Dual-System Simulation-Soundness with Applications to UC-PAKE and More

Charanjit S. Jutla[1] and Arnab Roy[2]

[1] IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
csjutla@us.ibm.com
[2] Fujitsu Laboratories of America, Sunnyvale, CA, USA
aroy@us.fujitsu.com

Abstract. We introduce a novel concept of dual-system simulation-sound non-interactive zero-knowledge (NIZK) proofs. Dual-system NIZK proof system can be seen as a two-tier proof system. As opposed to the usual notion of zero-knowledge proofs, dual-system defines an intermediate partial-simulation world, where the proof simulator may have access to additional auxiliary information about the word, for example a membership bit, and simulation of proofs is only guaranteed if the membership bit is correct. Further, dual-system NIZK proofs allow a quasi-adaptive setting where the CRS can be generated based on language parameters. This allows for the further possibility that the partial-world CRS simulator may have access to additional trapdoors related to the language parameters. We show that for important hard languages like the Diffie-Hellman language, such dual-system proof systems can be given which allow unbounded partial simulation soundness, and which further allow transition between partial simulation world and single-theorem full simulation world even when proofs are sought on non-members. The construction is surprisingly simple, involving only two additional group elements for general linear-subspace languages in asymmetric bilinear pairing groups.

As a direct application we give a short keyed-homomorphic CCA-secure encryption scheme. The ciphertext in this scheme consists of only six group elements (under the SXDH assumption) and the security reduction is tight. An earlier scheme of Libert et al based on their efficient unbounded simulation-sound QA-NIZK proofs only provided a loose security reduction, and further had ciphertexts almost twice as long as ours.

We also show a single-round universally-composable password authenticated key-exchange (UC-PAKE) protocol which is secure under adaptive corruption in the erasure model. The single message flow only requires **four** group elements under the SXDH assumption. This is the *shortest known* UC-PAKE even without considering adaptive corruption. The latest published scheme which considered adaptive corruption, by Abdalla et al [ABB+13], required non-constant (more than 10 times the bit-size of the password) number of group elements.

Keywords: NIZK, bilinear pairings, UC-PAKE, keyed-homomorphic encryption, SXDH.

1 Introduction

Since the introduction of simulation-sound non-interactive zero-knowledge proofs (NIZK) in [Sah99] (based on the concept of non-malleability [DDN91]), simulation-soundness has become an essential cryptographic tool. While the idea of zero-knowledge simulation [GMR89] brought rigor to the concept of semantic security, simulation-soundness of some form is usually implicit in most cryptographic applications. While the original construction of [Sah99] was rather inefficient, the advent of pairing based cryptography, and in particular Groth-Sahai NIZK proofs [GS08], has led to much more efficient simulation-sound NIZK constructions. Pairing-based cryptography has also led to efficient construction of powerful primitives where simulation-soundness is not very explicit.

It has been shown that different forms of simulation-soundness suffice for many applications. Indeed, the original application (CCA2-secure encryption) considered in [Sah99] only required what is known as single-theorem simulation-soundness (also known as one-time simulation-soundness). However, many other cryptographic constructions are known only using unbounded simulation-sound NIZK proofs. In this paper, we introduce the concept of **dual-system simulation-sound** NIZK proofs, which lie somewhere in between one-time and unbounded simulation-sound NIZK proofs. The aim is to show that this weaker concept suffices for constructions where unbounded simulation-soundness was being used till now. We also show that in many applications this new concept of dual-system simulation soundness is implicit, in the sense that although we cannot get a generic construction from a NIZK proof, we can use the underlying ideas of the dual-system simulation-sound NIZK proofs.

Indeed, our novel definition is inspired by the dual-system identity-based encryption (IBE) scheme of Waters [Wat09], where such a concept was implicit, and led to the first IBE scheme which was fully-secure under static and standard assumptions. So without further ado, we jump straight into the main idea of the new concept. In dual-system simulation-sound NIZK proof systems we will consider three worlds: the real-world, the partial-simulation world, and the one-time full-simulation world. The real world consists of a common-reference string (CRS), an efficient prover P, and an efficient verifier V. The concept of completeness and soundness of P and V with respect to a witness-relation R is well-understood. The full-simulation world is also standard, and it includes two simulators: a CRS simulator and a proof simulator. The proof simulator is a zero-knowledge simulator in the sense that it can simulate proofs even without access to the witness. In order to achieve this, the CRS simulator generates the CRS in a potentially different way and produces a trapdoor for the proof simulator. The **partial-simulation** world we consider also has a CRS simulator, and a proof simulator, but this proof simulator is allowed partial access to the witness (or some other auxiliary information) about the member on which the proof is sought.

At this point, we also bring in the possibility of the CRS being generated as a function of the language or witness-relation under consideration. The recent quasi-adaptive NIZK (QA-NIZK) proofs of [JR13] allow this possibility for dis-

tributions of witness-relations. The CRS in the real and the full-simulation world is generated based on a language parameter generated according to some distribution. Now we consider the possibility that in the partial-simulation world, the CRS simulator actually generates the language parameter itself. In other words, the CRS simulator has access to the "witness" of the language parameter. For example, the CRS simulator may know the discrete-logs of the language parameters. This leads to the possibility that in the partial simulation world the proof simulator may have access to additional trapdoors which makes simulation and/or simulation soundness easier to achieve.

In this paper, we will only define and consider dual-system simulation sound QA-NIZK proofs (called DSS-QA-NIZK), where the only auxiliary information that the partial proof simulator gets is a single bit which is called the **membership bit**. The membership bit indicates whether the word on which the proof is sought is in the language or not. We show that we can achieve unbounded partial-simulation soundness for important languages like the Diffie-Hellman language by relatively simple constructions. The constructions also allow one-time full-ZK simulation, and hence form a DSS-QA-NIZK for the Diffie-Hellman language. We actually give a general construction for arbitrary languages which allow smooth and universal$_2$ projective hash proofs [CS02] and have QA-NIZKs for the language augmented with such a hash proof. We show that for linear subspace languages (over bilinear groups), like the Diffie-Hellman and decisional-linear (DLIN) languages, the requirements for the general construction are easy to obtain. Thus, for all such languages, under the standard and static SXDH assumption in bilinear pairing groups, we get a DSS-QA-NIZK proof of only two group elements.

Table 1 summarizes comparison among existing schemes and ours. DSS is weaker than unbounded simulation soundness, and although incomparable with one time simulation soundness, it seems to enjoy better properties. Consistent with this, we observe that the proof sizes also place in the middle of the shortest known OTSS-NIZKs [ABP15,KW15] and the shortest known USS-NIZKs [KW15] for linear subspaces.

Applications. We now give the main idea as to why such a construction is useful. The security of most applications is shown by reduction to a hard language. However, a particular application may have a more complex language for which the NIZK proofs are required, and the security proof may require soundness of the NIZK system while proofs of many elements (real or fake) of such a complex language are being simulated. The idea is that multiple simulations of such elements can be performed in a partial-simulation manner (i.e. it is always possible to supply the correct membership-bit), and full simulation is only required of one member at a time, on which the hardness assumption can then be invoked.

Keyed-Homomorphic CCA-secure Encryption. As a first application we consider the keyed-homomorphic CCA-secure encryption scheme notion of [EHO$^+$13]. In such an encryption scheme, a further functionality called Eval is available which

Table 1. Comparison with existing NIZK schemes for linear subspaces with table adapted from [KW15]. The language of interest is a t dimensional subspace of an n dimensional ambient space. m is the bit-size of the tag. AS is adaptive-soundness. OTSS is one-time simulation-soundness and USS is unbounded simulation-soundness.

	Soundness	Assumption	Proof	CRS	#pairings
[GS08]	AS	DLIN	$2n+3t$	6	$3n(t+3)$
[LPJY14]	AS	DLIN	3	$2n+3t+3$	$2n+4$
[JR13]	AS	k-Linear	$k(n-t)$	$2kt(n-t)+k+1$	$k(n-t)(t+2)$
[JR14a]	AS	k-Linear	k	$kn+kt+k^2$	$kn+k^2$
[ABP15]	AS	k-Linear	k	$kn+kt+k$	$kn+k$
[KW15]	AS	k-Linear	k	$kn+kt+k-1$	$kn+k-1$
[ABP15]	OTSS	k-Linear	k	$2m(kn+(k+1)t)+k$	$mkn+k$
[KW15]	OTSS	k-Linear	k	$2m(kn+(k+1)t)$+k−1	mkn+k−1
This paper	DSS	k-Linear	$k+1$	$k(n+1)+kt+k^2$	$k(n+1)+k^2$
[CCS09]	USS	DLIN	$2n+6t+52$	18	$O(tn)$
[LPJY14]	USS	DLIN	20	$2n+3t+3m+10$	$2n+30$
[KW15]	USS	k-Linear	$2k+2$	$kn+4(k+t+1)k+2k$	$k(n+k+1)+k$

using a key can homomorphically combine valid ciphertexts. The scheme should provide IND-CCA2 security when this Eval key is unavailable to the adversary, and should continue to enjoy IND-CCA1 security when the Eval key is exposed to the adversary. Emura et al. also gave constructions for such a scheme, albeit schemes which are not publicly verifiable, and further satisfying a weaker notion than CCA1-security when Eval key is revealed. Recently, Libert et al gave a publicly-verifiable construction which is more efficient and also CCA1-secure when Eval key is revealed. Their construction is based on a new and improved unbounded simulation-sound QA-NIZK for linear subspace languages. We show in this paper that a DSS-QA-NIZK for the Diffie-Hellman language suffice, and leads to a much improved construction. While the construction in [LPJY14], under the SXDH assumption, requires nine group elements in one group, and two more in the other plus a one-time signature key pair, our construction *only requires six group elements* in any one of the bilinear groups. Further, while the earlier construction was loose (i.e. loses a factor quadratic in number of Eval calls), our reduction is tight.

UC Password-Authenticated Key Exchange (UC-PAKE). The UC-PAKE ideal functionality was introduced in [CHK$^+$05] where they also gave a three-round construction. In [KV11] a single-round construction for UC-PAKE was given using Groth-Sahai NIZK proofs along with unbounded simulation-soundness construction of [CCS09] (also see [JR12]). Later [BBC$^+$13] gave a UC-PAKE construction based on novel trapdoor smooth projective hash functions, but secure only under static corruption; each message consisted of six group elements in one group, and another five elements in the other group (under the SXDH assumption).

In this paper, we construct a a single-round construction based on dual-system simulation-soundness which is UC-secure under adaptive corruption (in the erasure model), and which has only a total of **four** group elements in each

message. The key is generated in the target group. The construction is not a black-box application of the DSS-QA-NIZK for the Diffie-Hellman language, but uses its underlying idea as well as the various component algorithms of the DSS-QA-NIZK. The main idea of the construction is given in more detail in Section 6.2.

To the best of our knowledge, this is the *shortest known UC-PAKE*, even without considering adaptive corruption. The first UC-PAKE to consider adaptive corruption was by Abdalla, Chevalier and Pointcheval [ACP09], which was a two round construction. Recently, Abdalla et al [ABB+13] also constructed a single round protocol, which required a non-constant (more than 10 times the bit-size of the password) number of group elements in each flow. Comparison with existing UC-PAKEs is given in Table 2.

Table 2. Comparison with existing UC-PAKE schemes. m is the password size in bits and λ is the security parameter. AC stands for Adaptive Corruption. For one-round schemes, message size is per flow.

	AC	One-round	Assumption	Message size
[ACP09]	yes	no	DDH	$O(m\lambda)$
[KV11]	no	yes	DLIN	$> 65 \times \mathbb{G}$
[JR12]	no	yes	SXDH	> 30 total group elements
[BBC+13]	no	yes	SXDH	$6 \times \mathbb{G}_1 + 5 \times \mathbb{G}_2$
[ABB+13]	yes	yes	SXDH	$10 * m \times \mathbb{G}_1 + m \times \mathbb{G}_2$
This paper	yes	yes	SXDH	$3 \times \mathbb{G}_1 + 1 \times \mathbb{G}_2$

Identity-Based Encryption (IBE). In the full version of this paper [JR14b], we show that the recent efficient dual-system IBE [JR13] (inspired by the original dual-system IBE of Waters [Wat09]) can also be obtained using the ideas of DSS-QA-NIZK. While the construction is not black-box and utilizes additional "smoothness" and "single-pairing-product test" properties of the verifier, it along with the other two applications clearly demonstrate the power and utility of the new notion, which we expect will find many more applications.

2 Preliminaries: Quasi-Adaptive NIZK Proofs

A witness relation is a binary relation on pairs of inputs, the first called a word and the second called a witness. Note that each witness relation R defines a corresponding language L which is the set of all x for which there exists a witness w, such that $R(x, w)$ holds.

We will consider Quasi-Adaptive NIZK proofs [JR13] for a probability distribution $\mathcal{D}$ on a collection of (witness-) relations $\mathcal{R} = \{R_\rho\}$ (with corresponding languages L_ρ). Recall that in a quasi-adaptive NIZK, the CRS can be set after the language parameter has been chosen according to $\mathcal{D}$. Please refer to [JR13] for detailed definitions.

Definition 1. *([JR13]) We call* $(\mathsf{pargen}, \mathsf{crsgen}, \mathsf{prover}, \mathsf{ver})$ *a* (labeled) quasi-adaptive non-interactive zero-knowledge *(QA-NIZK) proof system for witness-relations* $\mathcal{R}_\lambda = \{R_\rho\}$ *with parameters sampled from a distribution* $\mathcal{D}$ *over associated parameter language* Lpar*, if there exist simulators* $\mathsf{crs\text{-}sim}, \mathsf{sim}$ *such that for all non-uniform PPT adversaries* $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ *we have (in all of the following probabilistic experiments, the experiment starts by setting* λ *as* $\lambda \leftarrow \mathsf{pargen}(1^m)$*, and choosing* ρ *as* $\rho \leftarrow \mathcal{D}_\lambda$*):*

Quasi-Adaptive Completeness:

$$\Pr\left[\begin{array}{c} \text{CRS} \leftarrow \mathsf{crsgen}(\lambda, \rho); (x, w, l) \leftarrow \mathcal{A}_1(\text{CRS}, \rho); \\ \pi \leftarrow \mathsf{prover}(\text{CRS}, x, w, l): \quad \mathsf{ver}(\text{CRS}, x, l, \pi) = 1 \text{ if } R_\rho(x, w) \end{array}\right] = 1$$

Quasi-Adaptive Soundness:

$$\Pr[\text{CRS} \leftarrow \mathsf{crsgen}(\lambda, \rho); (x, l, \pi) \leftarrow \mathcal{A}_2(\text{CRS}, \rho) : x \notin L_\rho \wedge \mathsf{ver}(\text{CRS}, x, l, \pi) = 1] \approx 0$$

Quasi-Adaptive Zero-Knowledge:

$$\Pr[\text{CRS} \leftarrow \mathsf{crsgen}(\lambda, \rho) : \mathcal{A}_3^{\mathsf{prover}(\text{CRS}, \cdot, \cdot, \cdot)}(\text{CRS}, \rho) = 1] \approx$$
$$\Pr[(\text{CRS}, \mathsf{trap}) \leftarrow \mathsf{crs\text{-}sim}(\lambda, \rho) : \mathcal{A}_3^{\mathsf{sim}^*(\text{CRS}, \mathsf{trap}, \cdot, \cdot, \cdot)}(\text{CRS}, \rho) = 1],$$

where $\mathsf{sim}^*(\text{CRS}, \mathsf{trap}, x, w, l) = \mathsf{sim}(\text{CRS}, \mathsf{trap}, x, l)$ *for* $(x, w) \in R_\rho$ *and both oracles (i.e.* prover *and* sim^**) output failure if* $(x, w) \notin R_\rho$*.*

The QA-NIZK is called a **statistical zero-knowledge** QA-NIZK if the view of adversary $\mathcal{A}_3$ above in the two experiments is statistically indistinguishable.

3 Dual-System Simulation-Soundness

To define dual-system simulation soundness of QA-NIZK proofs, we will consider three worlds: the real-world, the partial-simulation world, and the one-time (or single theorem) full-simulation world. While the real-world and the full-simulation world should be familiar from earlier definitions of NIZK proof systems, the partial-simulation world leads to interesting possibilities. To start with, in the partial simulation world, one would like the proof simulator to have access to partial or complete witness of the word[3]. Finally, in the quasi-adaptive setting, the language parameters may actually be generated by the CRS simulator and hence the simulator may have access to, say, the discrete logs of the language parameters, which can serve as further trapdoors.

Rather than considering these general settings, we focus on a simple partial-simulation setting, where (a) the CRS simulator can generate the language parameters itself and (b) the proof simulator when invoked with a word x is given an additional bit β, which we call the **membership bit**, that represents the information whether x is indeed a member or not.

[3] In case the proof simulator is being invoked on a non-language word, it is not immediately clear what this witness can be, unless we also define a language and a distribution for a super-language which includes the language under consideration as a subset.

The partial simulation world is required to be unbounded simulation-sound, and hopefully this should be easier to prove than usual unbounded simulation-soundness (given that its simulators have additional information). We also allow the partial simulation world to be sound with respect to a private verifier (this concept has been considered earlier in [JR12]), and this further leads to the possibility of easier and/or simpler constructions. A surprising property achievable under such a definition is that one can go back and forth between the partial-simulation world and the one-time full-simulation world even when simulating fake tuples.

Definition 2 (Dual-System Non-Interactive Proofs). *A Dual-system non-interactive proof system consists of PPT algorithms defined in three worlds as follows:*

Real World *consisting of:*

- *A pair of* **CRS generators** $(\mathsf{K}_0, \mathsf{K}_1)$*, where* K_0 *takes a unary string and produces an ensemble parameter* λ*. (The ensemble parameter* λ *is used to sample a witness-relation parameter* ρ *using* $\mathcal{D}_\lambda$ *in the security definition.) PPT algorithm* K_1 *uses* ρ *(and* λ*) to produce the real-world CRS* ψ*.*
- *A* **prover** P *that takes as input a CRS, a language member and its witness, a label, and produces a proof.*
- *A* **verifier** V *that takes as input a CRS, a word, a label, and a proof, and outputs a single bit.*

Partial-Simulation World *consisting of:*

- *A* **semi-functional CRS simulator** sfK_1 *that takes ensemble parameter* λ *as input and produces a witness relation parameter* ρ*, a semi-functional CRS* σ*, as well as two trapdoors* τ *and* η*. The first trapdoor is used by the proof simulator, and the second by the private verifier.*
- *A* **semi-functional simulator** sfSim *that takes a CRS, a trapdoor* τ*, a word, a membership-bit* β*, and a label, to produce a proof.*
- *A* **private verifier** pV *that takes a CRS, a trapdoor* η*, a word, a label, and a proof and outputs a single bit.*

One-time Full Simulation World *consisting of:*

- *A* **one-time full-simulation CRS generator** otfK_1*, that takes as input the ensemble parameter* λ*, the witness relation parameter* ρ *to produce a CRS and three trapdoors* τ*,* τ_1 *and* η*.*
- *A* **one-time full simulator** otfSim *that takes as input a CRS, a trapdoor* τ_1*, a word, a label, and produces a proof*[4]*.*

[4] We remark here that the One-time Full Simulation World also uses a semi-functional simulator as can be seen in Figure 1. It has the same black-box properties as in the Partial-Simulation World, but could potentially have a different internal construction. In this paper it turns out that the same construction suffices for both the worlds, so for the sake of simplicity we forgo making this explicit in the definition.

– *A* **semi-functional verifier** sfV *that takes as input a CRS, a trapdoor* η*, a word, a label, a proof and outputs a bit. The adversaries also have access to the semi-functional simulator.*

Definition 3 (DSS-QA-NIZK). *The definition of the real-world components of a dual-system non-interactive proof to be complete and (computationally) sound are same as in QA-NIZK definition 1. Such a proof system is called a* **dual-system simulation-sound quasi-adaptive NIZK (DSS-QA-NIZK)** *for a collection of witness relations* $\mathcal{R}_\lambda = \{R_\rho\}$*, with parameters sampled from a distribution* $\mathcal{D}$*, if its real-world components are complete and (computationally) sound, and if for all non-uniform PPT adversaries* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4)$ *all of the following properties are satisfied (in all of the following probabilistic experiments, the experiment starts by setting* λ *as* $\lambda \leftarrow \mathsf{K}_0(1^m)$*):*

- **(Composable) Partial-ZK:**

$$\Pr[\rho \leftarrow \mathcal{D}_\lambda; \sigma \leftarrow \mathsf{K}_1(\lambda, \rho) : \mathcal{A}_0(\sigma, \rho) = 1] \approx$$
$$\Pr[(\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda) : \mathcal{A}_0(\sigma, \rho) = 1],$$

and

$$\Pr[(\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda) : \mathcal{A}_1^{\mathsf{P}(\sigma,\cdot,\cdot,\cdot), \mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{V}(\sigma,\cdot,\cdot,\cdot)}(\sigma, \rho) = 1] \approx$$
$$\Pr[(\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda) : \mathcal{A}_1^{\mathsf{sfSim}^*(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{pV}(\sigma,\eta,\cdot,\cdot,\cdot)}(\sigma, \rho) = 1],$$

where $\mathsf{sfSim}^*(\sigma, \tau, x, w, l)$ *is defined to be* $\mathsf{sfSim}(\sigma, \tau, x, \beta = 1, l)$ *(i.e. witness is dropped, and membership-bit* $\beta = 1$*),* **and** *the experiment aborts if either a call to the first oracle (i.e.* P *and* sfSim^**) is with* (x, w, l) *s.t.* $\neg R_\rho(x, w)$*, or call to the second oracle is with an* (x, β, l) *s.t.* $x \notin L_\rho$ *or* $\beta = 0$*.*

- **Unbounded Partial-Simulation Soundness:**

$$\Pr\left[\begin{array}{c}(\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda);\ (x, l, \pi) \leftarrow \mathcal{A}_2^{\mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{pV}(\sigma,\eta,\cdot,\cdot,\cdot)}(\sigma, \rho) : \\ ((x \notin L_\rho) \vee \mathsf{V}(\sigma, x, l, \pi) = 0) \wedge \mathsf{pV}(\sigma, \eta, x, l, \pi) = 1\end{array}\right] \approx 0.$$

- **One-time Full-ZK:**

$$\Pr\left[\begin{array}{c}(\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda); (x^*, l^*, \beta^*, s) \leftarrow \mathcal{A}_3^{\mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{pV}(\sigma,\eta,\cdot,\cdot,\cdot)}(\sigma, \rho); \\ \pi^* \leftarrow \mathsf{sfSim}(\sigma, \tau, x^*, \beta^*, l^*) : \mathcal{A}_4^{\mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{pV}(\sigma,\eta,\cdot,\cdot,\cdot)}(\pi^*, s) = 1\end{array}\right]$$
$$\approx \Pr\left[\begin{array}{c}\rho \leftarrow \mathcal{D}_\lambda;\ (\sigma, \tau, \tau_1, \eta) \leftarrow \mathsf{otfK}_1(\lambda, \rho); \\ (x^*, l^*, \beta^*, s) \leftarrow \mathcal{A}_3^{\mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{sfV}(\sigma,\eta,\cdot,\cdot,\cdot)}(\sigma, \rho); \\ \pi^* \leftarrow \mathsf{otfSim}(\sigma, \tau_1, x^*, l^*) : \mathcal{A}_4^{\mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot), \mathsf{sfV}(\sigma,\eta,\cdot,\cdot,\cdot)}(\pi^*, s) = 1\end{array}\right],$$

where the experiment aborts if either in the call to the first oracle, or in the (x^*, β^*) *produced by* $\mathcal{A}_3$*, the membership-bit provided is not correct for* L_ρ*,* **or** *if* $\langle x^*, l^*, \pi^* \rangle$ *is queried to* sfV/pV*. Here* s *is a state variable.*

The three worlds and the properties of a DSS-QA-NIZK are depicted in Figure 1.

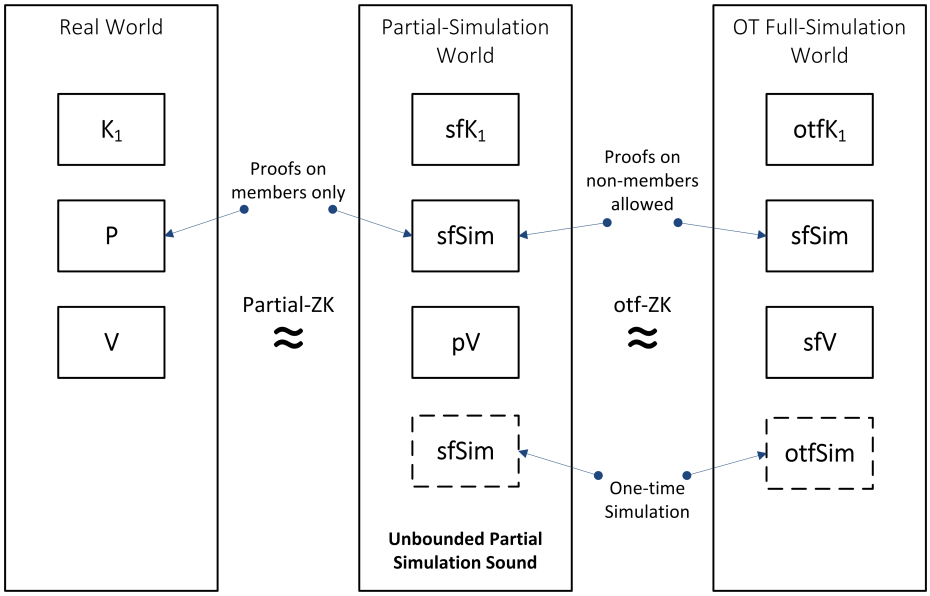

Fig. 1. The three worlds of a DSS-QA-NIZK

Remark 1. In the partial-simulation soundness definition, there is *no restriction* of x, l, π being *not* the same as that obtained from a call to the first oracle sfSim.

Remark 2. Note that in the partial-ZK definition, the calls to the prover are restricted to ones satisfying the relation. However, the calls to the simulator sfSim in the one-time full-ZK definition are only restricted to having the correct membership bit β.

Remark 3. It can be shown that sfSim generated proofs on words (whether members or not) are accepted by real-world verifier V (with semi-functional CRS). Of course, the private verifier pV will *even* reject proofs generated by sfSim on non-language words. This justifies the name "semi-functional simulator". See [JR14b] for a precise claim and proof.

It can also be shown that the semi-functional verifier sfV is still complete, i.e. it accepts language members and proofs generated on them by $\mathsf{P}(\sigma, \cdot, \cdot, \cdot)$ (with σ generated by otfK_1). As opposed to P and pV, it may no longer be sound. This justifies the name "semi-functional verifier" a la Waters' dual-system IBE construction. However, if the one-time full-ZK property holds statistically, it can be shown that the semi-functional verifier is sound in the one-time full-simulation world. See [JR14b] for a precise statement.

Remark 4. The composable partial-ZK and unbounded partial-simulation soundness imply that that the system is true-simulation-sound (cf. true-simulation extractable [Har11]) w.r.t. the semi-functional simulator, as stated below.

Lemma 1. (true-simulation-soundness) *For a DSS-QA-NIZK, for all PPT* $\mathcal{A}$,
$\Pr\left[\begin{array}{c}(\rho,\sigma,\tau,\eta)\leftarrow \mathsf{sfK}_1(\lambda);\ (x,l,\pi)\leftarrow \mathcal{A}^{\mathsf{sfSim}(\sigma,\tau,\cdot,\cdot,\cdot)}(\sigma,\rho):\\ (x\notin L_\rho)\ \wedge\ \mathsf{V}(\sigma,x,l,\pi)=1\end{array}\right]\approx 0$, *where the experiment aborts if* $\mathcal{A}$ *calls the oracle with some* (y,β,l), *s.t.* $y\notin L_\rho$ *or* $\beta=0$.

4 DSS-QA-NIZK for Linear Subspaces

In this section we show that languages that are linear subspaces of vector spaces of hard bilinear groups have very short dual-system simulation sound QA-NIZK. In fact, under the Symmetric-eXternal Diffie-Hellman (SXDH) assumption, such proofs only require two group elements, regardless of the subspace. It was shown in [JR14a] that such subspaces have a QA-NIZK proof of just one group element (under the SXDH assumption). Our construction essentially shows that with one additional group element, one can make the QA-NIZK dual-system simulation-sound. We will actually show a more general construction which is more widely applicable, and does not even refer to bilinear groups or linear subspaces. Informally speaking, the requirement for such a general construction for parameterized languages is that each language has a 2-universal projective hash proof system and the augmented language with this hash proof attached has a QA-NIZK proof system with statistical zero-knowledge. A few other properties of the QA-NIZK are required for this construction, and we show that such properties already hold for the construction of [JR14a]. Since for linear subspaces, 2-universal projective hash proofs are rather easy to obtain, the general construction along with the QA-NIZK of [JR14a] allows us to obtain a short DSS-QA-NIZK for linear subspaces. Apart from abstracting the main ideas involved in the DSS-QA-NIZK construction for linear subspaces, the general construction's wider applicability also allows us to extend our results to linear subspaces with tags.

We start this section by briefly reviewing projective hash proofs [CS02], and their extensions to distributions of languages, as they are extensively used in the rest of the section.

Projective Hash Proof System. For a language L, let X be a superset of L and let $H=(H_k)_{k\in K}$ be a collection of (hash) functions indexed by K with domain X and range another set Π. The hash function family is generalized to a notion of *projective hash function family* if there is a set S of projection keys, and a projection map $\alpha:\ K\to S$, and further the action of H_k on subset L of X is completely determined by the projection key $\alpha(k)$. Finally, the projective hash function family is defined to be ϵ-**universal$_2$** is for all $s\in S$, $x,x^*\in X$, and $\pi,\pi^*\in\Pi$ with $x\notin L\cup\{x^*\}$, the following holds:

$$\Pr[H_k(x)=\pi\mid H_k(x^*)=\pi^*\ \wedge\alpha(k)=s]\ \leq\ \epsilon.$$

A projective hash function family is called ϵ-**smooth** if for all $x\in X\setminus L$, the statistical difference between the following two distributions is ϵ: sample k uniformly from K and π' uniformly from Π; the first distribution is given by the

pair $(\alpha(k), H_k(x))$ and the second by the pair $(\alpha(k), \pi')$. For languages defined by a witness-relation R, the projective hash proof family constitutes a *projective hash proof system* (PHPS) if α, H_k, and another *public evaluation function* $\hat{H}$ that computes H_k on $x \in L$, given a witness of x and *only* the projection key $\alpha(k)$, are all efficiently computable. An efficient algorithm for sampling the key $k \in K$ is also assumed.

The above notions can also incorporate labels. In an *extended PHPS*, the hash functions take an additional input called *label*. The public evaluation algorithm also takes this additional input called label. All the above notions are now required to hold for each possible value of label. The extended PHPS is now defined to be ϵ-**universal$_2$** is for all $s \in S$, $x, x^* \in X$, all labels l and l^*, and $\pi, \pi^* \in \Pi$ with $x \notin L$ and $(x, l) \neq (x^*, l^*)$, the following holds: $\Pr[H_k(x, l) = \pi \mid H_k(x^*, l^*) = \pi^* \wedge \alpha(k) = s] \leq \epsilon$.

Since, we are interested in distributions of languages, we extend the above definition to distribution of languages. So consider a parametrized class of languages $\{L_\rho\}_{\rho \in \mathsf{Lpar}}$ with the parameters coming from an associated parameter language Lpar. Assume that all the languages in this collection are subsets of X. Let H as above be a collection of hash functions from X to Π. We say that the hash family is a projective hash family if for all L_ρ, the action of H_k on L_ρ is determined by $\alpha(k)$. Similarly, the hash family is ϵ-universal$_2$ (ϵ-smooth) for $\{L_\rho\}_{\rho \in \mathsf{Lpar}}$ if for all languages L_ρ the ϵ-universal$_2$ (resp. ϵ-smooth) property holds.

Intuition for the Construction. The main idea of the construction is to first attach (as a proof component) a universal$_2$ and smooth projective hash proof T. The DSS-QA-NIZK is then just (T, π), where π is a QA-NIZK proof of the original language augmented with hash proof T. So, why should this work? First note that the smooth projective hash function is a designated-verifier NIZK, and hence this component T is used in private verification. Secondly, since it is universal$_2$, its soundness will hold even when the Adversary gets to see the projection key $\alpha(k)$ plus one possibly fake hash proof (i.e. $H_k(x)$, where x not in the language).

We will assume in our general construction that the parameterized language is such that the simulator can sample the language parameters along with auxiliary information that allows it to easily verify a language member. For example, this auxiliary information can be discrete logs of the language parameters. The idea of obtaining partial-ZK and unbounded partial-simulation soundness is then pretty simple. The proof simulation of T is easy to accomplish given the hash keys and, crucially, the correct membership-bit. In fact, if the membership-bit is false, T can just be set randomly (by smoothness). The simulation of π part of the proof is done using the QA-NIZK simulation trapdoor. The private verification is done as conjunction of three separate checks: (a) using the auxiliary information, (b) using the hash proof and (c) using the real-world verifier.

Now, in the one-time full simulation, the auxiliary information is not available, but the semi-functional verifier can still use hash keys. Further, we can have one bad use of keys (in full simulation of one proof. Since the oracle calls to

semi-functional simulator sfSim are restricted to having correct membership-bit, they do not yield any additional information about the hash keys.

Requirements of the General Construction. Consider a parameterized class of languages $\{L_\rho\}_{\rho\in\mathsf{Lpar}}$, and a probability distribution $\mathcal{D}$ on Lpar. Assume that this class has a projective hash proof system as above. Let R_ρ be the corresponding witness relation of L_ρ. Now consider the augmented witness-relation $R^*_{\rho,s}$ defined as follows (for $\rho \in \mathsf{Lpar}$ and $s \in S$):

$$R^*_{\rho,s}(\langle x, T, l\rangle, w) \equiv (R_\rho(x, w) \wedge T \stackrel{?}{=} \hat{H}(s, \langle x, l\rangle, w)).$$

Note, the witness remains the same for the augmented relation. Since H is a projective hash function, it follows that for $s = \alpha(k)$, the corresponding augmented language is $L^*_{\rho,s} = \{(x, T, l) \mid x \in L_\rho \wedge T \stackrel{?}{=} H_k(x, l)\}$. Let the distribution $\mathcal{D}'$ on pairs (ρ, s) be defined by sampling ρ according to $\mathcal{D}$ and sampling k uniformly from K, and setting $s = \alpha(k)$. We remark that the language parameters of the augmented language include projection keys s (instead of keys k) because it is crucial that the CRS simulator in the quasi-adaptive NIZK gets only the projection key s (and not k).

We will also assume that the distribution $\mathcal{D}$ on Lpar is efficiently *witness samplable* which is defined by requiring that there are two efficient (probabilistic) algorithms E_1, E_2 such that E_1 can sample ρ from $\mathcal{D}$ along with auxiliary information ψ (which can be thought of as witness of ρ in the language Lpar), and E_2 can decide w.h.p. if a word x is in L_ρ given ρ and ψ, where the probability is defined over choice of ρ according to $\mathcal{D}$ and the internal coins of E_2.

Finally, we need a few additional properties of QA-NIZK proofs (Section 2) that we now define. We will later show that the single group element QA-NIZK construction for linear-subspaces of [JR14a] already satisfies these properties.

Definition 4. *There are various specializations of QA-NIZK of interest:*

- *The QA-NIZK (Section 2) is said to have* **composable zero-knowledge** *[GS08] if the CRS are indistinguishable in the real and simulation worlds, and the simulation is indistinguishable even if the adversary is given the trapdoor. More precisely, for all PPT adversary $\mathcal{A}_1, \mathcal{A}_2$,*
  $\Pr[\text{CRS} \leftarrow \mathsf{crsgen}(\lambda, \rho) : \mathcal{A}_1(\text{CRS}, \rho) = 1] \approx$
  $\Pr[(\text{CRS}, \mathsf{trap}) \leftarrow \mathsf{crs\text{-}sim}(\lambda, \rho)\colon \mathcal{A}_1(\text{CRS}, \rho) = 1]$,
  and
  $\Pr[(\text{CRS}, \mathsf{trap}) \leftarrow \mathsf{crs\text{-}sim}(\lambda, \rho) : \mathcal{A}_2^{\mathsf{prover}(\text{CRS},\cdot,\cdot,\cdot)}(\text{CRS}, \rho, \mathsf{trap}) = 1] \approx$
  $\Pr[(\text{CRS}, \mathsf{trap}) \leftarrow \mathsf{crs\text{-}sim}(\lambda, \rho) : \mathcal{A}_2^{\mathsf{sim}^*(\text{CRS},\mathsf{trap},\cdot,\cdot,\cdot)}(\text{CRS}, \rho, \mathsf{trap}) = 1]$,
  where $\mathcal{A}_2$ is restricted to calling the oracle only on (x, w, l) with $(x, w) \in R_\rho$.
- *The QA-NIZK is called* **true-simulation-sound** *[Har11] if the verifier is sound even when an adaptive adversary has access to simulated proofs on language members. More precisely, for all PPT $\mathcal{A}$,*

$$\Pr\left[\begin{array}{c}(\text{CRS}, \mathsf{trap}) \leftarrow \mathsf{crs\text{-}sim}(\lambda, \rho)\\ (x, l, \pi) \leftarrow \mathcal{A}^{\mathsf{sim}(\text{CRS},\mathsf{trap},\cdot,\cdot)}(\text{CRS}, \rho)\end{array} : x \notin L_\rho \wedge \mathsf{ver}(\text{CRS}, x, l, \pi) = 1\right] \approx 0,$$

where the experiment aborts if the oracle is called with some $y \notin L_\rho$.

– *The simulator is said to generate* **unique acceptable proofs** *if for all x, all labels l, and all proofs π^*,*

$$\Pr\left[\begin{array}{c}(\text{CRS}, \mathsf{trap}) \leftarrow \mathsf{crs\text{-}sim}(\lambda, \rho) \\ \pi \leftarrow \mathsf{sim}(\text{CRS}, \mathsf{trap}, x, l)\end{array} : (\pi^* \neq \pi) \wedge \mathsf{ver}(\text{CRS}, x, l, \pi^*) = 1\right] \approx 0.$$

General Construction. We now show that given:

1. An ϵ-smooth and ϵ-universal$_2$ (labeled) projective hash proof system for the collection $\{L_\rho\}_{\rho \in \mathsf{Lpar}}$, and
2. A composable zero-knowledge, true-simulation-sound QA-NIZK Q= (pargen, crsgen, prover, ver, $\mathsf{crs\text{-}sim}$, sim) for the augmented parameterized language $L^*_{\rho,s}$ with probability distribution $\mathcal{D}'$, such that the simulator *generates unique acceptable proofs*, and
3. Efficient algorithms (E_1, E_2) s.t. $\mathcal{D}$ is efficiently witness-samplable using (E_1, E_2), and
4. An efficient algorithm E_3 to sample uniformly from Π,

one can construct a DSS-QA-NIZK for $\{L_\rho\}_{\rho \in \mathsf{Lpar}}$ with probability distribution $\mathcal{D}$. We first give the construction, and then prove the required properties. The QA-NIZK Q *need not take any labels as input.* The various components of the dual-system non-interactive proof system Σ are as follows.

Real World consisting of:

– The algorithm K_0 takes a unary string 1^m as input and generates parameters λ using pargen of Q on 1^m. The CRS generation algorithm K_1 uses crsgen of Q and produces the CRS as follows: it takes λ and the language parameter ρ, and first samples k uniformly from K_λ (recalling that the hash function families are ensembles, one for each λ). It then outputs the CRS to be the pair $(\mathsf{crsgen}(\lambda, \langle \rho, \alpha(k)\rangle), \alpha(k))$.

– The **prover** P takes a CRS (σ, s), input x, witness w, and label l and outputs the proof to be $(T,\ W)$ where T is computed using the public evaluation algorithm $\hat{H}$ as $\hat{H}(s, \langle x, l\rangle, w)$ and $W = \mathsf{prover}(\sigma, \langle x, T, l\rangle, w)$.

– The **verifier** V on input CRS $= (\sigma', s)$, x, l, and proof (T, W), returns the value $\mathsf{ver}(\sigma', \langle x, T, l\rangle,\ W)$ (using ver of Q).

Partial-Simulation World consisting of:

– The **semi-functional CRS simulator** sfK_1 takes λ as input and samples (ρ, ψ) using E_1, and also samples k uniformly from K_λ. It then uses $\mathsf{crs\text{-}sim}$ of Q, and key projection algorithm α to generate the CRS σ as follows: Let $(\sigma', \mathsf{trap}) = \mathsf{crs\text{-}sim}(\lambda, \langle \rho, \alpha(k)\rangle)$. The CRS σ is then the pair $(\sigma', \alpha(k))$. sfK_1 also outputs k, trap as proof simulator trapdoors τ, and ρ, ψ, k as private verifier trapdoors η.

– The **semi-functional simulator** sfSim uses trapdoors k, trap to produce a (partially-simulated) proof for a word x, a label l and a binary bit β using sim of Q as follows: if $\beta = 1$, output

$$T = H_k(x, l),\ W = \mathsf{sim}(\sigma, \mathsf{trap}, \langle x, T, l\rangle),$$

else sample π' at random from Π (using E_3) and output

$$T = \pi' \ , \ W = \mathsf{sim}(\sigma, \mathsf{trap}, \langle x, T, l \rangle).$$

This proof is partially simulated as it uses the bit β.

– The **private verifier** pV uses trapdoors (ρ, ψ, k) to check a word x, label l and a proof T, W as follows: it outputs 1 iff (a) E_2 using ρ and ψ confirms that x is in L_ρ, and (b) $H_k(x, l) = T$, and (c) verifier of Q accepts, i.e. $\mathsf{ver}(\sigma, \langle x, T, l \rangle, W) = 1$.

One-time Full Simulation World consisting of:

– The **one-time full-simulation CRS generator** otfK_1 takes as input λ and language parameter ρ, and using $\mathsf{crs\text{-}sim}$ of Q outputs σ as follows: first it samples k uniformly from K_λ. Let $(\sigma', \mathsf{trap}) = \mathsf{crs\text{-}sim}(\lambda, \langle \rho, \alpha(k) \rangle)$. Then $\sigma = (\sigma', \alpha(k))$. otfK_1 also outputs k, trap as proof simulator trapdoors τ and τ_1, and outputs k as private verifier trapdoor η.

– The **one-time full simulator** otfSim takes as input the trapdoors k, trap and a word x and a label l to produce a proof as follows:

$$T = H_k(x, l), \quad W = \mathsf{sim}(\sigma, \mathsf{trap}, \langle x, T, l \rangle).$$

– The **semi-functional verifier** sfV uses trapdoors k to verify a word x, a label l and a proof T, W as follows: output 1 iff (a) $H_k(x, l) = T$, and (b) $\mathsf{ver}(\sigma, \langle x, T, l \rangle, W) = 1$.

Theorem 1. *For a parameterized class of languages* $\{L_\rho\}_{\rho \in \mathsf{Lpar}}$ *with probability distribution* $\mathcal{D}$*, if the above four conditions hold for projective hash family* H*, QA-NIZK* Q*, and efficient algorithms* E_1, E_2, E_3*, then the above dual-system non-interactive proof system* Σ *is a DSS-QA-NIZK for* $\{L_\rho\}_{\rho \in \mathsf{Lpar}}$ *with probability distribution* $\mathcal{D}$*.*

Remark. In [JR14b] we instantiate the general construction for linear subspaces of vector spaces of hard bilinear groups. As a corollary, it follows that under the SXDH assumption the Diffie-Hellman (DH) language has a DSS-QA-NIZK with only two group elements.

Due to space limitations, we will focus on only the proof of one-time zero-knowledge (otzk) property, as that is the most non-trivial proof. Indeed, this property is a significant generalization of the usual dual-system technique employed in IBE constructions because although in otzk only one proof needs to be fully simulated (i.e. without its membership bit being available), all the private verifier calls in the partial-simulation world need to be simulated in the otzk world without the quasi-adaptive trapdoors (i.e. trapdoor obtained by witness-sampling the language parameters). Recall, in the IBE construction the ciphertext is the counterpart of our verifier, and the IBE private keys are the QA-NIZK proofs. Thus, in IBE only a single ciphertext needs to be simulated when the different private keys are being "fixed" one-by-one by otzk simulation.

The detailed proof of all other properties is given in [JR14b]. The main idea of the proof of these properties is already sketched earlier in this section.

Lemma 2. *In the context of Theorem 1, let the maximum probability that the simulator of Q does not generate unique acceptable proofs be δ. Let H be an ϵ-smooth and ϵ-universal$_2$ (labeled) projective hash proof system for the collection $\{L_\rho\}_{\rho \in \mathsf{Lpar}}$. Let M be the number of calls to the second oracle (verifier) by $\mathcal{A}_3$ and $\mathcal{A}_4$ combined in the two experiments of the one-time full-ZK property of DSS-QA-NIZK Σ. Then the maximum statistical distance (over all PPT Adversaries $\mathcal{A}_3$ and $\mathcal{A}_4$) between the views of the adversaries $(\mathcal{A}_3, \mathcal{A}_4)$ in these two experiments, denoted* $\text{DIST}^{otzk}(\Sigma)$*, is at most* $(\epsilon + \delta) * (1 + M)$.

Proof. We will show that the one-time full-ZK property holds statistically. We will define a sequence of experiments and show that the view of the PPT adversary is statistically indistinguishable in every two consecutive experiments. The first experiment $\mathbf{H}_0$ is identical to the partial-simulation world. First, note that ρ is identically generated using $\mathcal{D}$ in both worlds. Next, note that the CRS σ and trapdoors τ generated by sfK_1 is identically distributed to the CRS σ and both the trapdoors τ and τ_1 generated by otfK_1.

The next experiment $\mathbf{H}_1$ is identical to $\mathbf{H}_0$ except that on $\mathcal{A}_3$ supplied input (x^*, l^*, β^*) the proof π^* generated by sfSim is replaced by proof generated by otfSim. If β^* provided by $\mathcal{A}_3$ is not the valid membership bit for x^* then both experiments abort. So, assume that β^* is the correct membership bit. In case $\beta^* = 1$, both sfSim and otfSim behave identically. When $\beta^* = 0$, the random T^* produced by sfSim is identically distributed to the T^* generated by $H_k(x^*, l^*)$ since H is assumed to be smooth.

The next experiment $\mathbf{H}_2$ is identical to $\mathbf{H}_1$ except that the second oracle is replaced by sfV (from being pV). In order to show that the view of the adversary is indistinguishable in experiments $\mathbf{H}_2$ and $\mathbf{H}_1$, we define several hybrid experiments $\mathbf{H}_{1,i}$ (for $0 \leq i \leq N$, where N is the total number of calls to the second-oracle by $\mathcal{A}_3$ and $\mathcal{A}_4$ combined). Experiment $\mathbf{H}_{1,0}$ is identical to $\mathbf{H}_1$, and the intermediate experiments are defined inductively, by modifying the response of one additional second-oracle call starting with the last (N-th) second-oracle call, and ending with the changed response of the first second-oracle call. The last hybrid experiment $\mathbf{H}_{1,N}$ will then be same as $\mathbf{H}_2$. The second-oracle call response in experiment $\mathbf{H}_{1,i+1}$ differs only in the $(N-i)$-th second-oracle call response in $\mathbf{H}_{1,i}$. In the latter experiment, this call is still served as in $\mathbf{H}_1$ (i.e. using pV). In the former experiment $\mathbf{H}_{1,i+1}$, the $(N-i)$-th call is responded to as defined in $\mathbf{H}_2$ above (i.e. using sfV).

To show that the view of the adversary is statistically indistinguishable in $\mathbf{H}_{1,i}$ and $\mathbf{H}_{1,i+1}$, first note that the view of the adversary ($\mathcal{A}_3$ and A_4 combined) till it's $(N-i)$-th call in both experiments is identical. Moreover, as we next show, the dependence on k of this partial view (i.e. till the $(N-i)$-th call) is limited to $\alpha(k)$ and at most one evaluation of H_k (by otfSim) on an input that is not in L_ρ. To start with, the CRS generated by sfK_1 depends only on $\alpha(k)$. Next, the first oracle sfSim produces T using H_k on its input only if the membership bit β is 1 and correct, and since H is projective this hash value is then completely determined by $\alpha(k)$. Finally, all calls to the second oracle till the $(N-i)$-th call are still served using pV, and again using the projective property of H, it is clear

that the conjunct (b) in pV can be computed using only $\alpha(k)$, because for non L_ρ members, the conjunct (a) is already false, and hence (b) is redundant.

Now, the difference in the $(N-i)$-th call is that the conjunct (a) of pV is missing in sfV. Let x, l, T, W be the input supplied by the PPT Adversary to this call. If $H_k(x, l)$ is not equal to the supplied T, then both pV and sfV return 0. So, suppose $H_k(x, l)$ is equal to T, and yet x is not in L_ρ, i.e. conjunct (a) of pV is false. First, if this input x, l, T, W is same as (x^*, l^*, T^*, W^*) associated with the one-time call to otfSim, then the experiment aborts. Thus, we can assume that this is a different input. If (x, l) is same as (x^*, l^*), then $(T, W) \neq (T^*, W^*)$. Now, by construction (i.e. by definition of otfSim) $T^* = H_k(x^*, l^*)$, and hence either $T \neq H_k(x, l)$ which is not possible by hypothesis, or $(x, l, T) = (x^*, l^*, T^*)$ and $W \neq W^*$. But, W^* is proof generated by the simulator of Q, and since the simulator of Q generates unique acceptable proofs (by assumption), the verifier ver of Q rejects (x, l, T, W), and thus both pV and sfV return 0.

On the other hand, if $(x, l) \neq (x^*, l^*)$ then by the ϵ-universal$_2$ property of H, the probability of T being same as $H_k(x, l)$ is at most ϵ. Thus, both pV and sfV return 0. That completes the induction step, and thus the view of the adversary in experiments $\mathbf{H}_1$ and $\mathbf{H}_2$ is statistically indistinguishable.

The next experiment $\mathbf{H}_3$ is identical to $\mathbf{H}_2$ except that the CRS is generated using otfK_1. The only difference is that the (verifier) trapdoor does not include ρ, ψ. But, since the second oracle is served by sfV and it does not need ρ, ψ, the experiment $\mathbf{H}_3$ is well-defined and statistically indistinguishable from $\mathbf{H}_2$, Further, $\mathbf{H}_3$ is identical to the one-time simulation world, and that completes the proof.

The statistical distance between the views of the adversaries $(\mathcal{A}_3, \mathcal{A}_4)$ in $\mathbf{H}_0$ and $\mathbf{H}_3$ is at most $(\epsilon + \delta) * (1 + M)$. □

5 Keyed-Homomorphic CCA Encryption

Keyed-Homomorphic Encryption is a primitive, first developed in [EHO+13], which allows homomorphic operations with a restricted evaluation key, while preserving different flavors of semantic security depending on whether access to the evaluation key is provided or not. For an adversary not having access to the evaluation key, the homomorphic operation should not be available and this is ensured by requiring CCA security. However, if an adversary comes into possession of the evaluation key, CCA security can no longer be preserved and thus weaker forms of security, such as CCA1, are required. In [LPJY14], the authors gave improved constructions for multiplicative homomorphism with better security guarantees.

A **KH-PKE** scheme consists of algorithms $(KeyGen, Enc, Dec, Eval)$, where the first three are familiar from public-key encryption, and KeyGen generates a public key pk, a decryption key sk_d and an Eval key sk_h. Algorithm Eval takes two ciphertexts and returns a ciphertext or $\perp$. Detailed definitions can be found in [JR14b]. The scheme is said to be correct if (i) for Enc we have $Dec(sk_d, Enc(pk, M)) = M$, where sk_d is the secret decryption key, and (ii) for

Eval we have $Dec(sk_d, Eval(sk_h, C_1, C_2)) = Dec(sk_d, C_1) \odot Dec(sk_d, C_2)$, where $\odot$ is a binary operation on plaintexts, and if any operand of $\odot$ is $\perp$ then the result is $\perp$. The KH-PKE scheme is defined to be **KH-CCA** secure by a usual public-key CCA experiment with the following twists: the challenger maintains a set D of ciphertexts dependent on the challenge ciphertext (via Eval); decryption queries are not allowed on ciphertexts in D. Further, an adversary $\mathcal{A}$ can adaptively ask for sk_h, which we call the *reveal event*. After the reveal event, the Eval oracle is not available. Similarly, decryption is not available after $\mathcal{A}$ has both requested sk_h and obtained the challenge ciphertext, in any order. Again, detailed definitions can be found in [JR14b].

Construction. We present a construction of a KH-CCA secure KH-PKE encryption scheme with multiplicative homomorphism which utilizes our general DSS-QA-NIZK construction for the Diffie-Hellman (DH) language. In fact, if we assume that the adversary never invokes RevHK, we can prove security generically assuming any DSS-QA-NIZK (with statistical one-time full-ZK) for the DH language. When the adversary invokes RevHK, the partial-simulation trapdoor is revealed to the Adversary, and hence the one-time full-ZK property of DSS-QA-NIZK may not hold. Thus, we a need a stronger notion of DSS-QA-NIZK that incorporates the reveal event, and includes an additional requirement that the semi-functional verifier remains sound as before. Using this stronger notion, we can prove generic security of the KH-PKE scheme even with RevHK, and we further show that our general construction of Section 4 continues to satisfy this stronger property.

We start with the observation that a standard ElGamal encryption scheme $(\mathbf{g}^x, m \cdot \mathbf{f}^x)$ is multiplicatively homomorphic, but is not CCA secure due to the exact same reason. The main idea of our construction is as follows. The ciphertexts include an ElGamal encryption of the message M, say $\mathbf{g}^r, M \cdot \mathbf{g}^{kr}$ for a public key $\mathbf{g}^k$. The public key also consists of a member $\mathbf{g}^a$, and the ciphertext also include $\mathbf{g}^{ar}$ (we refer to this triple in the ciphertext as *augmented ElGamal encryption*). It is well-known [JR12] that if a one-time simulation-sound NIZK proof of $\mathbf{g}^r$ and $\mathbf{g}^{ar}$ being of the correct form is also included in the ciphertext then it becomes a publicly-verifiable CCA2-secure encryption scheme. In our keyed-homomorphic construction, we include a DSS-QA-NIZK for $\mathbf{g}^r$ and $\mathbf{g}^{ar}$ being of the correct form (i.e. being a DH tuple). Although the DSS-QA-NIZK itself is not homomorphic, we can take advantage of the corresponding Semi-Functional Simulator sfSim and simulate the proof of a multiplicatively generated (augmented) ElGamal encryption when computing a homomorphic evaluation.

So, given a dual-system non-interactive proof Σ, consider the following algorithms for a KH-PKE scheme $\mathcal{P}$:

KeyGen: Generate $\mathbf{g}, a, k$ randomly. Use sfK_1 of Σ to get CRS σ and trapdoors τ and η, and language parameters $\rho = (\mathbf{g}, \mathbf{g}^a)$. Set $pk = (\mathbf{g}, \mathbf{g}^a, \mathbf{g}^k, \sigma)$, $sk_h = \tau$, $sk_d = k$.

Enc: Given plaintext m, generate $w \leftarrow \mathbb{Z}_q$ and compute (using P of Σ)
$c := (\mathbf{g}^w, \mathbf{g}^{aw}, \gamma, \mathsf{P}(\sigma, (\mathbf{g}^w, \mathbf{g}^{aw}), w, l = \gamma))$, where $\gamma := m \cdot \mathbf{g}^{kw}$.

Dec: Given ciphertext $c = (\rho, \hat{\rho}, \gamma, \pi)$, first check if $\mathsf{V}(\sigma, \pi, (\rho, \hat{\rho}), \gamma)$ of Σ holds, then compute $m := \gamma/\rho^k$.

Eval (multiplicative): Given ciphertexts $c_1 = (\rho_1, \hat{\rho}_1, \gamma_1, \pi_1)$ and $c_2 = (\rho_2, \hat{\rho}_2, \gamma_2, \pi_2)$, first check if $\mathsf{V}(\sigma, \pi_i, (\rho_i, \hat{\rho}_i), \gamma_i)$ of Σ holds for all $i \in \{1, 2\}$. Then compute: $\rho = \rho_1\rho_2\rho_3$, $\hat{\rho} = \hat{\rho}_1\hat{\rho}_2\hat{\rho}_3$, $\gamma = \gamma_1\gamma_2\gamma_3$, where $\langle \rho_3, \hat{\rho}_3, \gamma_3 \rangle$ is a fresh random tuple obtained by picking r at random and setting the tuple to be $\langle \mathbf{g}^r, (\mathbf{g}^a)^r, (\mathbf{g}^k)^r \rangle$. Then compute $\pi := \mathsf{sfSim}(\sigma, \tau, (\rho, \hat{\rho}), \beta = 1, l = \gamma)$ using sfSim of Σ. Output ciphertext $c := (\rho, \hat{\rho}, \gamma, \pi)$.

Theorem 2 (Security of Construction). *The above algorithms P= (KeyGen, Enc, Dec, Eval) constitute a KH-CCA secure Keyed-Homomorphic Public Key Encryption scheme with multiplicative homomorphism, if Σ is a DSS-QA-NIZK for the parameterized Diffie-Hellman language (with language parameters distributed randomly) and RevHK is not available.*

The main idea of the proof of the above theorem is similar to proofs of CCA2-secure public key encryption schemes using alternate decryption. In other words, the ciphertext can be decrypted as $m := \gamma/\rho^k$, or as $m := \gamma/(\rho^{k_0}\hat{\rho}^{k_1})$, where $k = k_0 + ak_1$. But, this requires that the ciphertext has correct $\hat{\rho}$ component, i.e. $\hat{\rho} = \rho^a$. The ciphertexts include a NIZK for this purpose, but the NIZK needs to be simulation-sound. Additional complication arises because of dependent ciphertexts. To handle this, we first build an intermediate experiment where all dependent ciphertexts are generated using fresh random ElGamal tuples. Indistinguishability of such an intermediate experiment from the KH-CCA experiment is shown inductively, by carefully employing one-time full-ZK and partial-simulation unbounded simulation soundness. The theorem is proved in detail in [JR14b]. The Adversary's advantage in the KH-CCA security game is at most $(8L + 1) \cdot \text{ADV}_{\text{DDH}} + O(L/q)$, where L is the total number of calls to Eval.

The more general theorem (with RevHK) is stated and proved in [JR14b]. Under the SXDH assumption, the above construction leads to ciphertexts of size only five group elements. Further, using an *augmented Diffie Hellman language* (augmented with a smooth hash proof of DH tuple) and its DSS-QA-NIZK, we also extend our result to get CCA1-security despite the key being revealed (see [JR14b]). The resulting scheme has KH-PKE ciphertexts of size six group elements.

6 Single-Round UC Password-Based Key Exchange

The essential elements of the Universal Composability framework can be found in [Can01]. In the following, we adopt the definition for password-based key exchange (UC-PAKE) from Canetti et al [CHK+05].

6.1 UC-PAKE Definition

Just as in the normal key-exchange functionality, if both participating parties are not corrupted, then they receive the same uniformly distributed session key

Functionality $\mathcal{F}_{\text{pake}}$

The functionality $\mathcal{F}_{\text{PAKE}}$ is parameterized by a security parameter k. It interacts with an adversary S and a set of parties via the following queries:

Upon receiving a query (NewSession, $sid, P_i, P_j, pw, role$) **from party** P_i**:**
Send (NewSession, $sid, P_i, P_j, role$) to S. In addition, if this is the first NewSession query, or if this is the second NewSession query and there is a record (P_j, P_i, pw'), then record (P_i, P_j, pw) and mark this record fresh.

Upon receiving a query (TestPwd, sid, P_i, pw') **from the adversary** S**:**
If there is a record of the form (P_i, P_j, pw) which is fresh, then do: If $pw = pw'$, mark the record compromised and reply to S with "correct guess". If $pw \neq pw'$, mark the record interrupted and reply with "wrong guess".

Upon receiving a query (NewKey, sid, P_i, sk) **from** S**, where** $|sk| = k$**:**
If there is a record of the form (P_i, P_j, pw), and this is the first NewKey query for P_i, then:
- If this record is compromised, or either P_i or P_j is corrupted, then output (sid, sk) to player P_i.
- If this record is fresh, and there is a record (P_j, P_i, pw') with $pw' = pw$, and a key sk' was sent to P_j, and (P_j, P_i, pw) was fresh at the time, then output (sid, sk') to P_i.
- In any other case, pick a new random key sk' of length k and send (sid, sk') to P_i.

Either way, mark the record (P_i, P_j, pw) as completed.

Upon receiving (Corrupt, sid, P_i) **from** S**:** if there is a (P_i, P_j, pw) recorded, return pw to S, and mark P_i corrupted.

Fig. 2. The password-based key-exchange functionality $\mathcal{F}_{\text{PAKE}}$

and the adversary learns nothing of the key except that it was generated. However, if one of the parties is corrupted, then the adversary determines the session key. This power to the adversary is *also* given in case it succeeds in guessing the parties' shared password. Participants also detect when the adversary makes an unsuccessful attempt. If the adversary makes a wrong password guess in a given session, then the session is marked interrupted and the parties are provided random and independent session keys. If however the adversary makes a successful guess, then the session is marked compromised, and the adversary is allowed to set the session key. If a session remains marked fresh, meaning that it is neither interrupted nor compromised. uncorrupted parties conclude with both parties receiving the same, uniformly distributed session key. The formal description of the UC-PAKE functionality $\mathcal{F}_{\text{PAKE}}$ is given in Figure 2.

The real-world protocol we provide is also shown to be secure when different sessions use the same common reference string (CRS). To achieve this goal, we consider the *universal Composability with joint state* (JUC) formalism of Canetti and Rabin [CR03]. This formalism provides a "wrapper layer" that deals with "joint state" among different copies of the protocol. In particular, defining a functionality $\mathcal{F}$ also implicitly defines the multi-session extension of $\mathcal{F}$(denoted by $\hat{\mathcal{F}}$): $\hat{\mathcal{F}}$ runs multiple independent copies of $\mathcal{F}$, where the copies are distinguished via sub-session IDs ssid. The JUC theorem [CR03] asserts that for any protocol π that uses multiple independent copies of $\mathcal{F}$, composing π instead with a single copy of a protocol that realizes $\hat{\mathcal{F}}$, preserves the security of π.

Generate $\mathbf{g}_1 \leftarrow \mathbb{G}_1, \mathbf{g}_2 \leftarrow \mathbb{G}_2$ and $a, b, c, d, e, u_1, u_2 \leftarrow \mathbb{Z}_q$, and let $\mathcal{H}$ be a CRHF. Compute $\mathbf{a} = \mathbf{g}_1^a,\ \mathbf{d} = \mathbf{g}_1^d,\ \mathbf{e} = \mathbf{g}_1^e,\ \mathbf{w}_1 = \mathbf{g}_1^{u_1},\ \mathbf{w}_2 = \mathbf{g}_1^{u_2}$ $\mathbf{b} = \mathbf{g}_2^b,\ \mathbf{c} = \mathbf{g}_2^c,\ \mathbf{v}_1 = \mathbf{g}_2^{u_1 b - d - ca},\ \mathbf{v}_2 = \mathbf{g}_2^{u_2 b - e}$. CRS $:= (\mathbf{g}_1, \mathbf{g}_2, \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{w}_1, \mathbf{w}_2, \mathbf{v}_1, \mathbf{v}_2, \mathcal{H})$.	
Party P_i	Network
Input $(\texttt{NewSession}, sid, \mathsf{ssid}, P_i, P_j, \text{pwd}, initiator/responder)$ Choose $r_1, s_1 \xleftarrow{\$} \mathbb{Z}_q$. Set $R_1 = \mathbf{g}_1^{r_1}$, $S_1 = \text{pwd} \cdot \mathbf{a}^{r_1}$, $T_1 = (\mathbf{d} \cdot \mathbf{e}^{i_1})^{r_1}$, $\hat{\rho}_1 = \mathbf{b}^{s_1}$, $W_1 = (\mathbf{w}_1 \mathbf{w}_2^{i_1})^{r_1}$, where $i_1 = \mathcal{H}(sid, \mathsf{ssid}, P_i, P_j, R_1, S_1, \hat{\rho}_1)$ and erase r_1. Send R_1, S_1, T_1 and $\hat{\rho}_1$, and retain W_1.	$\xrightarrow{R_1, S_1, T_1, \hat{\rho}_1}\ P_j$
Receive $R_2', S_2', T_2', \hat{\rho}_2'$. If any of $R_2', S_2', T_2', \hat{\rho}_2'$ is not in their respective group or is 1, set $\text{sk}_1 \xleftarrow{\$} \mathbb{G}_T$, else compute $i_2' = \mathcal{H}(sid, \mathsf{ssid}, P_j, P_i, R_2', S_2', \hat{\rho}_2')$, $\rho_1 = \mathbf{g}_2^{s_1},\ \theta_1 = \mathbf{c}^{s_1},\ \gamma_1 = (\mathbf{v}_1 \mathbf{v}_2^{i_2'})^{s_1}$. Compute $\text{sk}_1 = e(T_2', \rho_1) \cdot e(S_2'/\text{pwd}, \theta_1) \cdot e(R_2', \gamma_1) \cdot e(W_1, \hat{\rho}_2')$ Output $(sid, \mathsf{ssid}, \text{sk}_1)$.	$\xleftarrow{R_2', S_2', T_2', \hat{\rho}_2'}\ P_j$

Fig. 3. Single round UC-secure Password-authenticated KE under SXDH Assumption.

6.2 Main Idea of the UC Protocol using DSS-QA-NIZK

For the sake of exposition, let's call one party in the session the server and the other the client. (There is no such distinction in the actual protocol, and in fact each party will run two parallel protocols, one as a client and another as a server, and output the product of the two keys generated). The common reference string (CRS) defines a Diffie-Hellman language, i.e. $\rho = \mathbf{g}_1, \mathbf{g}_1^a$. The client picks a fresh Diffie-Hellman tuple by picking a witness r and computing $\langle \mathbf{x}_1 = \mathbf{g}_1^r, \mathbf{x}_2 = \mathbf{g}_1^{a \cdot r} \rangle$. It also computes a DSS-QA-NIZK proof on this tuple, which is a hash proof T and a QA-NIZK proof W of the augmented Diffie-Hellman tuple. Note, the QA-NIZK proof W is just a single group element [JR14a] (see [JR14b] for details). It next modifies the Diffie-Hellman tuple using the password pwd it possesses. Essentially, it multiplies $\mathbf{x}_2$ by pwd to get a modified group element which we will denote by S - in fact $(\mathbf{x}_1, S)$ is an ElGamal encryption of pwd. It next sends this ElGamal encryption $\mathbf{x}_1, S$ and the T component of the proof to the server. It retains W for later use. At this point it can erase the witness r.

As a first step, we intend to utilize an interesting property of the real-world verifier V of the DSS-QA-NIZK: the verifier is just the verifier of the QA-NIZK for the DH language augmented with the hash proof, and the QA-NIZK verifiers for linear subspaces are just a single bi-linear product test. Specifically (see [JR14b]), V on input $\mathbf{x}_1, \mathbf{x}_2$ and proof T, W, computes $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$, and outputs true iff

$$e(\mathbf{x}_1, (\mathbf{v}_1 \mathbf{v}_2^{\iota})) \cdot e(\mathbf{x}_2, \mathbf{c}) \cdot e(T, \mathbf{g}_2) \ = \ e(W, \mathbf{b}).$$

Thus, it outputs true iff the left-hand-size (LHS) equals the right-hand-side (RHS) of the above equation. Note that the client sent $\mathbf{x}_1, S$ (i.e. $\mathbf{x}_2$ linearly modified by pwd) and T to the server. Assuming the server has the same password pwd, it can un-modify the received message and get $\mathbf{x}_2 = S/\text{pwd}$, and hence can compute this LHS (using the CRS). The client retained W, and can compute the RHS (using the CRS).

The intuition is that unless an adversary out-right guesses the password, it cannot produce a different $\mathbf{x}'_1$, S', T', such that $\mathbf{x}'_1$, S'/pwd, T' used to compute the LHS will match the RHS above. While we make this intuition rigorous later by showing a UC simulator, to complete the description of the protocol, and using this intuition, the client and server actually compute the LHS and RHS respectively of the following equation (for a fresh random $s \in \mathbb{Z}_q$ picked by the server):

$$e(\mathbf{x}_1, (\mathbf{v}_1\mathbf{v}_2^{\iota})^s) \cdot e(\mathbf{x}_2, \mathbf{c}^s) \cdot e(T, \mathbf{g}_2^s) \;=\; e(W, \mathbf{b}^s). \tag{1}$$

Now note that for the client to be able to compute the RHS, it must have $\mathbf{b}^s$, since s was picked by the server afresh. For this purpose, the protocol requires that the server send $\mathbf{b}^s$ to the client (note this can be done independently and asynchronously of the message coming from the client). It is not difficult to see, from completeness of the prover and verifier of the DSS-QA-NIZK, that both parties compute the same quantity.

As mentioned earlier, each pair of parties actually run two versions of the above protocol, where-in each party plays the part of client in one version, and the part of server in the other version. Each party then outputs the product of the LHS of (1) computation (in the server version) and the RHS of (1) computation (in the client version) as the session-key. We will refer to these two factors in the session-key computation as the *server factor* and the client factor resp. This is the final UC-PAKE protocol described in Fig. 6.1 (with the parties identities, session identifiers and $\mathbf{b}^s$ from its server version, used as label). The quantity $\mathbf{x}_1$ is called R in the protocol, as subscripts will be used for other purposes.

Theorem 3. *Assuming the existence of SXDH-hard groups, the protocol given in Fig 6.1 securely realizes the $\widehat{\mathcal{F}}_{\text{PAKE}}$ functionality in the $\mathcal{F}_{\text{CRS}}$ hybrid model, in the presence of adaptive corruption adversaries.*

The theorem is proved in [JR14b]. We provide the intuition below.

6.3 Main Idea of the UC Simulator

We first *re-define* the various verifiers in the DSS-QA-NIZK for the DH language described in [JR14b], to bring them in line with the above description. In particular, the real-world verifier V is defined equivalently to be: the verifier V takes as input $\mathbf{CRS}_v$, a word $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$, and a proof $\pi = (T, W)$, computes $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$, picks a fresh random $s \in \mathbb{Z}_q$, and outputs true iff

$$e(\mathbf{x}_1, (\mathbf{v}_1\mathbf{v}_2^{\iota}))^s \cdot e(\mathbf{x}_2, \mathbf{c})^s \cdot e(T, \mathbf{g}_2)^s \;=\; e(W, \mathbf{b}^s).$$

This is equivalent as long as $s \neq 0$.

The partial-simulation world private-verifier pV is now defined as: it checks a word $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and a proof T, W as follows: compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$; pick s and s' randomly and independently from $\mathbb{Z}_q$, and if $\mathbf{x}_2 = \mathbf{x}_1^a$ and $T = \mathbf{x}_1^{d+\iota e}$ then set $\xi = \mathbf{1}_T$ else set $\xi = e(\mathbf{g}_1, \mathbf{g}_2)^{s'}$ and output true iff

$$e(\mathbf{x}_1, (\mathbf{v}_1 \mathbf{v}_2^{\iota}))^s \cdot e(\mathbf{x}_2, \mathbf{c})^s \cdot e(T, \mathbf{g}_2)^s \cdot \xi \;=\; e(W, \mathbf{b}^s). \tag{2}$$

This is equivalent to the earlier definition of pV with high probability by an information-theoretic argument, if the trapdoors used were generated by the semi-functional CRS generator sfK_1.

The UC simulator $\mathcal{S}$ works as follows: It will generate the CRS for $\widehat{\mathcal{F}}_{\text{PAKE}}$ using the semi-functional CRS generator sfK_1 for the Diffie-Hellman language. The next main difference is in the simulation of the outgoing message of the real world parties: $\mathcal{S}$ uses a dummy message μ instead of the real password which it does not have access to. Further, it postpones computation of W till the session-key generation time. Finally, another difference is in the processing of the incoming message, where $\mathcal{S}$ decrypts the incoming message R_2', S_2', T_2' to compute a pwd', which it uses to call the ideal functionality's test function. It next generates a sk similar to how it is generated in the real-world (recall the computation of server factor and client factor by LHS and RHS of (1)) except that it uses the equation (2) corresponding to the private verifier. It sends sk to the ideal functionality to be output to the party concerned.

Note, $\mathcal{S}$ simulating the server factor computation can compute the LHS of equation (2), except $\mathcal{S}$ does not have direct access to pwd and hence cannot get $\mathbf{x}_2$ from the modified $\hat{S}$ that it receives. However, it can do the following: Use the $\mathsf{TestPwd}$ functionality of the ideal functionality $\widehat{\mathcal{F}}_{\text{PAKE}}$ with a pwd' computed as $\hat{S}/\mathbf{x}_1^a$. If this pwd' does not match the pwd recorded in $\widehat{\mathcal{F}}_{\text{PAKE}}$ for this session and party, then $\widehat{\mathcal{F}}_{\text{PAKE}}$ anyway outputs a fresh random session key, which will then turn out to be correct simulation (note, this case is same as $\mathbf{x}_2 (= S/\text{pwd}) \neq \mathbf{x}_1^a$, which would also have resulted in the same computation on the LHS). If the pwd' matched the pwd, the simulator is notified the same, and hence it can now do the following: if $T = \mathbf{x}_1^{d+\iota e}$ then set $\xi = \mathbf{1}_T$ else set $\xi = e(\mathbf{g}_1, \mathbf{g}_2)^{s'}$. Next, it calls $\widehat{\mathcal{F}}_{\text{PAKE}}$'s NewKey with session key $e(\mathbf{x}_1, (\mathbf{v}_1 \mathbf{v}_2^{\iota}))^s \cdot e(\mathbf{x}_1^a, \mathbf{c})^s \cdot e(T, \mathbf{g}_2)^s \cdot \xi$ (multiplied by a RHS computation of (2) in simulation of the client factor, which we will discuss later).

The UC Simulator $\mathcal{S}$ must also simulate $\mathbf{g}_1^r, \text{pwd} \cdot (\mathbf{g}_1^a)^r$ and the T component of the DSS-QA-NIZK, as that is the message sent out to the adversary by the real party ("client" part of the protocol). However, $\mathcal{S}$ does not have access to pwd. It can just generate a fake tuple $\mathbf{g}_1^r, \mu \cdot (\mathbf{g}_1^a)^r \cdot \mathbf{g}_1^{r'}$ (for some constant or randomly chosen group element μ, and some random and independent $r' \in \mathbb{Z}_q$). Now, the semi-functional (proof) simulator sfSim of the DSS-QA-NIZK of [JR14b] has an interesting property that when the tuple $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ does not belong to the language (language membership-bit zero), the T component of the simulated proof can just be generated randomly.

The simulator also needs W to compute the client factor, and we had postponed it till the session-key computation phase. As mentioned above, if the

password pwd' "decrypted" from the incoming message is not correct then the key is anyway set to be random, and hence a proper W is not even required. However, if the pwd' is correct, the simulator is notified of same, and hence it can compute W component of the proof by passing $\mathbf{x}_2 = \mu \cdot (\mathbf{g}_1^a)^r \cdot \mathbf{g}_1^{r'}/\mathrm{pwd}'$ along with $\mathbf{x}_1$ $(= \mathbf{g}_1^r)$ to sfSim.

Of course, fixing the above fake tuples employs one-time full-simulation property of the DSS-QA-NIZK (and the DDH assumption).

6.4 Main Idea of the Proof of UC Realization

The proof that the simulator $\mathcal{S}$ described above simulates the Adversary in the real-world protocol, follows essentially from the properties of the DSS-QA-NIZK, although not generically since the real-world protocol and the simulator use the verifiers V and pV (resp.) in a split fashion. However, as described above the proof is very similar and we give a broad outline here. The proof will describe various experiments between a challenger $\mathcal{C}$ and the adversary, which we will just assume to be the environment $\mathcal{Z}$ (as the adversary $\mathcal{A}$ can be assumed to be just dummy and following $\mathcal{Z}$'s commands). In the first experiment the challenger $\mathcal{C}$ will just be the combination of the code of the simulator $\mathcal{S}$ above and $\widehat{\mathcal{F}}_{\text{PAKE}}$. In particular, after the environment issues a $\mathsf{NewSession}$ request with a password pwd, the challenger gets that password. So, while in the first experiment, the challenger (copying $\mathcal{S}$) does not use pwd directly, from the next experiment onwards, it can use pwd. Thus, the main goal of the ensuing experiments is to modify the fake tuples $\mathbf{g}_1^r, \mu \cdot (\mathbf{g}_1^a)^r \cdot \mathbf{g}_1^{r'}$ by real tuples (as in real-world) $\mathbf{g}_1^r, \mathrm{pwd} \cdot (\mathbf{g}_1^a)^r$, since the challenger has access to pwd. This is accomplished by a hybrid argument, modifying one instance at a time using DDH assumption in group $\mathbb{G}_1$ and using one-time full-ZK property (and using the otfSim proof simulator for that instance). A variant of the one-time full-ZK semi-functional verifier sfV (just as the variants for pV and V described above) is easily obtained. Note that in each experiment, whenever the simulator invokes partial proof simulation it can provide the correct membership bit (with high probability) as in each experiment it knows exactly which tuples are real and which are fake.

Once all the instances are corrected, i.e. R, S generated as $\mathbf{g}_1^r, \mathrm{pwd} \cdot (\mathbf{g}_1^a)^r$, the challenger can switch to the real-world because the tuples $R, S/\mathrm{pwd}$ are now Diffie-Hellman tuples. This implies that the session keys are generated using the V variant described above, which is exactly as in the real-world.

6.5 Adaptive Corruption

The UC protocol described above is also UC-secure against adaptive corruption of parties by the Adversary in the erasure model. In the real-world when the adversary corrupts a party (with a $\mathsf{Corrupt}$ command), it gets the internal state of the party. Clearly, if the party has already been invoked with a $\mathsf{NewSession}$ command then the password pwd is leaked at the minimum, and hence the ideal functionality $\mathcal{F}_{\text{PAKE}}$ leaks the password to the Adversary in the ideal world.

In the protocol described above, the Adversary also gets W and s, as this is the only state maintained by each party between sending $R, S, T, \hat{\rho}$, and the final issuance of session-key. Simulation of s is easy for the simulator $\mathcal{S}$ since $\mathcal{S}$ generates s exactly as in the real world. For generating W, which $\mathcal{S}$ had postponed to computing till it received an incoming message from the adversary, it can now use the pwd which it gets from $\widehat{\mathcal{F}}_{\text{PAKE}}$ by issuing a Corrupt call to $\widehat{\mathcal{F}}_{\text{PAKE}}$. More precisely, it issues the Corrupt call, and gets pwd, and then calls the semi-functional simulator with $\mathbf{x}_2 = \mu \cdot (\mathbf{g}_1^a)^r \cdot \mathbf{g}_1^{r'}/\text{pwd}$ along with $\mathbf{x}_1 (= \mathbf{g}_1^r)$ to get W. Note that this computation of W is identical to the postponed computation of W in the computation of client factor of sk_1 (which is really used in the output to the environment when $\text{pwd}' = \text{pwd}$).

References

[ABB+13] Michel Abdalla, Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, and David Pointcheval. SPHF-friendly non-interactive commitments. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 214–234. Springer, December 2013.

[ABP15] Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. Disjunctions for hash proof systems: New constructions and applications. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 69–100. Springer, April 2015.

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

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

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

[CCS09] Jan Camenisch, Nishanth Chandran, and Victor Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 351–368. Springer, April 2009.

[CHK+05] Ran Canetti, Shai Halevi, Jonathan Katz, Yehuda Lindell, and Philip D. MacKenzie. Universally composable password-based key exchange. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 404–421. Springer, May 2005.

[CR03] Ran Canetti and Tal Rabin. Universal composition with joint state. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 265–281. Springer, August 2003.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, April / May 2002.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *Proc. 23rd ACM STOC*, pages 542–552, 1991.

[EHO+13] Keita Emura, Goichiro Hanaoka, Go Ohtake, Takahiro Matsuda, and Shota Yamada. Chosen ciphertext secure keyed-homomorphic public-key encryption. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 32–50. Springer, February / March 2013.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, April 2008.

[Har11] Kristiyan Haralambiev. Efficient cryptographic primitives for non-interactive zero-knowledge proofs and applications. *PhD Dissertation*, 2011.

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

[JR13] Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 1–20. Springer, December 2013.

[JR14a] Charanjit S. Jutla and Arnab Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 295–312. Springer, August 2014.

[JR14b] Charanjit S. Jutla and Arnab Roy. Dual-system simulation-soundness with applications to UC-PAKE and more. Cryptology ePrint Archive, Report 2014/805. https://eprint.iacr.org/2014/805.

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

[KW15] Eike Kiltz and Hoeteck Wee. Quasi-adaptive NIZK for linear subspaces revisited. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 101–128. Springer, April 2015.

[LPJY14] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 514–532. Springer, May 2014.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, pages 543–553. IEEE Computer Society Press, October 1999.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, August 2009.