \bot$.

The basic properties required for commitments are *correctness* (for all correctly generated CRS ρ, all commitments and opening data honestly generated pass the verification VerCom test), the *hiding property* (the commitment does not leak any information about the committed value) and the *binding property* (no adversary can open a commitment in two different ways). More complex properties (equivocability and extractability) are required by the UC framework and described in Appendix A.2 for lack of space.

Smooth Projective Hash Functions. SPHF were introduced by Cramer and Shoup [30] for constructing encryption schemes. A projective hashing family is a family of hash functions that can be evaluated in two ways: using the (secret) hashing key, one can compute the function on every point in its domain, whereas using the (public) *projected* key one can only compute the function on a special subset of its domain. Such a family is deemed *smooth* if the value of the hash function on any point outside the special subset is independent of the projected key. The notion of SPHF has already found numerous applications in various contexts in cryptography (*e.g.* [2,19,35,41]).

Definition 1. Smooth Projective Hashing System. *A Smooth Projective Hash Function over a language $\mathfrak{L} \subset X$, is defined by five algorithms* $(\mathsf{Setup}, \mathsf{HashKG}, \mathsf{ProjKG}, \mathsf{Hash}, \mathsf{ProjHash})$*:*

- $\mathsf{Setup}(1^{\mathfrak{K}})$ *generates the global parameters* param *of the scheme, and the description of an* $\mathcal{NP}$ *language* $\mathfrak{L}$
- $\mathsf{HashKG}(\mathfrak{L}, \mathsf{param})$*, outputs a hashing key* hk *for the language* $\mathfrak{L}$*;*
- $\mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W)$*, derives the projection key* hp*, using the hashing key* hk*,*
- $\mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W)$*, outputs a hash value* v*, thanks to the hashing key* hk*, and* W*,*
- $\mathsf{ProjHash}(\mathsf{hp}, (\mathfrak{L}, \mathsf{param}), W, w)$*, outputs the hash value* v'*, thanks to* hp *and the witness* w *that* $W \in \mathfrak{L}$*.*

In the following, we consider $\mathfrak{L}$ as a hard-partitioned subset of X, *i.e.* it is computationally hard to distinguish a random element in $\mathfrak{L}$ from a random element in $X \setminus \mathfrak{L}$.

A Smooth Projective Hash Function SPHF should satisfy the following properties:

- *Correctness*: Let $W \in \mathfrak{L}$ and w a witness of this membership. Then, for all hashing keys hk and associated projection keys hp we have

$$\mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W) = \mathsf{ProjHash}(\mathsf{hp}, (\mathfrak{L}, \mathsf{param}), W, w).$$

- *Smoothness*: For all $W \in X \setminus \mathfrak{L}$ the following distributions are statistically indistinguishable:

$$\Delta_0 = \left\{ (\mathfrak{L}, \mathsf{param}, W, \mathsf{hp}, v) \;\middle|\; \begin{array}{l} \mathsf{param} = \mathsf{Setup}(1^{\mathfrak{K}}), \mathsf{hk} = \mathsf{HashKG}(\mathfrak{L}, \mathsf{param}), \\ \mathsf{hp} = \mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W), \\ v = \mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W) \in \mathbb{G} \end{array} \right\}$$

$$\Delta_1 = \left\{ (\mathfrak{L}, \mathsf{param}, W, \mathsf{hp}, v) \;\middle|\; \begin{array}{l} \mathsf{param} = \mathsf{Setup}(1^{\mathfrak{K}}), \mathsf{hk} = \mathsf{HashKG}(\mathfrak{L}, \mathsf{param}), \\ \mathsf{hp} = \mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W), v \overset{\$}{\leftarrow} \mathbb{G} \end{array} \right\}.$$

A third property called *Pseudo-Randomness*, is implied by the Smoothness on Hard Subset membership languages. If $W \in \mathfrak{L}$, then without a witness of membership the two previous distributions should remain computationally indistinguishable: for any adversary $\mathcal{A}$ within reasonable time the following advantage is negligible

$$\mathsf{Adv}^{\mathsf{pr}}_{\mathsf{SPHF},\mathcal{A}}(\mathfrak{K}) = |\Pr_{\Delta_1}[\mathcal{A}(\mathfrak{L}, \mathsf{param}, W, \mathsf{hp}, v) = 1] - \Pr_{\Delta_0}[\mathcal{A}(\mathfrak{L}, \mathsf{param}, W, \mathsf{hp}, v) = 1]|$$

In [14], the authors introduced a new notation for SPHF: for a language $\mathfrak{L}$, there exist a function Γ and a family of functions Θ, such that $\boldsymbol{u} \in \mathfrak{L}$, if and only if, $\Theta(\boldsymbol{u})$ is a linear combination $\boldsymbol{\lambda}$ of the rows of $\Gamma(\boldsymbol{u})$. We furthermore require that a user, who knows a witness of the membership $\boldsymbol{u} \in \mathfrak{L}$, can efficiently compute the linear combination $\boldsymbol{\lambda}$. The SPHF can now then be described as:

- $\mathsf{HashKG}(\mathfrak{L}, \mathsf{param})$, outputs a hashing key $\mathsf{hk} = \boldsymbol{\alpha}$ for the language $\mathfrak{L}$,
- $\mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), \boldsymbol{u})$, derives the projection key $\mathsf{hp} = \boldsymbol{\gamma}(\boldsymbol{u})$,
- $\mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), \boldsymbol{u})$, outputs a hash value $H = \Theta(\boldsymbol{u}) \odot \boldsymbol{\alpha}$,
- $\mathsf{ProjHash}(\mathsf{hp}, (\mathfrak{L}, \mathsf{param}), \boldsymbol{u}, \boldsymbol{\lambda})$, outputs the hash value $H' = \boldsymbol{\lambda} \odot \boldsymbol{\gamma}(\boldsymbol{u})$.

In the special case where $\mathsf{hp} = \boldsymbol{\gamma}(\boldsymbol{u}) = \boldsymbol{\gamma}$, we speak about KV-SPHF when the projection key can be given before seeing the word $\boldsymbol{u}$, and of CS-SPHF, when the projection key while independent of the word is given after seeing it. (In reference to [30,44] where those kinds of SPHF were first use). We give in Sect. 3.3 an example of KV-SPHF for Cramer-Shoup encryption, both in classical and new notations.

We will need a third property for our one-round PAKE protocol. This property, called strong pseudo-randomness in [14], is recalled in Appendix A.3 for lack of space.

2.3 Building Blocks

Decisional Diffie-Hellman (DDH). The Decisional Diffie-Hellman hypothesis says that in a multiplicative group $(p, \mathbb{G}, g)$ when we are given $(g^\lambda, g^\mu, g^\psi)$ for unknown random $\lambda, \mu, \psi \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, it is hard to decide whether $\psi = \lambda \times \mu$.

Pairing groups. Let GGen be a probabilistic polynomial time (PPT) algorithm that on input $1^{\mathfrak{K}}$ returns a description $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ of asymmetric pairing groups where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are cyclic groups of order p for a $\mathfrak{K}$-bit prime p, g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and $e : \mathbb{G}_1 \times \mathbb{G}_2$ is an efficiently computable (non-degenerated) bilinear map. Define $g_T := e(g_1, g_2)$, which is a generator in $\mathbb{G}_T$.

Matricial Notations. If $\boldsymbol{A} \in \mathbb{Z}_p^{(k+1)\times n}$ is a matrix, then $\overline{\boldsymbol{A}} \in \mathbb{Z}_p^{k\times n}$ denotes the upper matrix of $\boldsymbol{A}$ and $\underline{\boldsymbol{A}} \in \mathbb{Z}_p^{1\times n}$ denotes the last row of $\boldsymbol{A}$. We use classical notations from [36] for operations on vectors (. for the dot product and $\odot$ for the product component-wise). Concatenation of matrices having the same number of lines will be denoted by $\boldsymbol{A}||\boldsymbol{B}$ (where $a||b+c$ should be implicitly parsed as $a||(b+c)$).

We use implicit representation of group elements as introduced in [33]. For $s \in \{1, 2, T\}$ and $a \in \mathbb{Z}_p$ define $[a]_s = g_s^a \in \mathbb{G}_s$ as the *implicit representation* of a in $\mathbb{G}_s$ (we use $[a] = g^a \in \mathbb{G}$ if we consider a unique group). More generally, for a matrix $\boldsymbol{A} = (a_{ij}) \in \mathbb{Z}_p^{n\times m}$ we define $[\boldsymbol{A}]_s$ as the implicit representation of $\boldsymbol{A}$ in $\mathbb{G}_s$:

$$[\boldsymbol{A}]_s := \begin{pmatrix} g_s^{a_{11}} & \dots & g_s^{a_{1m}} \\ \\ g_s^{a_{n1}} & \dots & g_s^{a_{nm}} \end{pmatrix} \in \mathbb{G}_s^{n\times m}$$

We will always use this implicit notation of elements in $\mathbb{G}_s$, i.e., we let $[a]_s \in \mathbb{G}_s$ be an element in $\mathbb{G}_s$. Note that from $[a]_s \in \mathbb{G}_s$ it is generally hard to compute the value a (discrete logarithm problem in $\mathbb{G}_s$). Further, from $[b]_T \in \mathbb{G}_T$ it is hard to compute the value $[b]_1 \in \mathbb{G}_1$ and $[b]_2 \in \mathbb{G}_2$ (pairing inversion problem). Obviously, given $[a]_s \in \mathbb{G}_s$ and a scalar $x \in \mathbb{Z}_p$, one can efficiently compute $[ax]_s \in \mathbb{G}_s$. Further, given $[a]_1, [b]_2$ one can efficiently compute $[ab]_T$ using the pairing e. For $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}_p^k$ define $e([\boldsymbol{a}]_1, [\boldsymbol{b}]_2) := [\boldsymbol{a}^\top \boldsymbol{b}]_T \in \mathbb{G}_T$.

If $a \in \mathbb{Z}_p$, we define the $(k+1)$-vector: $\iota_s(a) := (1_s, \ldots, 1_s, [a]_s)$ (this notion can be implicitly extended to vectors $a \in \mathbb{Z}_p^n$), and the $k+1$ by $k+1$ matrix $\iota_T(a) := \begin{pmatrix} 1 & \ldots & 1 \\ \vdots & \ddots & 1 \\ 1 & 1 & a \end{pmatrix}$.

Assumptions. We recall the definition of the matrix Diffie-Hellman (MDDH) assumption [33].

Definition 2. Matrix Distribution. *Let $k \in \mathbb{N}$. We call $\mathcal{D}_k$ a matrix distribution if it outputs matrices in $\mathbb{Z}_p^{(k+1)\times k}$ of full rank k in polynomial time.*

Without loss of generality, we assume the first k rows of $\mathbf{A} \xleftarrow{\$} \mathcal{D}_k$ form an invertible matrix. The $\mathcal{D}_k$-Matrix Diffie-Hellman problem is to distinguish the two distributions $([\mathbf{A}], [\mathbf{A}\boldsymbol{w}])$ and $([\mathbf{A}], [\boldsymbol{u}])$ where $\mathbf{A} \xleftarrow{\$} \mathcal{D}_k$, $\boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^k$ and $\boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$.

Definition 3 ($\mathcal{D}_k$-Matrix Diffie-Hellman Assumption $\mathcal{D}_k$-MDDH). *Let $\mathcal{D}_k$ be a matrix distribution and $s \in \{1, 2, T\}$. We say that the $\mathcal{D}_k$-Matrix Diffie-Hellman ($\mathcal{D}_k$-MDDH) Assumption holds relative to GGen in group $\mathbb{G}_s$ if for all PPT adversaries $\mathcal{D}$,*

$$\begin{aligned}\mathbf{Adv}_{\mathcal{D}_k,\mathsf{GGen}}(\mathcal{D}) &:= |\Pr[\mathcal{D}(\mathcal{G}, [\boldsymbol{A}]_s, [\boldsymbol{A}\boldsymbol{w}]_s) = 1] - \Pr[\mathcal{D}(\mathcal{G}, [\boldsymbol{A}]_s, [\boldsymbol{u}]_s) = 1]| \\ &= \mathsf{negl}(\lambda),\end{aligned}$$

where the probability is taken over $\mathcal{G} \xleftarrow{\$} \mathsf{GGen}(1^\lambda)$, $\boldsymbol{A} \xleftarrow{\$} \mathcal{D}_k, \boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^k, \boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$.

For each $k \geq 1$, [33] specifies distributions $\mathcal{L}_k$, $\mathcal{U}_k$, ... such that the corresponding $\mathcal{D}_k$-MDDH assumption is the k-Linear assumption, the k-uniform and others. All assumptions are generically secure in bilinear groups and form a hierarchy of increasingly weaker assumptions. The distributions are exemplified for $k = 2$, where $a_1, \ldots, a_6 \xleftarrow{\$} \mathbb{Z}_p$.

$$\mathcal{L}_2 : \boldsymbol{A} = \begin{pmatrix} a_1 & 0 \\ 0 & a_2 \\ 1 & 1 \end{pmatrix} \quad \mathcal{U}_2 : \boldsymbol{A} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \\ a_5 & a_6 \end{pmatrix}.$$

It was also shown in [33] that $\mathcal{U}_k$-MDDH is implied by all other $\mathcal{D}_k$-MDDH assumptions. In the following, we write $k - \mathsf{MDDH}$ for $\mathcal{D}_k - \mathsf{MDDH}$.

Lemma 4 (Random self reducibility [33]). *For any matrix distribution $\mathcal{D}_k$, $\mathcal{D}_k$-MDDH is random self-reducible. In particular, for any $m \geq 1$,*

$$\mathbf{Adv}_{\mathcal{D}_k,\mathsf{GGen}}(\mathcal{D}) + \frac{1}{q-1} \geq \mathbf{Adv}^m_{\mathcal{D}_k,\mathsf{GGen}}(\mathcal{D}')$$

where $\mathbf{Adv}^m_{\mathcal{D}_k,\mathsf{GGen}}(\mathcal{D}') := \Pr[\mathcal{D}'(\mathcal{G}, [\boldsymbol{A}], [\boldsymbol{A}\,\boldsymbol{W}]) \Rightarrow 1] - \Pr[\mathcal{D}'(\mathcal{G}, [\boldsymbol{A}], [\boldsymbol{U}]) \Rightarrow 1]$, with $\mathcal{G} \leftarrow \mathsf{GGen}(1^\lambda)$, $\boldsymbol{A} \xleftarrow{\$} \mathcal{D}_k$, $\boldsymbol{W} \xleftarrow{\$} \mathbb{Z}_p^{k\times m}$, $\boldsymbol{U} \xleftarrow{\$} \mathbb{Z}_p^{(k+1)\times m}$.

Remark: It should be noted that $\mathcal{L}_1, \mathcal{L}_2$ are respectively the SXDH and DLin assumptions that we recall below for completeness.

Definition 5 (Decisional Linear (*DLin* [20])). *The Decisional Linear hypothesis says that in a multiplicative group* $(p, \mathbb{G}, g)$ *when we are given* $(g^\lambda, g^\mu, g^{\alpha\lambda}, g^{\beta\mu}, g^\psi)$ *for unknown random* $\alpha, \beta, \lambda, \mu \xleftarrow{\$} \mathbb{Z}_p$*, it is hard to decide whether* $\psi = \alpha + \beta$.

Definition 6 (Symmetric External Diffie Hellman (*SXDH* [9])). *This variant of DDH, used mostly in bilinear groups in which no computationally efficient homomorphism exists from* $\mathbb{G}_2$ *in* $\mathbb{G}_1$ *or* $\mathbb{G}_1$ *to* $\mathbb{G}_2$*, states that DDH is hard in both* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

Labelled Cramer-Shoup Encryption. We present here the well-known encryption schemes based on DDH, and we show in Sect. 4 how to extend it to $\mathcal{D}_k - \mathsf{MDDH}$. We focus on Cramer-Shoup [29] in all the following of the paper, but one easily obtains the same results on El Gamal $\mathsf{IND\text{-}CPA}$ scheme [32] by simply omitting the corresponding parts. We are going to rely on the $\mathsf{IND\text{-}CCA}$ property to be able to decrypt queries in the simulation.

Vanilla Cramer-Shoup Encryption. The Cramer-Shoup encryption scheme is an $\mathsf{IND\text{-}CCA}$ version of the ElGamal Encryption. We present it here as a labeled public-key encryption scheme, the classical version is done with $\ell = \emptyset$.

- $\mathsf{Setup}(1^\mathfrak{K})$ generates a group $\mathbb{G}$ of order p, with a generator g
- $\mathsf{KeyGen}(\mathsf{param})$ generates $(g_1, g_2) \xleftarrow{\$} \mathbb{G}^2$, $\mathsf{dk} = (x_1, x_2, y_1, y_2, z) \xleftarrow{\$} \mathbb{Z}_p^5$, and sets, $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, and $h = g_1^z$. It also chooses a Collision-Resistant hash function $\mathfrak{H}_K$ in a hash family $\mathcal{H}$ (or simply a Universal One-Way Hash Function). The encryption key is $\mathsf{ek} = (g_1, g_2, c, d, h, \mathfrak{H}_K)$.
- $\mathsf{Encrypt}(\ell, \mathsf{ek}, M; r)$, for a message $M \in \mathbb{G}$ and a random scalar $r \in \mathbb{Z}_p$, the ciphertext is $C = (\ell, \mathbf{u} = (g_1^r, g_2^r), e = M \cdot h^r, v = (cd^\xi)^r)$, where v is computed afterwards with $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$.
- $\mathsf{Decrypt}(\ell, \mathsf{dk}, C)$: one first computes $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$ and checks whether $u_1^{x_1 + \xi y_1} \cdot u_2^{x_2 + \xi y_2} \stackrel{?}{=} v$. If the equality holds, one computes $M = e/(u_1^z)$ and outputs M. Otherwise, one outputs $\perp$.

The security of the scheme is proven under the DDH assumption and the fact the hash function used is a Universal One-Way Hash Function.

In following work [30] they refined the proof, explaining that the scheme can be viewed as a 2-Universal Hash Proof on the language of valid Diffie Hellman tuple.

Vanilla Cramer-Shoup Encryption with Matricial Notations.

- $\mathsf{Setup}(1^\mathfrak{K})$ generates a group $\mathbb{G}$ of order p, with a generator g, with an underlying matrix assumption $\mathcal{D}_1$ using a base matrix $[\boldsymbol{A}] \in \mathbb{G}^{2 \times 1}$;
- $\mathsf{KeyGen}(\mathsf{param})$ generates $\mathsf{dk} = \boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{z} \xleftarrow{\$} \mathbb{Z}_p^2$ (with $\boldsymbol{t}_1 = (x_1, x_2)$, $\boldsymbol{t}_2 = (y_1, y_2)$ and $\boldsymbol{z} = (z, 1)$), and sets $c = \boldsymbol{t}_1 \boldsymbol{A}, d = \boldsymbol{t}_2 \boldsymbol{A}, h = \boldsymbol{z} \boldsymbol{A}$. It also chooses a hash function $\mathfrak{H}_K$ in a collision-resistant hash family $\mathcal{H}$ (or simply a Universal One-Way Hash Function). The encryption key is $\mathsf{ek} = ([A], [c], [d], [h], \mathfrak{H}_K)$.

- $\mathsf{Encrypt}(\ell, \mathsf{ek}, [m]; r)$, for a message $M = [m] \in \mathbb{G}$ and random scalar $r \xleftarrow{\$} \mathbb{Z}_p$, the ciphertext is $C = (\ell, \mathbf{u} = [\boldsymbol{A}r]), e = [\boldsymbol{h}r + m], v = [(\boldsymbol{c} + \boldsymbol{d} \odot \xi)\boldsymbol{r}]$, where v is computed afterwards with $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$.
- $\mathsf{Decrypt}(\ell, \mathsf{dk}, C)$: one first computes $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$ and checks whether v is consistent with $\boldsymbol{t}_1, \boldsymbol{t}_2$.
  If it is, one computes $M = [e - (\boldsymbol{uz})]$ and outputs M. Otherwise, one outputs $\perp$.

Groth-Sahai Proof System. Groth and Sahai [36] proposed non-interactive zero-knowledge proofs of satisfiability of certain equations over bilinear groups, called *pairing product equations*. Using as witness group elements (and scalars) which satisfy the equation, the prover starts with making commitments on them. To prove satisfiability of an equation (which is the statement of the proof), a Groth-Sahai proof uses these commitments and shows that the committed values satisfy the equation. The proof consists again of group elements and is verified by a pairing equation derived from the statement.

We refer to [36] for details of the Groth-Sahai proof system, and to [33] for the compatibility with the k-MDDH assumptions. More details can be found in the paper full version [15]. We are going to give a rough idea of the technique for SXDH.

To prove that committed variables satisfy a set of relations, the Groth-Sahai techniques require one commitment per variable and one proof element (made of a constant number of group elements) per relation. Such proofs are available for pairing-product relations and for multi-exponentiation equations.

When based on the SXDH assumption, the commitment key is of the form $\mathbf{u}_1 = (u_{1,1}, u_{1,2}), \mathbf{u}_2 = (u_{2,1}, u_{2,2}) \in \mathbb{G}_1^2$ and $\mathbf{v}_1 = (v_{1,1}, v_{1,2}), \mathbf{v}_2 = (v_{2,1}, v_{2,2}) \in \mathbb{G}_2^2$. We write

$$\mathbf{u} = \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{pmatrix} = \begin{pmatrix} u_{1,1} & u_{1,2} \\ u_{2,1} & u_{2,2} \end{pmatrix} \quad \text{and} \quad \mathbf{v} = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{pmatrix} = \begin{pmatrix} v_{1,1} & v_{1,2} \\ v_{2,1} & v_{2,2} \end{pmatrix}.$$

The Setup algorithm initializes the parameters as follows: $\mathbf{u}_1 = (g_1, u)$ with $u = g_1^\lambda$ and $\mathbf{u}_2 = \mathbf{u}_1{}^\mu$ with $\lambda, \mu \xleftarrow{\$} \mathbb{Z}_p^*$, which means that $\mathbf{u}$ is a Diffie-Hellman tuple in $\mathbb{G}_1$, since $\mathbf{u}_1 = (g_1, g_1^\lambda)$ and $\mathbf{u}_2 = (g_1^\mu, g_1^{\lambda\mu})$. The TSetup algorithm will use instead $\mathbf{u}_2 = \mathbf{u}_1{}^\mu \odot (1, g_1)^{-1}$: $\mathbf{u}_1 = (g_1, g_1^\lambda)$ and $\mathbf{u}_2 = (g_1^\mu, g_1^{\lambda\mu - 1})$. And it is the same in $\mathbb{G}_2$ for $\mathbf{v}$. Depending on the definition of $\mathbf{u}_2, \mathbf{v}_2$, this commitment can be either perfectly hiding or perfectly binding. The two parameter initializations are indistinguishable under the SXDH assumption.

To commit to $X \in \mathbb{G}_1$, one chooses randomness $s_1, s_2 \in \mathbb{Z}_p$ and sets $\mathcal{C}(X) = (1, X) \odot \mathbf{u}_1^{s_1} \odot \mathbf{u}_2^{s_2} = (1, X) \odot (u_{1,1}^{s_1}, u_{1,2}^{s_1}) \odot (u_{2,1}^{s_2}, u_{2,2}^{s_2}) = (u_{1,1}^{s_1} \cdot u_{2,1}^{s_2}, X \cdot u_{1,2}^{s_1} \cdot u_{2,2}^{s_2})$. Similarly, one can commit to element in $\mathbb{G}_2$ and scalars in $\mathbb{Z}_p$. The committed group elements can be extracted if $\mathbf{u}_2$ is linearly dependant of $\mathbf{u}_1$ by knowing the discrete logarithm x_1 between $\mathbf{u}_{1,1}$ and $\mathbf{u}_{2,2}$: $c_2/(c_1^{x_1}) = X$.

In the following we are going to focus on proof of linear multi-scalar exponentiation in $\mathbb{G}_1$, that is to say we are going to prove equations of the form $\prod_i A_i^{y_i} = A$ where A_i are public elements in $\mathbb{G}_1$ and y_i are going to be scalars committed into $\mathbb{G}_2$.

2.4 Protocols

UC Framework. The goal of this simulation-based model [22] is to ensure that UC-secure protocols will continue to behave in the ideal way even if executed in a concurrent way in arbitrary environments. Due to lack of space, a short introduction to the UC framework is given in the paper full version [15].

Oblivious Transfer and Password-Authenticated Key-Exchange. The security properties for these two protocols are given in terms of ideal functionalities in the paper full version [15].

3 Structure-Preserving Smooth Projective Hashing

3.1 Definition

In this section, we are now going to narrow the classical definition of Smooth Projective Hash Functions to what we are going to name Structure-Preserving Smooth Projective Hash Functions, in which both words, witnesses and projection keys are group elements.

Since witnesses now become group elements, this allows a full compatibility with Groth and Sahai methodology [36], such that for instance possessing a Non-Interactive Zero-Knowledge Proof of Knowledge can become new witnesses of our SP-SPHF, leading to interesting applications, as described later on.

As we are in the context of Structure Preserving cryptography, we assume the existence of a (prime order) bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, g_1, g_2, \mathbb{G}_T, e)$, and consider Languages (sets of elements) $\mathfrak{L}$ defined over this group. The hash space is usually $\mathbb{G}_T$, the projection key space a group $\mathbb{G}_1^m \times \mathbb{G}_2^n$ and the witness space a group $\mathbb{G}_1^n \times \mathbb{G}_2^m$.

Definition 7 Structure-Preserving Smooth Projective Hash Functions. *A Structure-Preserving Smooth Projective Hash Function over a language $\mathfrak{L} \subset X$ onto a set $\mathcal{H}$ is defined by 4 algorithms* $(\mathsf{HashKG}, \mathsf{ProjKG}, \mathsf{Hash}, \mathsf{ProjHash})$*:*

- $\mathsf{HashKG}(\mathfrak{L}, \mathsf{param})$*, outputs a hashing key* hk *for the language* $\mathfrak{L}$*;*
- $\mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W)$*, derives the projection key* hp *thanks to the hashing key* hk*.*
- $\mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W)$*, outputs a hash value $H \in \mathcal{H}$, thanks to the hashing key* hk*, and W*
- $\mathsf{ProjHash}(\mathsf{hp}, (\mathfrak{L}, \mathsf{param}), W, w)$*, outputs the value $H' \in \mathcal{H}$, thanks to* hp *and the witness w that $W \in \mathfrak{L}$.*

Remark 8. We stress that, contrarily to classical SPHF, both hp, W and more importantly w are base group elements, and so live in the same space.

3.2 Properties

Properties are then inherited by those of classical Smooth Projective Hash Functions.

- *Correctness*: On honest computations with (W, w) compatible with $\mathfrak{L}$, we have $\mathsf{ProjHash}(\mathsf{hp}, (\mathfrak{L}, \mathsf{param}), W, w) = \mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W)$.
- *Smoothness*: For all $W \in X \setminus \mathfrak{L}$ the following distributions are statistically indistinguishable:

$$\Delta_0 = \left\{ (\mathfrak{L}, \mathsf{param}, W, \mathsf{hp}, v) \;\middle|\; \begin{array}{l} \mathsf{param} = \mathsf{Setup}(1^{\mathfrak{K}}), \mathsf{hk} = \mathsf{HashKG}(\mathfrak{L}, \mathsf{param}), \\ \mathsf{hp} = \mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W), \\ v = \mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W) \in \mathbb{G}_T \end{array} \right\}$$

$$\Delta_1 = \left\{ (\mathfrak{L}, \mathsf{param}, W, \mathsf{hp}, v) \;\middle|\; \begin{array}{l} \mathsf{param} = \mathsf{Setup}(1^{\mathfrak{K}}), \mathsf{hk} = \mathsf{HashKG}(\mathfrak{L}, \mathsf{param}), \\ \mathsf{hp} = \mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W), v \xleftarrow{\$} \mathbb{G}_T \end{array} \right\}.$$

This is formalized by

$$\mathsf{Adv}^{\mathsf{smooth}}_{\mathsf{SPHF}}(\mathfrak{K}) = \sum_{V \in \mathbb{G}} \left| \Pr_{\Delta_1}[v = V] - \Pr_{\Delta_0}[v = V] \right| \text{ is negligible.}$$

As usual, a derivative property called *Pseudo-Randomnness*, says the previous distribution are computationally indistinguishable from words in the language while the witnesses remain unknown. This is implied by the Smoothness on Hard Subset membership languages.

3.3 Retro-Compatibility

Constructing SP-SPHF is not that hard of a task. A first naive approach allows to transform every pairing-less SPHF into a SP-SPHF in a bilinear setting. It should be noted that while the resulting Hash/ProjHash values live in the target group, nearly all use cases encourage to use a proper hash function on them before computing anything using their value, hence the communication cost would remain the same. (Only applications where one of the party has to provide an additional proof that the ProjHash was honestly computed might be lost, but besides proof of negativity from [18], this never arises.)

To this goal, simply given a new generator $f \in \mathbb{G}_2$, and a scalar witness vector λ, one generates the new witness vector $\Lambda = [f \odot \boldsymbol{\lambda}]_2$. Words and projection keys belong to $\mathbb{G}_1$, and hash values to $\mathbb{G}_T$. Any SPHF can thus be transformed into an SP-SPHF in the following way:

	SPHF	SP-SPHF
Word $\boldsymbol{u}$	$[\boldsymbol{\lambda} \odot \Gamma(\boldsymbol{u})]_1$	$[\boldsymbol{\lambda} \odot \Gamma(\boldsymbol{u})]_1$
Witness w	$\boldsymbol{\lambda}$	$\boldsymbol{\Lambda} = [f \odot \boldsymbol{\lambda}]_2$
hk	$\boldsymbol{\alpha}$	$\boldsymbol{\alpha}$
$\mathsf{hp} = [\gamma(\boldsymbol{u})]_1$	$[\Gamma(\boldsymbol{u}) \odot \boldsymbol{\alpha}]_1$	$[\Gamma(\boldsymbol{u}) \odot \boldsymbol{\alpha}]_1$
$\mathsf{Hash}(\mathsf{hk}, \boldsymbol{u})$	$[\Theta(\boldsymbol{u}) \odot \boldsymbol{\alpha}]_1$	$[f \odot \Theta(\boldsymbol{u}) \odot \boldsymbol{\alpha}]_T$
$\mathsf{ProjHash}(\mathsf{hp}, \boldsymbol{u}, w)$	$[\boldsymbol{\lambda} \odot \gamma(\boldsymbol{u})]_1$	$[\boldsymbol{\Lambda} \odot \gamma(\boldsymbol{u})]_T$

- *Correctness* is inherited for words in $\mathcal{L}$ as this reduces to computing the same values but in $\mathbb{G}_T$.
- *Smoothness*: For words outside the language, the projection keys, remaining unchanged, do not reveal new information, so that the smoothness will remain preserved.
- *Pseudo-Randomness*: Without any witness, words inside the language are indistinguishable from words outside the language (under the subgroup decision assumption), hence the hash values remain pseudo-random.

It should be noted that in case this does not weaken the subgroup decision assumption (k-MDDH in the following) linked to the original language, one can set $\mathbb{G}_1 = \mathbb{G}_2$.

We give in Fig. 1 two examples of regular Smooth Projective Hash Functions on Diffie-Hellman and Cramer-Shoup encryption of M, where $\alpha = \mathcal{H}(\mathbf{u}, e)$, and their counterparts with SP-SPHF. ElGamal being a simplification of Cramer-Shoup, we skip the description of the associated SP-SPHF. We also give in Fig. 2 the matricial version of Cramer-Shoup encryption, in which we denote by C' the Cramer-Shoup encryption C of M in which we removed M.

	SPHF	SP-SPHF
DH	h^r, g^r	h^r, g^r
Witness w	r	g_2^r
hk	λ, μ	λ, μ
hp	$h^\lambda g^\mu$	$h^\lambda g^\mu$
$\mathsf{Hash}(\mathsf{hk}, \boldsymbol{u})$	$(h^r)^\lambda (g^r)^\mu$	$e((h^r)^\lambda (g^r)^\mu, g_2)$
$\mathsf{ProjHash}(\mathsf{hp}, \boldsymbol{u}, w)$	hp^r	$e(\mathsf{hp}, g_2^r)$
CS(M;r)	$h^r M, f^r, g^r, (cd^\alpha)^r$	$h^r M, f^r, g^r, (cd^\alpha)^r$
Witness w	r	g_2^r
hk	$\lambda_1, \lambda_2, \mu, \nu, \eta$	$\lambda_1, \lambda_2, \mu, \nu, \eta$
hp	$h^{\lambda_1} f^\mu g^\nu c^\eta, h^{\lambda_2} d^\nu$	$h^{\lambda_1} f^\mu g^\nu c^\eta, h^{\lambda_2} d^\nu$
$\mathsf{Hash}(\mathsf{hk}, \boldsymbol{u})$	$H = (h^r)^{\lambda_1 + \alpha\lambda_2} (f^r)^\mu (g^r)^\nu ((cd^\alpha)^r)^\mu$	$e(H, g_2)$
$\mathsf{ProjHash}(\mathsf{hp}, \boldsymbol{u}, w)$	$(\mathsf{hp}_1 \mathsf{hp}_2^\alpha)^r$	$e(\mathsf{hp}_1 \mathsf{hp}_2^\alpha, g_2^r)$
(with $\mathsf{hp} = (\mathsf{hp}_1, \mathsf{hp}_2)$)		

Fig. 1. Example of conversion of classical SPHF into SP-SPHF

	SPHF	SP-SPHF
CS(M;r)	$[hr + M, \boldsymbol{A}r, (c + d\alpha)r]$	$[hr + M, \boldsymbol{A}r, (c + d\alpha)r]_1$
$\boldsymbol{B} : \begin{pmatrix} h \\ f \\ g \\ c \end{pmatrix}$	$\left[\boldsymbol{B}r + \begin{pmatrix} 0 \\ 0 \\ 0 \\ d \end{pmatrix} \alpha r + \begin{pmatrix} M \\ 0 \\ 0 \\ 0 \end{pmatrix}\right]$	$\left[\boldsymbol{B}r + \begin{pmatrix} 0 \\ 0 \\ 0 \\ d \end{pmatrix} \alpha r + \begin{pmatrix} M \\ 0 \\ 0 \\ 0 \end{pmatrix}\right]_1$
Witness w	r	$[r]_2$
hk	$\lambda_1, \lambda_2, \mu, \nu, \eta$	$\lambda_1, \lambda_2, \mu, \nu, \eta$
hp	$\left[\mathsf{hp}_1 = \left(\lambda_1\ \mu\ \nu\ \eta\right) \boldsymbol{B}\right]$, $\left[\mathsf{hp}_2 = \left(\lambda_2\ 0\ 0\ \eta\right) \begin{pmatrix} h \\ 0 \\ 0 \\ d \end{pmatrix}\right]$	$[\mathsf{hp}_1]_1, [\mathsf{hp}_2]_1$
$\mathsf{Hash}(\mathsf{hk}, \boldsymbol{u})$	$\left[\left(\lambda_1 + \alpha\lambda_2\ \mu\ \nu\ \eta\right)(C')\right]$	$\left[\left(\lambda_1 + \alpha\lambda_2\ \mu\ \nu\ \eta\right)(C')\right]_T$
$\mathsf{ProjHash}(\mathsf{hp}, \boldsymbol{u}, w)$	$[(\mathsf{hp}_1 + \alpha\mathsf{hp}_2)r]$	$[(\mathsf{hp}_1 + \alpha\mathsf{hp}_2)r]_2$

Fig. 2. Example of conversion of SPHF into SP-SPHF (matricial notations)

3.4 Possible Applications

Nearly Constant 1-out-of-m Oblivious Transfer Using FLM. Recent pairing-based constructions [1,28] of Oblivious Transfer use SPHF to mask each line of a database with the hash value of as SPHF on the language corresponding to the first flow being a commitment of the said line.

Sadly, those constructions require special UC commitment on scalars, with equivocation and extraction capacities, leading to very inefficient constructions. In 2011, [34] proposed a UC commitment, whose decommitment operation is done via group elements. In Sect. 5, we are going to show how to combine the existing constructions with this efficient commitment using SP-SPHF, in order to obtain a very efficient round-optimal where there is no longer a growing overhead due to the commitment. As a side result, we show how to generalize the FLM commitment to any MDDH assumption.

Round-Optimal Password Authenticated Key Exchange with Adaptive Corruptions. Recent developments around SPHF-based PAKE have either lead to Round-Optimal PAKE in the BPR model [11], or with static corruptions [14,44]. In order to achieve round-optimality, [1] needs to do a bit-per-bit commitment of the password, inducing a communication cost proportional to the maximum password length.

In the following, we show how to take advantage of the SP-SPHF constructed on the FLM commitment to propose a One-Round PAKE UC secure against adaptive adversaries, providing a constant communication cost.

Using a ZKPK as a witness, Anonymous Credentials. Previous applications allow more efficient instantiations of protocols already using scalar-based SPHF. However, one can imagine additional scenarios, where a scalar based approach may not be possible, due to the inherent nature of the witness used.

For example, one should consider a strong authentication scenario, in which each user possesses an identifier delivered by an authority, and a certification on a commitment to this identifier, together with a proof of knowledge that this commitment is indeed a commitment to this identifier. (Such scenario can be transposed to the delivery of a Social Security Number, where a standalone SSN may not be that useful, but a SSN officially linked to someone is a sensitive information that should be hidden.) In this scenario, a user who wants to access his record on a government service where he is already registered, should give the certificate, and then would use an implicit proof that this corresponds to his identifier. With our technique, the server would neither learn the certificate in the clear nor the user identifier (if he did not possess it earlier), and the user would be able to authenticate only if his certificate is indeed on his committed identifier.

In our scenario, we could even add an additional step, such that Alice does not interact directly with Bob but can instead use a pawn named Carol. She could send to Carol a commitment to the signature on her identity, prove in a black box way that it is a valid signature on an identity, and let Carol do the interaction on her behalf. For example, to allow a medical practitioner to access some subpart of her medical record concerning on ongoing treatment, in this case, Carol would need to anonymously prove to the server that she is indeed a registered medical practitioner, and that Alice has given her access to her data.

4 Encryption and Commitment Schemes Based on k-MDDH

4.1 k-MDDH Cramer-Shoup Encryption

In this paper, we supersede the previous constructions with a k-MDDH based one:

- $\mathsf{Setup}(1^{\mathfrak{K}})$ generates a group $\mathbb{G}$ of order p, with an underlying matrix assumption using a base matrix $[\boldsymbol{A}] \in \mathbb{G}^{k+1 \times k}$;
- $\mathsf{KeyGen}(\mathsf{param})$ generates $\mathsf{dk} = \boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{z} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$, and sets, $\boldsymbol{c} = \boldsymbol{t}_1 \boldsymbol{A} \in \mathbb{Z}_p^k, \boldsymbol{d} = \boldsymbol{t}_2 \boldsymbol{A} \in \mathbb{Z}_p^k, \boldsymbol{h} = \boldsymbol{z} \boldsymbol{A} \in \mathbb{Z}_p^k$. It also chooses a hash function $\mathfrak{H}_K$ in a collision-resistant hash family $\mathcal{H}$ (or simply a Universal One-Way Hash Function). The encryption key is $\mathsf{ek} = ([\boldsymbol{c}], [\boldsymbol{d}], [\boldsymbol{h}], [\boldsymbol{A}], \mathfrak{H}_K)$.
- $\mathsf{Encrypt}(\ell, \mathsf{ek}, [m]; \boldsymbol{r})$, for a message $M = [m] \in \mathbb{G}$ and random scalars $\boldsymbol{r} \xleftarrow{\$} \mathbb{Z}_p^k$, the ciphertext is $C = (\mathbf{u} = [\boldsymbol{A}\boldsymbol{r}]), e = [\boldsymbol{h}\boldsymbol{r} + m], v = [(\boldsymbol{c} + \boldsymbol{d} \odot \xi)\boldsymbol{r}]_1$, where v is computed afterwards with $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$.
- $\mathsf{Decrypt}(\ell, \mathsf{dk}, C)$: one first computes $\xi = \mathfrak{H}_K(\ell, \mathbf{u}, e)$ and checks whether v is consistent with $\boldsymbol{t}_1, \boldsymbol{t}_2$.
  If it is, one computes $M = [e - (\boldsymbol{u}\boldsymbol{z})]$ and outputs M. Otherwise, one outputs $\perp$.

Theorem 9. *The k-MDDH Cramer-Shoup Encryption is IND-CCA 2 under k-MDDH assumption and the collision resistance (universal one-wayness) of the Hash Family.*

Proof. To sketch the proof of the theorem, one should remember that the original proof articulate around three main cases noting $\ell, \mathbf{u}, e, v$ the challenge query, and $\ell', \mathbf{u}, e', v'$ the current decryption query:

- $(\ell, \mathbf{u}, e) = (\ell', \mathbf{u}', e')$ but $v \neq v'$. This will fail as v is computed to be the correct checksum, hence we can directly reject the decryption query.
- $(\ell, \mathbf{u}, e) \neq (\ell', \mathbf{u}', e')$ but $\xi = \xi'$, this is a collision on the Hash Function.
- $(\ell, \mathbf{u}, e, v) \neq (\ell, \mathbf{u}, e, v)$ and $\xi \neq \xi'$. This is the argument revolving around the 2-Universality of the Hash Proof system defined by $\boldsymbol{c}, \boldsymbol{d}$. $\boldsymbol{c}, \boldsymbol{d}$ gives $2k$ equations in $2k+2$ variables, hence answering decryption queries always in the same span can give at most 1 more equation leaving at least 1 degree of freedom in the system. □

Structure-Preserving Smooth Projective Hash Function

For ease of readability we are going to set $\boldsymbol{B} = \left[\begin{pmatrix} h \\ \boldsymbol{A} \\ \boldsymbol{c} \end{pmatrix}\right]$ and $\boldsymbol{D} = \left[\begin{pmatrix} 0 \\ \vdots \\ \boldsymbol{d} \end{pmatrix}\right]$, and write $C' = [\boldsymbol{B}\boldsymbol{r} + \xi \boldsymbol{D}\boldsymbol{r}]_1$ the ciphertext without the message M.

- $\mathsf{HashKG}(\mathfrak{L}, \mathsf{param})$, chooses $\Lambda \xleftarrow{\$} \mathbb{Z}_p^{(k+2)\times 1}$, $\lambda \xleftarrow{\$} \mathbb{Z}_p$ and sets
$$\mathsf{hk}_1 = \Lambda, \mathsf{hk}_2 = \begin{pmatrix} \lambda \\ \boldsymbol{0} \\ \Lambda_{k+2} \end{pmatrix};$$
- $\mathsf{ProjKG}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W)$, outputs $\mathsf{hp}_1 = \mathsf{hk}_1^\top \boldsymbol{B}, \mathsf{hp}_2 = \mathsf{hk}_2^\top \begin{pmatrix} h \\ \boldsymbol{0} \\ d \end{pmatrix}$;
- $\mathsf{Hash}(\mathsf{hk}, (\mathfrak{L}, \mathsf{param}), W)$, outputs a hash value $H = [(\mathsf{hk}_1 + \xi \mathsf{hk}_2)^\top C']_T$;
- $\mathsf{ProjHash}(\mathsf{hp}, (\mathfrak{L}, \mathsf{param}), W, w)$, outputs the value $H' = [(\mathsf{hp}_1 + \xi \mathsf{hp}_2)\boldsymbol{r}]_T$.

The Smoothness comes inherently from the fact that we have $2k+2$ unknowns in hk while hp gives at most $2k$ equations. Hence an adversary has a negligible chance to find the real values.

4.2 A Universally Composable Commitment with Adaptive Security Based on MDDH

We first show how to simply generalize FLM's commitment [34] from DLin to $\mathbf{k}$-MDDH.

FLM's Commitment on DLin. At Asiacrypt 2011, Fischlin, Libert and Manulis presented a universally composable commitment [34] with adaptive security based on the Decision Linear assumption [20]. We show here how to generalize their scheme to the Matrix Decisional Diffie-Hellman assumption from [33] and recalled in Sect. 2. We first start by recalling their original scheme. Note that sid denotes the session identifier and cid the commitment identifier and that the combination $(\mathsf{sid}, \mathsf{cid})$ is globally unique, as in [34, 37].

- **CRS Generation:** $\mathsf{SetupCom}(1^{\mathfrak{K}})$ chooses a bilinear group $(p, \mathbb{G}, \mathbb{G}_T)$ of order $p > 2^{\mathfrak{K}}$, a generator g of $\mathbb{G}$, and sets $g_1 = g^{\alpha_1}$ and $g_2 = g^{\alpha_2}$ with random $\alpha_1, \alpha_2 \in \mathbb{Z}_p^*$. It defines the vectors $\mathbf{g_1} = (g_1, 1, g)$, $\mathbf{g_2} = (1, g_2, g)$ and $\mathbf{g_3} = \mathbf{g_1}^{\xi_1}\mathbf{g_2}^{\xi_2}$ with random $\xi_1, \xi_2 \in \mathbb{Z}_p^*$, which form a Groth-Sahai CRS $\mathbf{g} = (\mathbf{g_1}, \mathbf{g_2}, \mathbf{g_3})$ for the perfect soundness setting. It then chooses a collision-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p$ and generates a public key $\mathsf{pk} = (X_1, \ldots, X_6)$ for the linear Cramer-Shoup encryption scheme. The CRS consists of $\mathsf{crs} = (\mathfrak{K}, \mathbb{G}, \mathbb{G}_T, g, \mathbf{g}, H, \mathsf{pk})$.
- **Commitment algorithm:** $\mathsf{Com}(\mathsf{crs}, M, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$, to commit to message $M \in \mathbb{G}$ for party P_j, party P_i parses crs as $(\mathfrak{K}, \mathbb{G}, \mathbb{G}_T, g, \mathbf{g}, H, \mathsf{pk})$ and conducts the following steps:
    - It chooses random exponents r, s in $\mathbb{Z}_p$ and computes a linear Cramer-Shoup encryption $\psi_{CS} = (U_1, U_2, U_3, U_4, U_5)$ of $M \in \mathbb{G}$ under the label $\ell = P_i \| \mathsf{sid} \| \mathsf{cid}$ and the public key pk.
    - It generates a NIZK proof $\pi_{val-enc}$ that $\psi_{CS} = (U_1, U_2, U_3, U_4, U_5)$ is indeed a valid encryption of $M \in \mathbb{G}$. This requires to commit to exponents r, s and prove that these exponents satisfy the multi-exponentiation equations $U_1 = {g_1}^r$, $U_2 = {g_2}^s$, $U_3 = g^{r+s}$, $U_4/M = {X_5}^r {X_6}^s$ and $U_5 = (X_1 {X_3}^{\alpha})^r \cdot (X_2 {X_4}^{\alpha})^s$.
    - P_i erases (r, s) after the generation of $\pi_{val-enc}$ but retains the $D_M = \pi_{val-enc}$.

  The commitment is ψ_{CS}.
- **Verification algorithm:** the algorithm $\mathsf{VerCom}(\mathsf{crs}, M, D_M, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$ checks the proof $\pi_{val-enc}$ and ignores the opening if the verification fails.
- **Opening algorithm:** $\mathsf{OpenCom}(\mathsf{crs}, M, D_M, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$ reveals M and $D_M = \pi_{val-enc}$ to P_j.

The extraction algorithm uses Cramer-Shoup decryption algorithm, while the equivocation uses the simulator of the NIZK. It is shown in [1] that the $\mathsf{IND\text{-}CCA}$ security notion for C and the computational soundness of π make it strongly-binding-extractable, while the $\mathsf{IND\text{-}CCA}$ security notion and the zero-knowledge property of the NIZK provide the strong-simulation-indistinguishability.

Moving to k-MDDH: We now show how to extend the previous commitment to the k-MDDH assumption. Compared to the original version of the commitment, we split the proof $\pi_{val-enc}$ into its two parts: the NIZK proof denoted here as $[\boldsymbol{\Pi}]_1$ is still revealed during the opening algorithm, while the Groth-Sahai commitment $[\boldsymbol{R}]_2$ of the randomness $\mathbf{r}$ of the Cramer-Shoup encryption is sent during the commitment phase. Furthermore, since the hash value in the Cramer Shoup encryption is used to link the commitment with the session, we include this value $[\boldsymbol{R}]_2$ to the label, in order to ensure that this extra commitment information given with the ciphertext is the original one. We refer the reader to the original security proof in [34, Theorem 1], which remains exactly the same, since this additional commitment provides no information (either computationally or perfectly, depending on the CRS), and since the commitment $[\boldsymbol{R}]_2$ is not modified in the equivocation step (only the value $[\boldsymbol{\Pi}]_1$ is changed).

- **CRS Generation:** algorithm $\mathsf{SetupCom}(1^{\mathfrak{K}})$ chooses a bilinear asymmetric group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ of order $p > 2^{\mathfrak{K}}$, and a set of generators $[\boldsymbol{A}]_1$ corresponding to the underlying matrix assumption.

  As explained in [33], following their notations, one can define a Groth-Sahai CRS by picking $\boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$, and setting $[\boldsymbol{U}]_2 = [\boldsymbol{B}||\boldsymbol{B}\boldsymbol{w}]_2$ for a hiding CRS, and $[\boldsymbol{B}||\boldsymbol{B}\boldsymbol{w} + (0||z)^\top]_2$ otherwise, where $[\boldsymbol{B}]_2$ is an k-MDDH basis, and $\boldsymbol{w}, z$ are the elements defining the challenge vector.

  For the Cramer-Shoup like CCA-2 encryption, one additionally picks $\boldsymbol{t}_1, \boldsymbol{t}_2$, $\boldsymbol{z} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$, and a Universal One-Way Hash Function $\mathcal{H}$ and sets $[\boldsymbol{h}]_1 = [\boldsymbol{z} \cdot \boldsymbol{A}]_1$, $[\boldsymbol{c}]_1 = [\boldsymbol{t}_1 \boldsymbol{A}]_1, [\boldsymbol{d}]_1 = [\boldsymbol{t}_2 \boldsymbol{A}]_1$.

  The CRS consists of $\mathsf{crs} = (\mathfrak{K}, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, [\boldsymbol{A}]_1 \in \mathbb{G}_1^{k \times k+1}, [\boldsymbol{U}]_2, [\boldsymbol{h}]_1 \in \mathbb{G}_1^k, [\boldsymbol{c}]_1 \in \mathbb{G}_1^k, [\boldsymbol{d}]_1 \in \mathbb{G}_1^k, \mathcal{H})$.
- **Commitment algorithm:** $\mathsf{Com}(\mathsf{crs}, M, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$, to commit to message $M \in \mathbb{G}_1$ for party P_j, party P_i conducts the following steps:
  - It chooses random exponents $\boldsymbol{r}$ in $\mathbb{Z}_p^k$ and commits to $\boldsymbol{r}$ in $[\boldsymbol{R}]_2$ with randomness $\rho \xleftarrow{\$} \mathbb{Z}_p^{k \times k+1}$, setting $[\boldsymbol{R}]_2 = [\boldsymbol{U}\boldsymbol{\rho} + \iota_2(\boldsymbol{r})]_2 \in \mathbb{G}_2^{k \times k+1}$. It also computes a Cramer-Shoup encryption $\psi_{CS} = [\boldsymbol{C}]_1$ of $M \in \mathbb{G}_1$ under the label $\ell = P_i||\mathsf{sid}||\mathsf{cid}$ and the public key pk:

    $$[\boldsymbol{C}]_1 = [\boldsymbol{A}\boldsymbol{r}||\boldsymbol{h}\boldsymbol{r} + M||(\boldsymbol{c} + \boldsymbol{d} \odot \mathcal{H}(\ell||\boldsymbol{C}_1||\boldsymbol{C}_2||\boldsymbol{R}))\boldsymbol{r}]_1 = [\boldsymbol{C}_1||\boldsymbol{C}_2||\boldsymbol{C}_3]_1$$

    For simplicity we write $\ell' = \ell||[\boldsymbol{C}_1]_1||[\boldsymbol{C}_2]_1||[\boldsymbol{R}]_2$.
  - It generates a NIZK proof $D_M = [\boldsymbol{\Pi}]_1$ that ψ_{CS} is indeed a valid encryption of $M \in \mathbb{G}_1$ for the committed $\boldsymbol{r}$ in $[\boldsymbol{R}]_2$. This requires to prove that these exponents satisfy the multi-exponentiation equations:

    $$[\boldsymbol{C}_1]_1 = [\boldsymbol{A}\boldsymbol{r}]_1, [\boldsymbol{C}_2 - M]_1 = [\boldsymbol{h}\boldsymbol{r}]_1, [\boldsymbol{C}_3 = (\boldsymbol{c} + \boldsymbol{d} \odot \mathcal{H}(\ell'))\boldsymbol{r}]_1$$

    The associated proof is then $[\boldsymbol{\Pi}]_1 = [\boldsymbol{\rho}^\top(\boldsymbol{A}||\boldsymbol{h}||\boldsymbol{c} + \boldsymbol{d} \odot \mathcal{H}(\ell'))]_1$.
  - P_i erases r after the generation of $[\boldsymbol{R}]_2$ and $[\boldsymbol{\Pi}]_1$ but retains $D_M = [\boldsymbol{\Pi}]_1$.

  The commitment is $([\boldsymbol{C}]_1, [\boldsymbol{R}]_2)$.
- **Verification algorithm:** the algorithm $\mathsf{VerCom}(\mathsf{crs}, M, D_M, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$ checks the consistency of the proof $\pi_{val-enc}$ with respect to $[\boldsymbol{C}]_1$ and $[\boldsymbol{R}]_2$. and ignores the opening if the verification fails.
- **Opening algorithm:** $\mathsf{OpenCom}(\mathsf{crs}, M, D_M, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$ reveals M and $D_M = [\Pi]_1$ to P_j.

One can easily see that $[\boldsymbol{C}_3]_1$ is the projective hash computation of a 2-universal hash proof on the language "$[\boldsymbol{C}_1]_1$ in the span of $\boldsymbol{A}$", with $[\boldsymbol{C}_2]_1$ being an additional term that uses the same witness to mask the committed message, so that $[\boldsymbol{C}]_1$ is a proper generalization of the Cramer-Shoup CCA-2 encryption. Details on the k-MDDH Groth-Sahai proofs are given in the paper full version [15].

It is thus easy to see that this commitment is indeed a generalization of the FLM non-interactive UC commitment with adaptive corruption under reliable erasures (in which we switched the CRS, the Cramer-Shoup encryption and the Groth-Sahai proof in the k-MDDH setting).

4.3 A Structure-Preserving Smooth Projective Hash Function Associated with This Commitment

Structure-Preserving Smooth Projective Hash Function. We now want to supersede the verification equation of the commitment by a smooth projective hash function providing implicit decommitment, simply using the proof as a witness. We consider the language of the valid encryptions of M using a random r which is committed into $[\boldsymbol{R}]_2$:

$$\begin{aligned}\mathcal{L}_M = \{[\boldsymbol{C}]_1 \mid \exists r \exists \rho \text{ such that } & [\boldsymbol{R}]_2 = [\boldsymbol{U}\boldsymbol{\rho} + \iota_2(\boldsymbol{r})]_2 \\ \text{and } & [\boldsymbol{C}]_1 = [\boldsymbol{A}\boldsymbol{r} || \boldsymbol{h}\boldsymbol{r} + M || (\boldsymbol{c} + \boldsymbol{d} \odot \mathcal{H}(\ell || \boldsymbol{C}_1 || \boldsymbol{C}_2 || \boldsymbol{R}))\boldsymbol{r}]_1\}\end{aligned}$$

The verifier picks a random $\mathsf{hk} = \alpha \xleftarrow{\$} \mathbb{Z}_p^{k+3 \times k+1}$ and sets $\mathsf{hp} = [\alpha \odot \boldsymbol{U}]_2$. On one side, the verifier then computes:

$$\mathsf{Hash}(\mathsf{hk}, ([\boldsymbol{C}]_1, [\boldsymbol{R}]_2)) = [\alpha \odot ((\boldsymbol{C}_1 || \boldsymbol{C}_2 - M || \boldsymbol{C}_3) - (\boldsymbol{A} || \boldsymbol{h} || \boldsymbol{c} + \boldsymbol{d} \odot \mathcal{H}(\ell')) \cdot \boldsymbol{R})]_T$$

While the prover computes $\mathsf{ProjHash}(\mathsf{hp}, \boldsymbol{\Pi}) = [\boldsymbol{\Pi} \cdot \mathsf{hp}]_T$.

- *Correctness*: comes directly from the previous equations.
- *Smoothness*: on a binding CRS, $[\boldsymbol{U}]_2$'s last column is in the span of the k first (which are simply $[\boldsymbol{B}]_2$), hence as $\mathsf{hk} \in \mathbb{Z}_p^{k+1}$, the k equations given in hp are not enough to determine its value and so it is still perfectly hidden from an information theoretic point of view.
- *Pseudo-Randomness*: Under the MDDH assumption, the subset membership decision is a hard problem, as the generalized Cramer-Shoup is IND-CCA-2, and $[\boldsymbol{R}]_2$ is an IND-CPA commitment to $\boldsymbol{r}$.

Theorem 10. *Under the k-MDDH assumption, the above SP-SPHF is strongly pseudo-random on a perfectly hiding CRS.*

For sake of compactness, the proof is postponed to the paper full version [15].

Efficiency. The rough size of a projection key is $k \times (k+3)$ (number of elements in each proof times number of proofs). It should be noted, that for a CS-SPHF (in the case of the oblivious transfer), instead of repeating the projection key $k+3$ times (in order to verify each component of the Cramer-Shoup), one can generate a value $\varepsilon \xleftarrow{\$} \mathbb{Z}_p$, an hp for a single equation, and say that for the other component, one simply uses $\mathsf{hp}^{\varepsilon^i}$, as the trick explained in [1].

5 Application: Nearly Optimal Size 1-out-of-m Oblivious Transfer

5.1 Main Idea of the Construction

Our oblivious transfer scheme builds upon that presented by Abdalla *et al.* at Asiacrypt 2013 [1]. In their scheme, the authors use a SPHF-friendly commitment (which is a notion stronger than a UC commitment) along with its associated SPHF in a now classical way to implicitly open the commitment. They claim that the commitment presented in [34] cannot be used in such an application, since it is not "robust", which is a security notion meaning that one cannot produce a commitment and a label that extracts to x' (possibly $x' = \perp$) such that there exists a valid opening data to a different input x, even with oracle access to the extraction oracle (ExtCom) and to fake commitments (using SCom). Indeed, because of the perfectly-hiding setting of Groth-Sahai proofs, for any ciphertext C and for any message x, there exists a proof Π that makes the verification of C on x. However, we show in this section that in spite of this result, such a commitment can indeed be used in a relatively close construction of oblivious transfer scheme. To this aim, we use our construction of structure-preserving SPHF on FLM's commitment, simply using the decommitment value (a Groth-Sahai proof) as the witness, presented in Sect. 4.3.

It should be noted that the commitment used in [1,2] has the major drawback of leaking the bit-length of the committed message. While in application to Oblivious Transfer this is not a major problem, for PAKE this is a way more sensitive issue, as we show in the next section. Moreover, using FLM's commitment is conceptually simpler, since the equivocation only needs to modify the witness, allowing the user to compute honestly its message in the commitment phase, whereas in the original commitments, a specific flow had to be sent during the commitment phase (with a different computation and more witnesses for the SPHF, than in the honest computation of the commitment).

5.2 A Universally Composable Oblivious Transfer with Adaptive Security Based on MDDH

We denote by DB the database of the server containing $t = 2^m$ lines, and j the line requested by the user in an oblivious way. We assume the existence of a Pseudo-Random Generator (PRG) F with input size equal to the plaintext size, and output size equal to the size of the messages in the database and a $\mathsf{IND\text{-}CPA}$ encryption scheme $\mathcal{E} = (\mathsf{Setup}_{\mathrm{cpa}}, \mathsf{KeyGen}_{\mathrm{cpa}}, \mathsf{Encrypt}_{\mathrm{cpa}}, \mathsf{Decrypt}_{\mathrm{cpa}})$ with plaintext size at least equal to the security parameter. The commitment used is the variant of [34] described above. It is denoted as Com^{ℓ} in the description of the scheme, with ℓ being a label. Note that sid denotes the session identifier, ssid the subsession identifier and cid the commitment identifier and that the combination $(\mathsf{sid}, \mathsf{cid})$ is globally unique, as in [34,37].

We present our construction, in Fig. 3, following the global framework presented in [1], for an easier efficiency comparison (we achieve nearly optimality in the sense that it is linear in the number of lines of the database, but with a constant equal to 1 only).

Theorem 11. *The oblivious transfer scheme described in Fig. 3 is UC-secure in the presence of adaptive adversaries, assuming reliable erasures and authenticated channels.*

The proof is given in the paper full version [15] for completeness.

CRS generation:
$\mathsf{crs} \xleftarrow{\$} \mathsf{SetupCom}(1^{\mathfrak{K}}), \mathsf{param}_{\mathrm{cpa}} \xleftarrow{\$} \mathsf{Setup}_{\mathrm{cpa}}(1^{\mathfrak{K}})$.

Pre-flow:

1. Server generates a key pair $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathrm{cpa}}(\mathsf{param}_{\mathrm{cpa}})$ for $\mathcal{E}$, stores sk and completely erases the random coins used by KeyGen
2. Server sends pk to User

Index query on j:

1. User chooses a random value J, computes $S \leftarrow F(J)$ and encrypts J under pk: $c \xleftarrow{\$} \mathsf{Encrypt}_{\mathrm{cpa}}(\mathsf{pk}, J)$
2. User computes $([\boldsymbol{C}]_1, [\boldsymbol{R}]_2, [\boldsymbol{\Pi}]_1) \xleftarrow{\$} \mathsf{Com}^{\ell}(\mathsf{crs}, j, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$ with the label $\ell = (\mathsf{sid}, \mathsf{ssid}, P_i, P_j)$
3. User stores $[\boldsymbol{\Pi}]_1$ and completely erases J and the random coins used by Com and $\mathsf{Encrypt}_{\mathrm{cpa}}$ and sends $[\boldsymbol{C}]_1, [\boldsymbol{R}]_2$ and c to Server

Database input $(n_1, \ldots, n_t)$:

1. Server decrypts $J \leftarrow \mathsf{Decrypt}_{\mathrm{cpa}}(\mathsf{sk}, c)$ and computes $S \leftarrow F(J)$
2. For $s = 1, \ldots, t$: Server computes $\mathsf{hk}_s \xleftarrow{\$} \mathsf{HashKG}(\mathcal{L}_s)$, $\mathsf{hp}_s \leftarrow \mathsf{ProjKG}(\mathsf{hk}_s, \mathcal{L}_s)$, $K_s \leftarrow \mathsf{Hash}(\mathsf{hk}_s, (\mathcal{L}_s, (\ell, [\boldsymbol{C}]_1, [\boldsymbol{R}]_2)))$, and $N_s \leftarrow S \oplus K_s \oplus n_s$
3. Server erases everything except $(\mathsf{hp}_s, N_s)_{s=1,\ldots,t}$ and sends them over a secure channel

Data recovery:
Upon receiving $(\mathsf{hp}_s, N_s)_{s=1,\ldots,t}$, User computes
$K_j \leftarrow \mathsf{ProjHash}(\mathsf{hp}_j, (\mathcal{L}_j, \ell, [\boldsymbol{C}]_1, [\boldsymbol{R}]_2), [\boldsymbol{\Pi}]_1)$ and gets $n_j \leftarrow S \oplus K_j \oplus N_j$.

Fig. 3. UC-Secure 1-out-of-t OT from an SPHF-Friendly Commitment (for Adaptive Security)

6 Application: Adaptive and Length-Independent One-Round PAKE

Password-authenticated key exchange (PAKE) protocols allow two players to agree on a shared high entropy secret key, that depends on their own passwords only. Katz and Vaikuntanathan recently came up with the first concrete one-round PAKE protocols [43], where the two players just have to send simultaneous flows to each other. Following their idea, [14] proposed a round-optimal PAKE protocol UC secure against passive corruptions. On the other hand, [2] proposed the first protocol UC secure against adaptive corruptions, and [1] built upon both [43] and [2], to propose the first one-round protocol UC secure against adaptive corruptions. Unfortunately, both of them share a drawback, which is that they use a commitment growing linearly with the length of a password. Besides being an efficiency problem, it is over all a security issue in the UC framework. Indeed, the simulator somehow has to "guess" the length of the password of the player it simulates, otherwise it is unable to equivocate the commitment (since the commitment reveals the length of the password it commits to). Since such a guess is impossible, the apparently only solution to get rid of this limitation seems to give the users an upper-bound on the length of their passwords and to ask them to compute commitments of this length, which leads to costly computations.

In this section, we are now going to present a constant-size, round-optimal, PAKE UC secure against adaptive corruptions. It builds upon the protocol proposed in [1], using the same techniques as in the former section to avoid the apparent impossibility to use FLM's commitment.

It should be noted that we need the classical requirement for extraction capabilities (see for example [16,47] for a detailed explanation), *i.e.* a password pw is assumed to be a bit-string of length bounded by $\log p - 2$, and then one can use a bijective embedding function G mapping $\{0,1\}^{|p|-2}$ in $\mathbb{G}_1$. For the sake of simplicity, we continue to write pw_i in the high level description, but it should be interpreted as a commitment to $G(\mathsf{pw}_i)$.

The language $\mathcal{L}_{\mathsf{pw}_i}$ is then the language of valid Cramer-Shoup encryptions of the embedded password $G(\mathsf{pw}_i)$, consistent with the randomness committed in the second part, and the rest of the label.

Theorem 12. *The Password Authenticated Key Exchange scheme described in Fig. 4 is UC-secure in the presence of adaptive adversaries, assuming reliable erasures and authenticated channels.*

The proof is given in the paper full version [15] for completeness.

CRS: $\mathsf{crs} \xleftarrow{\$} \mathsf{SetupCom}(1^{\mathfrak{K}})$.
Protocol execution by P_i with pw_i:

1. P_i generates $\mathsf{hk}_i \xleftarrow{\$} \mathsf{HashKG}(\mathcal{L}_{\mathsf{pw}_i})$, $\mathsf{hp}_i \leftarrow \mathsf{ProjKG}(\mathsf{hk}_i, \mathcal{L}_{\mathsf{pw}_i})$ and erases any random coins used for the generation
2. P_i computes $([\boldsymbol{C}_i]_1, [\boldsymbol{R}_i]_2, [\boldsymbol{\Pi}_i]_1) = \mathsf{Com}^{\ell_i}(\mathsf{crs}, \mathsf{pw}_i, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$ with $\ell_i = (\mathsf{sid}, P_i, P_j, \mathsf{hp}_i)$
3. P_i stores $[\boldsymbol{\Pi}_i]_1$, completely erases random coins used by Com and sends $\mathsf{hp}_i, [\boldsymbol{C}_i]_1, [\boldsymbol{R}_i]_2$ to P_j

Key computation: Upon receiving $\mathsf{hp}_j, [\boldsymbol{C}_j]_1, [\boldsymbol{R}_j]_2$ from P_j

1. P_i computes $H'_i \leftarrow \mathsf{ProjHash}(\mathsf{hp}_j, (\mathcal{L}_{\mathsf{pw}_i}, \ell_i, [\boldsymbol{C}_i]_1, [\boldsymbol{R}_i]_2), [\Pi_i]_1))$ and $H_j \leftarrow \mathsf{Hash}(\mathsf{hk}_i, (\mathcal{L}_{\mathsf{pw}_i}, \ell_j, [\boldsymbol{C}_j]_1, [\boldsymbol{R}_j]_2))$ with $\ell_j = (\mathsf{sid}, P_j, P_i, \mathsf{hp}_j)$
2. P_i computes $\mathsf{sk}_i = H'_i \cdot H_j$ and erases everything else, except pw_i.

Fig. 4. UC-Secure PAKE from the revisited FLM Commitment

7 Application: Anonymous Credential-Based Message Transmission

Anonymous Credential protocols [21,27,31] allow to combine security and privacy. Typical credential use involves three main parties. Users need to interact with some authorities to obtain their credentials (assumed to be a set of attributes validated / signed), and then prove to a server that a subpart of their attributes verifies an expect policy.

In this section, we give another go to Anonymous Credential, this time to allow message recovery. This is between Anonymous Credential but also Conditional Oblivious Transfer [51] and Oblivious Signature-Based Envelope [46].

We present a constant-size, round-optimal protocol that allow to use a Credential to retrieve a message without revealing the Anonymous Credentials in a UC secure way, by simply building on the commitment proposed earlier in the paper.

7.1 Anonymous Credential System

In a Attribute-Based Credential system, we assume that different organization issue credentials to users. A user i possesses a set of credential Cred_i of the form $\{\mathsf{Cred}_{i,j}, \mathsf{vk}_j\}$ where organization j assesses that the user verifies some property. (The DMV will assess that the user is indeed capable of driving, the university that she has a bachelor in Computer Science, while Squirrel Airways that she reached the gold membership, all those authorities don't communicate with each other).

A Server might have an access Policy P requiring some elements (For example being a female, with a bachelor, and capable of driving).

- $\mathsf{Setup}(1^{\mathfrak{K}})$: A probabilistic algorithm that gets a security parameter $\mathfrak{K}$, an upper bound t for the size of attribute sets and returns the public parameters param
- $\mathsf{OKeyGen}(\mathsf{param})$: Generates a pair of signing keys $\mathsf{sk}_j, \mathsf{vk}_j$ for each organization.
- $\mathsf{UKeyGen}(\mathsf{param})$: Generates a pair of keys $\mathsf{sk}_i, \mathsf{vk}_i$ for each use.
- $\mathsf{CredObtain}(\langle U_i, \mathsf{sk}_i\rangle, \langle O_j, \mathsf{sk}_j\rangle)$ Interactive process that allows a user i to obtain some credentials from organization j by providing his public key vk_j and a proof that it belongs to him.
- $\mathsf{CredUse}(\langle U_i, \mathsf{Cred}_i, \mathsf{sk}_i\rangle, \langle S, P, M\rangle)$ Interactive process that allows a user i to access a message guarded by the server S under some policy P by using the already obtained credentials.

An attribute-based anonymous credential system is called secure if it is correct, unforgeable and anonymous.

CRS generation:
$\mathsf{crs} \overset{\$}{\leftarrow} \mathsf{SetupCom}(1^{\mathfrak{K}}), \mathsf{param}_{\mathrm{cpa}} \overset{\$}{\leftarrow} \mathsf{Setup}_{\mathrm{cpa}}(1^{\mathfrak{K}})$.

Pre-flow:

1. Server generates a key pair $(\mathsf{pk}, \mathsf{sk}) \overset{\$}{\leftarrow} \mathsf{KeyGen}_{\mathrm{cpa}}(\mathsf{param}_{\mathrm{cpa}})$ for $\mathcal{E}$, stores sk and completely erases the random coins used by KeyGen
2. Server sends pk to User

Credential Use by user i:

1. User chooses a random value J, computes $S \leftarrow F(J)$ and encrypts J under pk: $c \overset{\$}{\leftarrow} \mathsf{Encrypt}_{\mathrm{cpa}}(\mathsf{pk}, J)$
2. User computes $([\boldsymbol{C}]_1, [\boldsymbol{R}]_2, [\boldsymbol{\Pi}]_1) \overset{\$}{\leftarrow} \mathsf{Com}^{\ell}(\mathsf{crs}, \mathsf{Cred}_i, \mathsf{sid}, \mathsf{cid}, P_i, P_j)$ with $\ell = (\mathsf{sid}, \mathsf{ssid}, P_i, P_j)$
3. User stores $[\boldsymbol{\Pi}]_1$ and completely erases J and the random coins used by Com and $\mathsf{Encrypt}_{\mathrm{cpa}}$ and sends $[\boldsymbol{C}]_1, [\boldsymbol{R}]_2$ and c to Server

Database input M with policy P:

1. Server decrypts $J \leftarrow \mathsf{Decrypt}_{\mathrm{cpa}}(\mathsf{sk}, c)$ and computes $S \leftarrow F(J)$
2. Server computes $\mathsf{hk}_P \overset{\$}{\leftarrow} \mathsf{HashKG}(\mathcal{L}_P)$, $\mathsf{hp}_P \leftarrow \mathsf{ProjKG}(\mathsf{hk}_P, \mathcal{L}_P)$, $K_P \leftarrow \mathsf{Hash}(\mathsf{hk}_P, (\mathcal{L}_P, (\ell, [\boldsymbol{C}]_1, [\boldsymbol{R}]_2)))$, and $N_P \leftarrow S \oplus K_P \oplus M$
3. Server erases everything except (hp_P, N_P) and sends them over a secure channel

Data recovery:
Upon receiving (hp_P, N_P), User computes
$K \leftarrow \mathsf{ProjHash}(\mathsf{hp}_P, (\mathcal{L}_P, \ell, [\boldsymbol{C}]_1, [\boldsymbol{R}]_2), [\boldsymbol{\Pi}]_1)$ and gets $M \leftarrow S \oplus K \oplus N_P$.

Fig. 5. UC-Secure Anonymous Credential from an SPSPHF-Friendly Commitment (for Adaptive Security).

7.2 Construction

Smooth Projective Hash Functions have been shown to handle complex languages [2,13], those properties can naturally be extended to Structure Preserving Smooth Projective Hash Function, allowing credentials to be expressive as disjunction / conjunction of sets of credentials, range proofs, or even composition (having a credential from authority A signed by authority B for example).

What is really new with the Structure Preserving part is that now a user can request to have a credential on a witness by requiring a Structure-Preserving signature on it, while before scalars either required to give too much information to the server B or prevented chaining as most signatures requires some sort of Hashing (BLS requires an explicit Hash, while signature *à la* Waters requires to handle a bit per bit version of the message hindering drastically the efficiency of the protocol). This allows more possibilities in both the Credential Generation step and the policy required for accessing messages, while maintaining an efficient construction.

Theorem 13. *The Anonymous Credential Protocol described in Fig. 5 is* UC*-secure in the presence of adaptive adversaries, assuming reliable erasures and authenticated channels.*

The ideal functionality and a sketch of the proof are given in the paper full version [15] for completeness.

A Commitments and Smooth Projective Hash Functions

A.1 Encryption

An encryption scheme $\mathcal{C}$ is described through four algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$:

- $\mathsf{Setup}(1^{\mathfrak{K}})$, where $\mathfrak{K}$ is the security parameter, generates the global parameters param of the scheme;
- $\mathsf{KeyGen}(\mathsf{param})$ outputs a pair of keys, a (public) encryption key pk and a (private) decryption key dk;
- $\mathsf{Encrypt}(\mathsf{ek}, M; \rho)$ outputs a ciphertext $\mathcal{C}$, on M, under the encryption key pk, with the randomness ρ;
- $\mathsf{Decrypt}(\mathsf{dk}, \mathcal{C})$ outputs the plaintext M, encrypted in the ciphertext $\mathcal{C}$ or $\perp$.

Such encryption scheme is required to have the following security properties:

- *Correctness*: For every pair of keys $(\mathsf{ek}, \mathsf{dk})$ generated by KeyGen, every messages M, and every random ρ, we should have
$$\mathsf{Decrypt}(\mathsf{dk}, \mathsf{Encrypt}(\mathsf{ek}, M; \rho)) = M.$$
- *Indistinguishability under Adaptive Chosen Ciphertext Attack* $\mathsf{IND\text{-}CCA}$ (see [49,52]): An adversary should not be able to efficiently guess which message has been encrypted even if he chooses the two original plaintexts, and ask several decryption of ciphertexts different from challenge one.
The $\mathsf{ODecrypt}$ oracle outputs the decryption of c under the challenge decryption key dk. The input queries (c) are added to the list $\mathcal{CT}$ of decrypted ciphertexts.

```
$\mathsf{Exp}^{\mathsf{ind\text{-}cca}-b}_{\mathcal{E},\mathcal{A}}(\mathfrak{K})$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^{\mathfrak{K}})$
2. $(\mathsf{pk}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$
3. $(M_0, M_1) \leftarrow \mathcal{A}(\texttt{FIND} : \mathsf{pk}, \mathsf{ODecrypt}(\cdot))$
4. $c^* \leftarrow \mathsf{Encrypt}(\mathsf{ek}, M_b)$
5. $b' \leftarrow \mathcal{A}(\texttt{GUESS} : c^*, \mathsf{ODecrypt}(\cdot))$
6. IF $(c^*) \in \mathcal{CT}$ RETURN 0
7. ELSE RETURN $b'$
```

A.2 Commitments

A commitment scheme is said *equivocable* if it has a second setup $\mathsf{SetupComT}(1^{\mathfrak{K}})$ that additionally outputs a trapdoor τ, and two algorithms

- $\mathsf{SimCom}^{\ell}(\tau)$ takes as input the trapdoor τ and a label ℓ and outputs a pair (C, eqk), where C is a commitment and eqk an equivocation key;
- $\mathsf{OpenCom}^{\ell}(\mathsf{eqk}, C, x)$ takes as input a commitment C, a label ℓ, a message x, an equivocation key eqk, and outputs an opening data δ for C and ℓ on x.

such as the following properties are satisfied: *trapdoor correctness* (all simulated commitments can be opened on any message), *setup indistinguishability* (one cannot distinguish the CRS ρ generated by $\mathsf{SetupCom}$ from the one generated by $\mathsf{SetupComT}$) and *simulation indistinguishability* (one cannot distinguish a real commitment (generated by Com) from a fake commitment (generated by SCom), even with oracle access to fake commitments), denoting by SCom the algorithm that takes as input the trapdoor τ, a label ℓ and a message x and which outputs $(C, \delta) \overset{\$}{\leftarrow} \mathsf{SCom}^{\ell}(\tau, x)$, computed as $(C, \mathsf{eqk}) \overset{\$}{\leftarrow} \mathsf{SimCom}^{\ell}(\tau)$ and $\delta \leftarrow \mathsf{OpenCom}^{\ell}(\mathsf{eqk}, C, x)$.

A commitment scheme $\mathcal{C}$ is said to be *extractable* if it has a second setup $\mathsf{SetupComT}(1^{\mathfrak{K}})$ that additionally outputs a trapdoor τ, and a new algorithm

- $\mathsf{ExtCom}^{\ell}(\tau, C)$ which takes as input the trapdoor τ, a commitment C, and a label ℓ, and outputs the committed message x, or $\perp$ if the commitment is invalid.

such as the following properties are satisfied: *trapdoor correctness* (all commitments honestly generated can be correctly extracted: for all ℓ, x, if $(C, \delta) \overset{\$}{\leftarrow} \mathsf{Com}^{\ell}(x)$ then $\mathsf{ExtCom}^{\ell}(C, \tau) = x$), *setup indistinguishability* (as above) and *binding extractability* (one cannot fool the extractor, *i.e.*, produce a commitment and a valid opening data to an input x while the commitment does not extract to x).

A commitment scheme is said *extractable and equivocable* if the indistinguishable setup algorithm outputs a common trapdoor that allows both equivocability and extractability, and the following properties are satisfied: *strong simulation indistinguishability* (one cannot distinguish a real commitment (generated by Com) from a fake commitment (generated by SCom), even with oracle access to the extraction oracle (ExtCom) and to fake commitments (using SCom)) and *strong binding extractability* (one cannot fool the extractor, *i.e.*, produce a commitment and a valid opening data (not given by SCom) to an input x while the commitment does not extract to x, even with oracle access to the extraction oracle (ExtCom) and to fake commitments (using SCom)).

A.3 Smooth Projective Hash Functions Used with Commitments

The strong pseudo-randomness property, from [14], is defined by the experiment $\mathsf{Exp}_{\mathcal{A}}^{\texttt{c-s-ps-rand}}(\mathfrak{K})$ depicted in Fig. 6. It is a strong version of the pseudo-randomness where the adversary is also given the hash value of a commitment

```
$\mathsf{Exp}_{\mathcal{A}}^{\texttt{c-s-ps-rand-}b}(\mathfrak{K})$
    $(\rho, \tau) \overset{\$}{\leftarrow} \mathsf{SetupComT}(1^{\mathfrak{K}})$
    $(\ell, x, \mathsf{state}) \overset{\$}{\leftarrow} \mathcal{A}^{\mathsf{SCom}^{\cdot}(\tau,\cdot),\mathsf{ExtCom}^{\cdot}(\tau,\cdot)}(\rho)$; $C \overset{\$}{\leftarrow} \mathsf{SimCom}^{\ell}(\tau)$
    $\mathsf{hk} \overset{\$}{\leftarrow} \mathsf{HashKG}(L_x)$; $\mathsf{hp} \leftarrow \mathsf{ProjKG}(\mathsf{hk}, L_x, \perp)$
    If $(b = 0)$ $H \leftarrow \mathsf{Hash}(\mathsf{hk}, L_x, (\ell, C))$
    Else $H \overset{\$}{\leftarrow} \Pi$
    $(\ell', C', \mathsf{state}) \overset{\$}{\leftarrow} \mathcal{A}^{\mathsf{SCom}^{\cdot}(\tau,\cdot),\mathsf{ExtCom}^{\cdot}(\tau,\cdot)}(\mathsf{state}, C, \mathsf{hp}, H)$
    If $((\ell', ?, C') \in \Lambda)$ THEN $H' \leftarrow \perp$
    Else $H' \leftarrow \mathsf{Hash}(\mathsf{hk}, L_x, (\ell', C')$
        Return $\mathcal{A}^{\mathsf{SCom}^{\cdot}(\tau,\cdot),\mathsf{ExtCom}^{\cdot}(\tau,\cdot)}(H')$
```

Fig. 6. Strong Pseudo-Randomness

of its choice (obviously not generated by SCom or SimCom though, hence the test with Λ which also contains (C, ℓ, x)). This property only makes sense when the projection key does not depend on the word C to be hashed. It thus applies to KV-SPHF, and CS-SPHF only.

References

1. Abdalla, M., Benhamouda, F., Blazy, O., Chevalier, C., Pointcheval, D.: SPHF-friendly non-interactive commitments. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 214–234. Springer, Heidelberg (2013). doi:10.1007/978-3-642-42033-7_12
2. 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 (2009). doi:10.1007/978-3-642-03356-8_39
3. Abe, M., Chase, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Constant-size structure-preserving signatures: generic constructions and simple assumptions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 4–24. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34961-4_3
4. Abe, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Tagged one-time signatures: tight security and optimal tag size. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 312–331. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36362-7_20
5. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010). doi:10.1007/978-3-642-14623-7_12
6. Abe, M., Groth, J., Haralambiev, K., Ohkubo, M.: Optimal structure-preserving signatures in asymmetric bilinear groups. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 649–666. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22792-9_37
7. Abe, M., Groth, J., Ohkubo, M., Tibouchi, M.: Structure-preserving signatures from type II pairings. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 390–407. Springer, Heidelberg (2014). doi:10.1007/978-3-662-44371-2_22

8. Abe, M., Groth, J., Ohkubo, M., Tibouchi, M.: Unified, minimal and selectively randomizable structure-preserving signatures. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 688–712. Springer, Heidelberg (2014). doi:10.1007/978-3-642-54242-8_29
9. Ateniese, G., Camenisch, J., Hohenberger, S., de Medeiros, B.: Practical group signatures without random oracles. Cryptology ePrint Archive, Report 2005/385 (2005). http://eprint.iacr.org/2005/385
10. Barak, B., Canetti, R., Lindell, Y., Pass, R., Rabin, T.: Secure computation without authentication. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 361–377. Springer, Heidelberg (2005). doi:10.1007/11535218_22
11. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000). doi:10.1007/3-540-45539-6_11
12. Bellovin, S.M., Merritt, M.: Encrypted key exchange: Password-based protocols secure against dictionary attacks. In: 1992 IEEE Symposium on Security and Privacy, pp. 72–84. IEEE Computer Society Press, May 1992
13. 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 (2013). doi:10.1007/978-3-642-36362-7_18
14. 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. LNCS, vol. 8042, pp. 449–475. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40041-4_25
15. Blazy, O., Chevalier, C.: Structure-preserving smooth projective hashing. Cryptology ePrint Archive, Report 2016/258 (2016). http://eprint.iacr.org/2016/258
16. Blazy, O., Chevalier, C., Pointcheval, D., Vergnaud, D.: Analysis and improvement of Lindell's UC-secure commitment schemes. In: Jacobson, M., Locasto, M., Mohassel, P., Safavi-Naini, R. (eds.) ACNS 2013. LNCS, vol. 7954, pp. 534–551. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38980-1_34
17. Blazy, O., Chevalier, C.: Generic construction of uc-secure oblivious transfer. Cryptology ePrint Archive, Report 2015/560 (2015)
18. Blazy, O., Chevalier, C., Vergnaud, D.: Non-interactive zero-knowledge proofs of non-membership. Cryptology ePrint Archive, Report 2015/072 (2015). http://eprint.iacr.org/
19. Blazy, O., Pointcheval, D., Vergnaud, D.: Round-optimal privacy-preserving protocols with smooth projective hash functions. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 94–111. Springer, Heidelberg (2012). doi:10.1007/978-3-642-28914-9_6
20. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004). doi:10.1007/978-3-540-28628-8_3
21. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001). doi:10.1007/3-540-44987-6_7
22. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd FOCS, pp. 136–145. IEEE Computer Society Press, October 2001

23. Canetti, R., Fischlin, M.: Universally composable commitments. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 19–40. Springer, Heidelberg (2001). doi:10.1007/3-540-44647-8_2
24. Canetti, R., Halevi, S., Katz, J., Lindell, Y., MacKenzie, P.: Universally composable password-based key exchange. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 404–421. Springer, Heidelberg (2005). doi:10.1007/11426639_24
25. Canetti, R., Krawczyk, H.: Universally composable notions of key exchange and secure channels. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 337–351. Springer, Heidelberg (2002). doi:10.1007/3-540-46035-7_22
26. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: 34th ACM STOC, pp. 494–503. ACM Press, May 2002
27. Chaum, D.: Showing credentials without identification. In: Pichler, F. (ed.) EUROCRYPT 1985. LNCS, vol. 219, pp. 241–244. Springer, Heidelberg (1986). doi:10.1007/3-540-39805-8_28
28. Choi, S.G., Katz, J., Wee, H., Zhou, H.-S.: Efficient, adaptively secure, and composable oblivious transfer with a single, global CRS. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 73–88. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36362-7_6
29. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998). doi:10.1007/BFb0055717
30. 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 (2002). doi:10.1007/3-540-46035-7_4
31. Damgård, I.B.: Payment systems and credential mechanisms with provable security against abuse by individuals. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 328–335. Springer, Heidelberg (1990). doi:10.1007/0-387-34799-2_26
32. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985). doi:10.1007/3-540-39568-7_2
33. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.: An algebraic framework for diffie-hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 129–147. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40084-1_8
34. Fischlin, M., Libert, B., Manulis, M.: Non-interactive and re-usable universally composable string commitments with adaptive security. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 468–485. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25385-0_25
35. 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 (2003). doi:10.1007/3-540-39200-9_33
36. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). doi:10.1007/978-3-540-78967-3_24
37. Hofheinz, D., Müller-Quade, J.: Universally composable commitments using random oracles. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 58–76. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24638-1_4

38. Horvitz, O., Katz, J.: Universally-composable two-party computation in two rounds. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 111–129. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74143-5_7
39. Jutla, C., Roy, A.: Smooth nizk arguments with applications to asymmetric uc-pake. Cryptology ePrint Archive, Report 2016/233 (2016). http://eprint.iacr.org/
40. Jutla, C.S., Roy, A.: Dual-system simulation-soundness with applications to uc-pake and more. Cryptology ePrint Archive, Report 2014/805 (2014)
41. Kalai, Y.T.: Smooth projective hashing and two-message oblivious transfer. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 78–95. Springer, Heidelberg (2005). doi:10.1007/11426639_5
42. Katz, J., Ostrovsky, R., Yung, M.: Efficient password-authenticated key exchange using human-memorable passwords. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 475–494. Springer, Heidelberg (2001). doi:10.1007/3-540-44987-6_29
43. 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 (2009). doi:10.1007/978-3-642-10366-7_37
44. 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 (2011). doi:10.1007/978-3-642-19571-6_18
45. Kiltz, E., Pan, J., Wee, H.: Structure-preserving signatures from standard assumptions, revisited. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 275–295. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48000-7_14
46. Li, N., Du, W., Boneh, D.: Oblivious signature-based envelope. In: Borowsky, E., Rajsbaum, S. (eds.) 22nd ACM PODC, pp. 182–189. ACM, Jul 2003
47. Lindell, Y.: Highly-efficient universally-composable commitments based on the DDH assumption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 446–466. Springer, Heidelberg (2011). doi:10.1007/978-3-642-20465-4_25
48. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: Kosaraju, S.R. (ed.) 12th SODA, pp. 448–457. ACM-SIAM, January 2001
49. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC, pp. 427–437. ACM Press, May 1990
50. 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 (2008). doi:10.1007/978-3-540-85174-5_31
51. Rabin, M.O.: How to exchange secrets with oblivious transfer. Technical Report TR81, Harvard University (1981)
52. Rackoff, C., Simon, D.R.: Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992). doi:10.1007/3-540-46766-1_35