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

Charanjit S. Jutla
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
csjutla@us.ibm.com

Arnab Roy
Fujitsu Laboratories of America
Sunnyvale, CA 94058, USA
arnab@cs.stanford.edu

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 potential language member, 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 further 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 in asymmetric bilinear pairing groups.

As a first application we show a first single-round universally-composable password authenticated key-exchange (UC-PAKE) protocol which is secure under dynamic corruption in the erasure model. The single message flow only requires four group elements under the SXDH assumption, which is at least two times shorter than earlier schemes.

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

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

Contents

1 Introduction

Since the introduction of simulation-sound non-interactive zero-knowledge (NIZK) proofs in [Sah99], simulation-soundness has become an essential cryptographic tool. While the idea of zero-knowledge simulation [GMR89] brought rigour 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 proof systems, which lie somewhere in between one-time and unbounded simulation-soundness. 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 we cannot get a generic construction from a NIZK proof, but 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 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 distributions 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 further 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 potential member 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 (and more general such languages) 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. Indeed, for the Diffie-Hellman language, under the standard SXDH assumption in bilinear pairing groups, the DSS-QA-NIZK only requires two more group elements.

We now give the main idea as to why such a construction is useful. Most applications only need a full-simulation NIZK proof for the hard language under whose hardness the application is to be proven secure. However, the 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, and full simulation is only required of one member at a time, and that too of the underlying hard language member.

Applications.

1. 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 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 ideas from 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 only had a quadratic-preserving security reduction, our reduction is linear-preserving.

The main idea of our construction is as follows. The ciphertexts consist of an El-Gamal encryption of the message M, say $\mathbf{g}^r, M \cdot \mathbf{g}^{k \cdot r}$ for a public key $\mathbf{g}^k$. The public key also consists of a member $\mathbf{g}^a$, and the ciphertext includes $\mathbf{g}^{a \cdot r}$. It is well-known [JR12] that if a one-time simulation-sound NIZK proof of $\mathbf{g}^r$ and $\mathbf{g}^{a \cdot r}$ being of the correct form is 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 $\mathbf{g}^{a \cdot r}$ being of the correct form, which is just the requirement that this be a Diffie-Hellman tuple. The DSS-QA-NIZK we give for this language has the nice property that the CRS in the real-world and the partial world are statistically identical. Thus, the public key can be generated by the partial-world CRS simulator, and the partial-simulation trapdoor it generates can be the Eval key. Since the partial simulation world is unbounded simulation-sound, the Eval functionality can be provided by first verifying that the Eval is performed on valid ciphertexts, homomorphically combining the triples of the form $\mathbf{g}^r, \mathbf{g}^{a \cdot r}, M \cdot \mathbf{g}^{k \cdot r}$, and then calling

the partial-simulator with the membership bit set as one (and this is a valid membership bit as the partial-world verifier is unbounded simulation sound). We also show that our DSS-QA-NIZK construction remains CCA1-style secure when the partial-simulator trapdoor is revealed.

2. Universally-Composable 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]. In [JR12], further improvement was obtained by generating the key in the target bilinear-group. The construction still required unbounded simulation-soundness. Later [BBC+13] gave a construction based on novel trapdoor smooth projective hash functions instead of unbounded simulation-sound NIZK proofs. Their construction yielded a single-round UC-secure PAKE scheme under static corruption, and each message consisted of six group elements in one group, and another five elements in the other group, under the SXDH assumption. The key is generated in the target group.

We now give a single-round construction based on dual-system simulation-soundness which is UC-secure under dynamic corruption (in the erasure model), and which has only a total of four group elements in each message (three in one group and one in the other). The key is generated in the target group. The construction is not a generic construction from 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 5.3. All previous constructions, with the exception of a two-round construction of [ACP09], were only secure in the static corruption model.

1.1 Dual-System Simulation-Soundness

A witness relation is a binary relation on pairs of inputs, the first called the witness and the second called a language member. 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 $R(w, x)$ holds.

We will consider Quasi-Adaptive NIZK proofs for a probability distribution $\mathcal{D}$ on a collection of (witness-) relations $\mathcal{R} = \{R_\rho\}$ [JR13]. Recall that in a quasi-adaptive NIZK, the CRS can be set after the language parameter has been chosen according to $\mathcal{D}$. To define dual-system simulation soundness of such 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 or quasi-adaptive 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 potential language member. In case, the proof simulator is being invoked on a non-language member, 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. 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 trapdoors.

Rather than consider 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 potential language member x is given an additional bit β, which we call the **membership bit** which represents the information whether the potential language member x is indeed a member or not.

5

The partial simulation world is required to be unbounded simulation-sound and hopefully it should be easier to prove (given that its simulators have additional information) than usual unbounded simulation-soundness. 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. However, between the real-world and the partial-simulation world one must only pass the prover/simulator real language members.

Dual-system Simulation-Sound Quasi-Adaptive NIZK (DSS-QA-NIZK). A Dual-system proof system for a collection of witness-relations $\mathcal{R}_\lambda = \{R_\rho\}$, with parameters sampled from a distribution $\mathcal{D}$ over associated parameter language $\mathcal{L}_{\mathrm{par}}$ consists of:

- Following real-world PPT components:
  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 $\mathcal{D}_\lambda$. Next, 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 and produces a proof. A **verifier** V that takes as input a CRS, a potential language member, and a proof, and outputs a single bit.

- In addition, there are the following partial-simulation world PPT components:
  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** sfS that takes a CRS, a trapdoor τ, a potential language member and a membership-bit β to produce a proof. A **private verifier** pV that takes a CRS, a trapdoor η, a potential language member and a proof and outputs a single bit.

- Finally, there are the following one-time full simulation PPT components:
  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** otfS that takes as input a CRS, a trapdoor τ_1, and a potential language member and produces a proof. A **semi-functional verifier** sfV that takes as input a CRS, a trapdoor η, a potential language member, a proof and outputs a bit. The adversaries also have access to semi-functional simulator.

All of the above provers, proof simulators and verifiers may also take a **label** l as an additional argument.

The definition of the real-world quasi-adaptive system to be complete and (computationally) sound are standard and can be found in [JR13]. Such a proof system is called a **dual-system simulation-sound quasi-adaptive NIZK** for a collection of witness relations $\mathcal{R}_\lambda = \{R_\rho\}$, with parameters sampled from a distribution $\mathcal{D}$, if for all non-uniform PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4)$ all of the following properties are satisfied:

- *partial-ZK:*
  $$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); \rho \leftarrow \mathcal{D}_\lambda; \psi \leftarrow \mathsf{K}_1(\lambda, \rho); \ \mathcal{A}_1^{\mathsf{P}(\psi, \cdot, \cdot; \cdot), \, \mathsf{V}(\psi, \cdot, \cdot; \cdot)}(\psi) = 1] \approx$$

$$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); (\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda);\ \mathcal{A}_1^{\mathsf{X}(\sigma,\tau,\cdot,\cdot;\cdot),\,\mathsf{pV}(\sigma,\eta,\cdot,\cdot;\cdot)}(\sigma) = 1],$$

where the calls to P (with inputs a witness and a language member) are restricted to ones satisfying R_ρ, **and** $\mathsf{X}(\sigma, \tau, x, w; l)$ is defined to be $\mathsf{sfS}(\sigma, \tau, x, R_\rho(w, x); l)$ (i.e. witness is dropped, and membership bit β is just 1),

- *unbounded partial-simulation (relative) soundness:*

$$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); (\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda);\ (x, l, \pi) \leftarrow \mathcal{A}_2^{\mathsf{sfS}(\sigma,\tau,\cdot,\cdot;\cdot),\,\mathsf{pV}(\sigma,\eta,\cdot,\cdot;\cdot)}(\sigma) :$$
$$[\neg \exists w \text{ s.t. } R_\rho(w, x) = 1, \text{ and } \mathsf{pV}(\sigma, \eta, x, \pi; l) = 1] \approx 0.$$

- *one-time full-ZK:*

$$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); (\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda);\ (x, l, \beta, s) \leftarrow \mathcal{A}_3^{\mathsf{sfS}(\sigma,\tau,\cdot,\cdot;\cdot),\,\mathsf{pV}(\sigma,\eta,\cdot,\cdot;\cdot)}(\sigma);$$
$$\pi \leftarrow \mathsf{sfS}(\sigma, \tau, x, \beta; l) : \mathcal{A}_4^{\mathsf{sfS}(\sigma,\tau,\cdot,\cdot;\cdot),\,\mathsf{pV}(\sigma,\eta,\cdot,\cdot;\cdot)}(\pi, s) = 1]\ \approx$$
$$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); \rho \leftarrow \mathcal{D}_\lambda;\ (\sigma, \tau, \tau_1, \eta)\ \leftarrow\ \mathsf{otfK}_1(\lambda, \rho);\ (x, l, \beta, s) \leftarrow \mathcal{A}_3^{\mathsf{sfS}(\sigma,\tau,\cdot,\cdot;\cdot),\,\mathsf{sfV}(\sigma,\eta,\cdot,\cdot;\cdot)}(\sigma);$$
$$\pi \leftarrow \mathsf{otfS}(\sigma, \tau_1, x; l) : \mathcal{A}_4^{\mathsf{sfS}(\sigma,\tau,\cdot,\cdot;\cdot),\,\mathsf{sfV}(\sigma,\eta,\cdot,\cdot;\cdot)}(\pi, s) = 1],$$

where β is a correct L_ρ-membership bit for x, **and** all calls to sfS also have correct L_ρ-membership bits **and** $\langle x, \pi; l \rangle$ is not queried to sfV/pV. Here s is a state variable.

Remark 1. Note that there is *no restriction* in the unbounded partial-simulation soundness definition of x, l (along with a β) being called to the first oracle sfS and returning π as proof.

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 provers (i.e. sfS and otfS) in the one-time full-ZK definition are only restricted to having the correct membership bit β. In particular, one can go back and forth with non-members.

Remark 3. We now show that sfS generated proofs on potential language members (whether members or not, i.e. β being 1 or 0) are accepted by real-world verifier V (with semi-functional CRS σ). Of course, the private verifier pV will *even* reject proofs generated on non-language members by sfS. This justifies the name "semi-functional simulator". To be precise, for distributions $\mathcal{E}$ of fake language members (i.e. $\beta = 0$) such that with high probability no efficient adversary can distinguish them from real language members (i.e. $\beta = 1$), the semi-functional proofs generated on such tuples are accepted by V. The proof is standard and uses both the partial-ZK and the one-time full ZK property, and will be given in the full version of the paper.

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 $\mathsf{P}(\sigma, \cdot, \cdot; \cdot)$ (i.e. P instantiated with CRS 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.

2 Dual-System Simulation-Sound QA-NIZK for DH Language

We will consider bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, with an efficiently computable pairing e from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$. Each group will be assumed to be cyclic with prime order q. For the construction to be secure, we will also assume that the external Diffie-Hellman (XDH) assumption holds for group $\mathbb{G}_1$. Essentially, it says that the DDH assumption holds in group $\mathbb{G}_1$. Later, we also give a construction based on the decisional linear (DLIN) assumption.

We consider the following class of languages parameterized by $\mathbf{g}_2, \mathbf{g}_2^a$ ($\in \mathbb{G}_2^2$). The **Diffie-Hellman language** corresponding to one such parameter $\rho = \langle \mathbf{g}_2, \mathbf{g}_2^a \rangle$ is $L_\rho = \{\langle \mathbf{g}_2^r, \mathbf{g}_2^{a \cdot r} \rangle \,|\, r \in \mathbb{Z}_q\}$.

Note that the witness relation R_ρ corresponding to this language is defined by $R_\rho(r, \langle \mathbf{x}_1, \mathbf{x}_2 \rangle)$ being true iff $\mathbf{x}_1 = \mathbf{g}_2^r$ and $\mathbf{x}_2 = (\mathbf{g}_2^a)^r$.

The distribution $\mathcal{D}$ under which the DSS-QA-NIZK is obtained is defined by picking $\mathbf{g}_2$ as a generator for $\mathbb{G}_2$ according to a bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ generation algorithm for which DDH assumption holds for $\mathbb{G}_1$, and then choosing a randomly from $\mathbb{Z}_q$.

We now define the various components of the DSS-QA-NIZK system for the above class of Diffie-Hellman languages under distribution $\mathcal{D}$. In all the algorithms below l will denote label, the additional input.

- The real-world algorithms:

  - The algorithm K_0 is just the group generation algorithm (it takes a unary string 1^m as input), and it also generates a collision-resistant hash function $\mathcal{H}$. The CRS generation algorithm K_1 takes language parameter $\rho = \langle \mathbf{g}_2, \mathbf{g}_2^a \rangle$ and other bilinear group parameters as input and generates the CRS as follows: it picks $c, b, d, e, \Delta_1', \Delta_2'$ randomly and independently from $\mathbb{Z}_q$ and sets the CRS to be $(\mathbf{CRS}_p, \mathbf{CRS}_v, \mathcal{H})$:
    $\mathbf{CRS}_p = \{\rho, \ \mathbf{d} = \mathbf{g}_2^d, \ \mathbf{e} = \mathbf{g}_2^e, \ \mathbf{w}_1 = \mathbf{g}_2^{(\Delta_1'+d)/b} \cdot (\mathbf{g}_2^a)^{c/b}, \ \mathbf{w}_2 = \mathbf{g}_2^{(\Delta_2'+e)/b}\}$,
    $\mathbf{CRS}_v = \{\mathbf{g}_1, \ \mathbf{c} = \mathbf{g}_1^c, \ \mathbf{b} = \mathbf{g}_1^b, \ \mathbf{v}_1 = \mathbf{g}_1^{-\Delta_1'}, \ \mathbf{v}_2 = \mathbf{g}_1^{-\Delta_2'}\}$.
  - The **prover** P takes as input $\mathbf{CRS}_p$, a language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and its witness r and produces a proof π consisting of two group $\mathbb{G}_2$ elements T and W as follows: Compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$. Next, compute $T = (\mathbf{d} \cdot \mathbf{e}^\iota)^r$, and $W = (\mathbf{w}_1 \cdot \mathbf{w}_2^\iota)^r$.
  - The **verifier** V takes as input $\mathbf{CRS}_v$, a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$, and a proof $\pi = (T, W)$, computes $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$, and outputs true iff
    $$e((\mathbf{v}_1 \mathbf{v}_2^\iota), \mathbf{x}_1^{-1}) \cdot e(\mathbf{c}, \mathbf{x}_2) \cdot e(\mathbf{g}_1, T) \ = \ e(\mathbf{b}, W).$$

- The partial-simulation world PPT components:

  - The **semi-functional CRS simulator** sfK_1 takes group parameters and $\mathcal{H}$ as input and produces a witness relation parameter $\rho = \langle \mathbf{g}_2, \mathbf{g}_2^a \rangle$ by picking a at random from $\mathbb{Z}_q$. It produces the semi-functional CRS $\sigma = (\mathbf{CRS}_p, \mathbf{CRS}_v, \mathcal{H})$ as follows:
    it further picks $c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'$ randomly and independently from $\mathbb{Z}_q$, and computes $d = d_1 + a \cdot d_2$, $e = e_1 + a \cdot e_2$, $\Delta_1 = (\Delta_1' + d + a \cdot c)/b$, $\Delta_2 = (\Delta_2' + e)/b$, and sets the CRS to be $(\sigma_p, \sigma_v, \mathcal{H})$:
    $\sigma_p = \{\rho, \ \mathbf{d} = \mathbf{g}_2^d, \ \mathbf{e} = \mathbf{g}_2^e, \ \mathbf{w}_1 = \mathbf{g}_2^{\Delta_1}, \ \mathbf{w}_2 = \mathbf{g}_2^{\Delta_2}\}$,
    $\sigma_v = \{\mathbf{g}_1, \ \mathbf{c} = \mathbf{g}_1^c, \ \mathbf{b} = \mathbf{g}_1^b, \ \mathbf{v}_1 = \mathbf{g}_1^{-\Delta_1'}, \ \mathbf{v}_2 = \mathbf{g}_1^{-\Delta_2'}\}$.
    It outputs $c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'$ as proof simulator trapdoors τ, and outputs a, c, b, d, e, Δ_1, Δ_2 as private-verifier trapdoors η.
  - The **semi-functional simulator** sfS uses trapdoors $\{c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'\}$ to produce a (partially-simulated) proof for a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and a binary bit β as follows: compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$; if $\beta = 1$, compute and output
    $$T = \mathbf{x}_1^{d_1+\iota e_1} \cdot \mathbf{x}_2^{d_2+\iota e_2} \ , \ W = \mathbf{x}_1^{(\Delta_1'+d_1+\iota\cdot(\Delta_2'+e_1))/b} \cdot \mathbf{x}_2^{(d_2+\iota e_2+c)/b},$$
    else choose y at random from $\mathbb{Z}_q$; Compute and output
    $$T = \mathbf{x}_1^{d_1+\iota e_1} \cdot \mathbf{g}_2^y \ , \ W = \mathbf{x}_1^{(\Delta_1'+d_1+\iota\cdot(\Delta_2'+e_1))/b} \cdot \mathbf{x}_2^{c/b} \cdot \mathbf{g}_2^{y/b}.$$
    This proof is partially simulated as it uses the bit β.

- The **private verifier** pV uses trapdoors $\{a, c, b, d, e, \Delta_1, \Delta_2\}$ to check a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and a proof T, W as follows: it outputs 1 iff $\mathbf{x}_2 = \mathbf{x}_1^a$ and $T = \mathbf{x}_1^{d + \iota e}$ and $W = \mathbf{x}_1^{\Delta_1 + \iota \Delta_2}$, where $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$.

- Finally, the one-time full simulation PPT components:

  - The **one-time full-simulation CRS generator** otfK$_1$ takes as input the bilinear group parameters, $\mathcal{H}$, and the language parameter $\rho = \langle \mathbf{g}_2, \mathbf{g}_2^a \rangle$ to produce the CRS as in the real-world, but with a little twist: it picks $c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'$ randomly and independently from $\mathbb{Z}_q$ and sets the CRS to be $(\sigma_p, \sigma_v, \mathcal{H})$:
  $\sigma_p = \{\rho,\ \mathbf{d} = \mathbf{g}_2^{d_1} \cdot (\mathbf{g}_2^a)^{d_2},\ \mathbf{e} = \mathbf{g}_2^{e_1} \cdot (\mathbf{g}_2^a)^{e_2},\ \mathbf{w}_1 = \mathbf{g}_2^{(\Delta_1' + d_1)/b} \cdot (\mathbf{g}_2^a)^{(d_2 + c)/b},\ \mathbf{w}_2 = \mathbf{g}_2^{(\Delta_2' + e_1)/b} \cdot (\mathbf{g}_2^a)^{e_2/b}\}$,
  $\sigma_v = \{\mathbf{g}_1,\ \mathbf{c} = \mathbf{g}_1^c,\ \mathbf{b} = \mathbf{g}_1^b,\ \mathbf{v}_1 = \mathbf{g}_1^{-\Delta_1'},\ \mathbf{v}_2 = \mathbf{g}_1^{-\Delta_2'}\}$.
  It also outputs $c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'$ as all three trapdoors τ, τ_1 and η.

  - The **one-time full simulator** otfS takes as input the trapdoors $\{c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'\}$ and a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ to produce a proof as follows: $T = \mathbf{x}_1^{d_1 + \iota e_1} \cdot \mathbf{x}_2^{d_2 + \iota e_2}$, $W = \mathbf{x}_1^{(\Delta_1' + \iota \Delta_2' + d_1 + \iota e_1)/b} \cdot \mathbf{x}_2^{(d_2 + \iota e_2 + c)/b}$, where $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$.

  - The **semi-functional verifier** sfV uses trapdoors $\{c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'\}$ to verify a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and a proof T, W as follows: compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, l)$ and output 1 iff

$$(\mathbf{x}_1^{\Delta_1' + \iota \Delta_2'} \cdot \mathbf{x}_2^c \cdot T = W^b) \ \textbf{and} \ (T = \mathbf{x}_1^{(d_1 + \iota e_1)} \cdot \mathbf{x}_2^{(d_2 + \iota e_2)}),$$

Theorem 1 *The above algorithms constitute a computationally-sound, complete and dual-system simulation sound quasi-adaptive NIZK proof system for the Diffie-Hellman language with language parameters sampled according to $\mathcal{D}$ above, given any bilinear-group generation algorithm for which the DDH assumption holds for groups $\mathbb{G}_1$. Let ADV_{XDH} denote any adversary's advantage in breaking the DDH assumption in group $\mathbb{G}_1$. Then, the probability of Adversary's success in the soundness experiment is at most $2 \cdot \text{ADV}_{XDH} + O(1/q)$. Further, unbounded partial-simulation soundness is perfect. The advantage of adversary $\mathcal{A}_1$ in the partial-ZK property experiment is at most $2 \cdot N \cdot \text{ADV}_{XDH} + O(N/q)$, where N is the number of $\mathcal{A}_1$'s oracle calls to the second oracle. Finally, $\mathcal{A}_4$'s advantage in the one-time full-ZK property experiment is at most $O((N + M)/q)$, where M and N are the number of calls of $\mathcal{A}_3$ and $\mathcal{A}_4$ combined to the first and second oracle resp.*

The theorem is proved in the next section.

3 Proof of DSS-QA-NIZK for DH Language

In this section we prove the theorem stated in the previous section about the security of the DSS-QA-NIZK construction for the Diffie-Hellman Language. After the proof, in the next sub-section, we show how the proof extends to the situation where the partial-world simulation key is revealed adaptively to the Adversary. Finally, we also show how to extend the above construction so that it is secure in a CCA1-security sense even after the simulation key is revealed.

Proof: [Theorem 1]
Completeness: Completeness of the real-world prover follows by simple inspection.

Soundness: We first define an extended Diffie-Hellman tag-based language

$$\overline{L}_\rho = \{\mathbf{x}_1, \mathbf{x}_2, T, \iota \mid \exists r : \mathbf{x}_1 = \mathbf{g}_2^r, \mathbf{x}_2 = \mathbf{a}^r, T = (\mathbf{d}\mathbf{e}^\iota)^r\}. \tag{1}$$

We will show that the element W is the single element proof for the quasi-adaptive NIZK for $\overline{L}_\rho$ [JR14]. For this extended language d and e are additional language parameters. Hence, from the soundness of this QA-NIZK [JR14] follows the soundness of the system in consideration. But to be self-contained, we provide a proof of soundness for the language $\overline{L}_\rho$ here, with the adversary getting the full CRS generated by K_1.

We will consider several experiments between a challenger $\mathcal{C}$ and adversary $\mathcal{A}$ here, starting with $\mathbf{H}_0$ which is same as the experiment of the soundness definition except that the challenger also generates ρ according to $\mathcal{D}$. The challenger $\mathcal{C}$ runs the verifier V on $\mathcal{A}$'s input $\mathbf{x}_1, \mathbf{x}_2, T$ and proof W and outputs the result of the verifier.

In $\mathbf{H}_1$, the challenger $\mathcal{C}$ generates the CRS differently by first picking Δ_1 and Δ_2 at random, and then setting $\Delta_1' = \Delta_1 \cdot b - d - c \cdot a$, and $\Delta_2' = \Delta_2 \cdot b - e$. It is easy to see that the CRS generated in $\mathbf{H}_1$ is identically distributed to that generated in $\mathbf{H}_0$. Further, instead of running the verifier V on $\mathcal{A}$'s input, the challenger runs the following test: it first picks s at random and then outputs:

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

Since probability of s being zero is only $1/q$, and further that $\mathbb{G}_1$ is a group, the probability of $\mathcal{C}$ outputting 1 in $\mathbf{H}_1$ is at most the probability of $\mathcal{C}$ outputting 1 in $\mathbf{H}_0$ (plus $1/q$).

In $\mathbf{H}_2$, the $\mathcal{C}$ also picks a random and independent s', and first computes a $\xi = e(\mathbf{g}_1^{s'}, T \cdot \mathbf{x}_2^c \cdot \mathbf{x}_1^{-(d+\iota e+a\cdot c)})$, and then outputs as follows:

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

The probability of $\mathcal{C}$ outputting 1 in $\mathbf{H}_2$ differs from it outputting 1 in $\mathbf{H}_1$ by at most the probability of an adversary's advantage in the DDH game. This follows by using DDH assumption on tuples $\mathbf{g}_1, \mathbf{g}_1^b, \mathbf{g}_1^{b\cdot s}$ and either $\mathbf{g}_1^s$ (real DDH challenge) or $\mathbf{g}_1^{s+s'}$ (fake DDH challenge). The important thing to note is that b is not used at all in simulating group $\mathbb{G}_2$ elements in the CRS (as opposed to $\mathbf{H}_0$). When the DDH challenge is real, this sets up $\mathbf{H}_1$ (i.e. $\mathcal{C}$ and the code of PPT $\mathcal{A}$), and when the DDH challenge is fake this sets up $\mathbf{H}_2$.

In $\mathbf{H}_3$, $\mathcal{C}$ also picks a random and independent c' and computes ξ as follows: $\xi = e(\mathbf{g}_1^{c'}, \mathbf{x}_2 \cdot \mathbf{x}_1^{-a}) \cdot e(\mathbf{g}_1^{s'}, T \cdot \mathbf{x}_1^{-(d+\iota e)})$.

Rest of the computation remains same. Again, the probability of $\mathcal{C}$ outputting 1 in $\mathbf{H}_3$ differs from it outputting 1 in $\mathbf{H}_2$ by at most the probability of an adversary's advantage in the DDH game. This follows by using DDH assumption on tuples $\mathbf{g}_1, \mathbf{g}_1^{s'}, \mathbf{g}_1^c$ and either $\mathbf{g}_1^{s'c}$ (real DDH challenge) or $\mathbf{g}_1^{c'}$ (fake DDH challenge). The important thing to note is that c is not used at all in simulating group $\mathbb{G}_2$ elements in the CRS (as opposed to $\mathbf{H}_0$). When the DDH challenge is real, this sets up $\mathbf{H}_2$ (i.e. $\mathcal{C}$ and the code of PPT $\mathcal{A}$), and when the DDH challenge is fake this sets up $\mathbf{H}_3$.

Now, a simple information theoretic argument shows that the probability of $\mathcal{C}$ outputting 1 in $\mathbf{H}_3$ when $\mathbf{x}_2 \cdot \mathbf{x}_1^{-a} \neq \mathbf{1}_1$ or $T \cdot \mathbf{x}_1^{-(d+\iota e)} \neq \mathbf{1}_1$ is negligible. That completes the proof of soundness.
Partial-ZK: We first note that the semi-functional CRS generator does a statistically indistinguishable simulation of the real-world CRS generator. This is seen by noting that the former generator generates ρ identically to $\mathcal{D}$, and that b is non-zero with high probability.

Now we show, by induction over the order of calls to the two oracles, that $\mathcal{A}_1$ cannot computationally distinguish between oracle responses in the real-world and the partial-simulation world. First, looking at oracle calls to the first oracle (i.e. P or sfS), we first note that if the call in the real-world to P is with language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and a valid witness r, then since this property is polynomial time verifiable given language parameter ρ, by induction, the adversary $\mathcal{A}_1$ in the partial simulation world also generates a $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ which belongs to L_ρ (with probability close to 1). Next, note that when $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ belongs to L_ρ (i.e. $\mathbf{x}_2 = \mathbf{x}_1^a$), the proof simulation is perfect; the membership bit β is 1, and hence $y = 0$, which leads to identical distribution for proof T, W in the two worlds.

We next show that for each call to the second oracle (i.e. V or pV), the difference in the probability of the oracle returning 1 in the two worlds is negligible. Since the CRS generated in the two worlds has statistically indistinguishable distribution, the probability of V (using real-world CRS) returning 1 in the real-world is non-negligibly close to the probability of the same V (with partial-simulation world CRS) returning 1. Thus, we just need to bound the difference in the probability of V and pV returning 1 in the partial-simulation world.

Consider the language $\overline{L}_\rho$ defined above. Now, from definition of pV it is clear that it accepts iff $\mathbf{x}_1, \mathbf{x}_2, T, \iota$ are in $\overline{L}_\rho$, and W is the unique "proof" matching this tuple (i.e $W = (\mathbf{w}_1 \mathbf{w}_2^\iota)^r$). It is also clear that V accepts the triplet and W if the triplet is in $\overline{L}_\rho$ and W is this unique "proof". Further, it is clear that V rejects if the quadruplet is in $\overline{L}_\rho$ and W is not this unique proof. Thus, we just need to show that if the quadruplet is not in $\overline{L}_\rho$ then (regardless of W) V rejects with high probability. But, indeed we showed in the proof of soundness above that W is the (single group element) proof of the language $\overline{L}_\rho$ which is sound under the verifier V and with adversary getting the full CRS generated by K_1 (and hence also the full CRS generated by sfK_1).

Unbounded partial-simulation (relative) soundness: This follows trivially, since the private verifier has access to a (see the private verifier trapdoor), and directly checks if the potential language member submitted is indeed a language member.

One-time full-ZK: We will show that the one-time full-ZK property holds statistically, i.e. against all powerful adversaries. Now we define a sequence of experiments, and show that they are statistically indistinguishable.

First note that the distribution of the CRS generated by the one-time full-simulation CRS generator is statistically indistinguishable from the distribution of the CRS generated by the semi-functional CRS generator. We next show, by induction over the order of calls of $\mathcal{A}_3$ and $\mathcal{A}_4$ to the two oracles and the one-time generation of proof π, that $\mathcal{A}_4$ cannot distinguish between the responses of the oracle calls and π in the partial-simulation world and this one-time full simulation world.

To this end, we will define a sequence of hybrid experiments and show that the view of $\mathcal{A}_4$ in every two consecutive experiments is computationally indistinguishable.

The first experiment $\mathbf{H}_0$ is identical to the partial-simulation world, but we describe it in detail here. The challenger $\mathcal{C}$ interacts with two adversaries $\mathcal{A}_3$ and $\mathcal{A}_4$ each with two different types of oracle calls. To start with, $\mathcal{C}$ produces a CRS to be used by both the adversaries. $\mathcal{A}_3$ also produces a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$, and a membership bit β. The challenger then produces π and $\mathcal{A}_4$ takes this as additional input, and finally outputs a bit.

$\mathcal{C}$ starts by picking a at random and sets $\rho = \langle \mathbf{g}_2, \mathbf{g}_2^a \rangle$. Next, it picks $c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'$ randomly and independently from $\mathbb{Z}_q$, and computes $d = d_1 + a \cdot d_2$, $e = e_1 + a \cdot e_2$, $\Delta_1 = (\Delta_1' + d + a \cdot c)/b$, $\Delta_2 = (\Delta_2' + e)/b$, and sets the CRS to be $(\sigma_p, \sigma_v, \mathcal{H})$:

$\sigma_p = \{\rho, \mathbf{d} = \mathbf{g}_2^d, \mathbf{e} = \mathbf{g}_2^e, \mathbf{w}_1 = \mathbf{g}_2^{\Delta_1}, \mathbf{w}_2 = \mathbf{g}_2^{\Delta_2}\}$,

$\sigma_v = \{\mathbf{g}_1, \mathbf{c} = \mathbf{g}_1^c, \mathbf{b} = \mathbf{g}_1^b, \mathbf{v}_1 = \mathbf{g}_1^{-\Delta_1'}, \mathbf{v}_2 = \mathbf{g}_1^{-\Delta_2'}\}$.

It retains $c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'$ as proof simulator trapdoors τ, and retains $a, c, b, d, e, \Delta_1, \Delta_2$ as private-verifier trapdoors η.

The challenger $\mathcal{C}$ serves all calls to the first oracle using the algorithm sfS described above on trapdoors τ. It also produces π using the same algorithm sfS on $\mathcal{A}_3$ supplied input. It serves all calls to the second oracle using private verifier pV described above using trapdoors η.

In experiment $\mathbf{H}_1$, the challenger $\mathcal{C}$, instead of picking d_1, d_2 at random (and similarly e_2, e_2 at random), picks d and d_2 randomly and independently and sets $d_1 = d - a \cdot d_2$ (and $e_1 = e - a \cdot e_2$). Rest of the CRS computation remains the same, and it retains a, c, b, d, d_2, e, e_2, Δ_1', Δ_2', Δ_1, Δ_2 as trapdoors (for both oracles). It further replaces sfS by the following: on input a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and a binary bit β as follows: compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$; if $\beta = 1$ set $y = 0$, else choose y at random from $\mathbb{Z}_q$; Compute and output

$$T = \mathbf{x}_1^{d+\iota e} \cdot \mathbf{g}_2^y, \ W = \mathbf{x}_1^{(\Delta_1' + d + \iota \cdot (\Delta_2' + e))/b} \cdot \mathbf{x}_2^{c/b} \cdot \mathbf{g}_2^{y/b}. \tag{2}$$

We show that when the membership bit β is valid for $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$, the view of the adversaries in Expt_0 and Expt_1 is identical. First note that the CRS is identically distributed in the two experiments. Next, let $\mathbf{x}_2 = \mathbf{x}_1^a \cdot z$. Thus, if $\beta = 1$ then $z = \mathbf{1}_2$, otherwise z is not one. When $\beta = 1$, $\mathcal{C}$ chooses $y = 0$ in $\mathbf{H}_1$ above, and hence this is identical to behavior of sfS. Otherwise, y is chosen randomly in both sfS and $\mathbf{H}_1$, and now note that

$$\mathbf{x}_1^{d+\iota e} \cdot \mathbf{g}_2^y = \mathbf{x}_1^{d_1 + \iota e_1} \cdot (\mathbf{x}_2/z)^{d_2 + \iota e_2} \cdot \mathbf{g}_2^y,$$

$$\mathbf{x}_1^{(\Delta_1' + d + \iota \cdot (\Delta_2' + e))/b} \cdot \mathbf{x}_2^{c/b} \cdot \mathbf{g}_2^{y/b} = \mathbf{x}_1^{(\Delta_1' + d_1 + \iota \cdot (\Delta_2' + e_1))/b} \cdot \mathbf{x}_2^{c/b} \cdot (\mathbf{x}_2/z)^{(d_2 + \iota e_2)/b} \cdot \mathbf{g}_2^{y/b}.$$

Since, y is chosen randomly and independently the experiments $\mathbf{H}_0$ and $\mathbf{H}_1$ produce identical distributions.

In the next experiment $\mathbf{H}_2$, the challenger simulates otfS by producing the proof π on the potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and binary bit β produced by $\mathcal{A}_3$ as follows (all calls to the first oracle remain as in previous experiment): it ignores the bit β, computes $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$, and next computes and outputs

$$T = \mathbf{x}_1^{d+\iota e - a \cdot (d_2 + \iota e_2)} \cdot \mathbf{x}_2^{d_2 + \iota e_2}, \ W = \mathbf{x}_1^{(\Delta_1' + d + \iota \cdot (\Delta_2' + e) - a \cdot (d_2 + \iota e_2))/b} \cdot \mathbf{x}_2^{(d_2 + \iota e_2 + c)/b}. \tag{3}$$

Again, let $\mathbf{x}_2 = \mathbf{x}_1^a \cdot z$, and note the T and W being computed in this experiment can be written as

$$T = \mathbf{x}_1^{d+\iota e} \cdot z^{d_2 + \iota e_2}, \ W = \mathbf{x}_1^{(\Delta_1' + d + \iota \cdot (\Delta_2' + e))/b} \cdot \mathbf{x}_2^{c/b} \cdot z^{(d_2 + \iota e_2)/b}.$$

Next, note that d_2 was chosen randomly and independently of d. Further, the CRS computation, all first oracle response computations, and all second oracle response (see pV) computations only need d and e (and not d_2). Thus, since $z = \mathbf{1}_2$ iff $\beta = 1$ (given that β is a valid membership bit), the distribution of π is identical in $\mathbf{H}_1$ and $\mathbf{H}_2$.

In the next experiment $\mathbf{H}_3$, the private-verifier pV is replaced by the following code: output 1 iff

$$T = \mathbf{x}_1^{d+\iota e - a \cdot (d_2 + \iota e_2)} \cdot \mathbf{x}_2^{(d_2 + \iota e_2)} \quad \textbf{and} \quad \mathbf{x}_1^{\Delta_1' + \iota \Delta_2'} \cdot \mathbf{x}_2^c \cdot T = W^b, \tag{4}$$

where $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$.

12

In order to show that the view of the adversary is indistinguishable in experiments $\mathbf{H}_3$ and $\mathbf{H}_2$, we define several hybrid experiments $\mathbf{H}_{2,i}$ (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 $\mathbf{H}_{2,0}$ is identical to $\mathbf{H}_2$, 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 first second-oracle call. The last hybrid experiment $\mathbf{H}_{2,N}$ will then be same as $\mathbf{H}_3$. The second-oracle call response in experiment $\mathbf{H}_{2,i+1}$ differs only in the $(N-i)$-th second-oracle call response in $\mathbf{H}_{2,i}$. In the latter experiment, this call is still served as in $\mathbf{H}_2$ (i.e. using pV). In the former experiment $\mathbf{H}_{2,i+1}$, the $(N-i)$-th call is responded to as defined in $\mathbf{H}_3$ above.

To show that the view of the adversary is statistically indistinguishable in $\mathbf{H}_{2,i}$ and $\mathbf{H}_{2,i+1}$, 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, this view is independent of d_2 and e_2 *except* for the information $d_2 + \iota' e_2$, where $\iota' = \mathcal{H}(\mathbf{x}_1', \mathbf{x}_2')$, and $\langle \mathbf{x}_1', \mathbf{x}_2' \rangle$ is the potential language member on which $\mathcal{A}_3$ sought a proof π (if the $(N-i)$-th second-oracle call happens before $\mathcal{A}_3$ outputs this request, then even this information $d_2 + \iota' e_2$ is not available to the adversary). Now, consider the case that the adversary does not query this $\langle \mathbf{x}_1', \mathbf{x}_2' \rangle$ and π to the second-oracle (and hence to this $(N-i)$-th call).

First, suppose the $(N-i)$-th call has the same $\langle \mathbf{x}_1', \mathbf{x}_2' \rangle$, but a different proof π' then the proof π output by the challenger $\mathcal{C}$ (see definition of experiment $\mathbf{H}_2$ above). Then, we show that both experiments $\mathbf{H}_{2,i}$ and $\mathbf{H}_{2,i+1}$ reject this call (i.e. output 0). This follows from uniqueness of the valid proof, but to be rigorous, note that the proof π produced in both experiments is

$$T = (\mathbf{x}_1')^{d+\iota' e - a \cdot (d_2 + \iota' e_2)} \cdot (\mathbf{x}_2')^{d_2 + \iota' e_2}, \; W = (\mathbf{x}_1')^{(\Delta_1' + d + \iota' \cdot (\Delta_2' + e) - a \cdot (d_2 + \iota' e_2))/b} \cdot (\mathbf{x}_2')^{(d_2 + \iota' e_2 + c)/b},$$

where $\iota' = \mathcal{H}(\mathbf{x}_1', \mathbf{x}_2')$. Since, the $(N-i)$-th call has the same $\langle \mathbf{x}_1', \mathbf{x}_2' \rangle$, and hence the same hash $\mathcal{H}$-value, it is easy to inspect that pV accepts the above proof π, and in particular $T = (\mathbf{x}_1')^{d+\iota' e}$ and $W = (\mathbf{x}_1')^{\Delta_1 + \iota' \Delta_2}$. Thus, any change in this T or W will be rejected by pV (and hence experiment $\mathbf{H}_{2,i}$). It is also easy to inspect that the response to $(N-i)$-th call in $\mathbf{H}_{2,i+1}$ with input $\langle \mathbf{x}_1', \mathbf{x}_2' \rangle$ and above proof π will be 1, i.e.

$$T \; = \; (\mathbf{x}_1')^{d+\iota' e - a \cdot (d_2 + \iota' e_2)} \cdot (\mathbf{x}_2')^{(d_2 + \iota' e_2)} \textbf{ and } (\mathbf{x}_1')^{\Delta_1' + \iota' \Delta_2'} \cdot (\mathbf{x}_2')^c \cdot T \; = \; W^b,$$

Thus, any change in T will fail the first conjunct, and any change in just W (and not T) will fail the second conjunct, except for the negligible probability that $b = 0$. Thus, experiment $\mathbf{H}_{2,i+1}$ also outputs 0 on its $(N-i)$-th call.

Now, suppose that $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ in the $(N-i)$-th second-oracle call is different from $\langle \mathbf{x}_1', \mathbf{x}_2' \rangle$ on which the adversary sought proof π. Since $\mathcal{H}$ is a collision resistant hash function, no efficient adversary can choose these inputs to $\mathcal{H}$ such that their hash is the same. So, assume that $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$ is different from $\iota' = \mathcal{H}(\mathbf{x}_1', \mathbf{x}_2')$.

Thus, $d_2 + \iota e_2$ is random and independent given $d_2 + \iota' e_2$ (recall, d_2, e_2 are chosen randomly and independently of d and e from $\mathbf{H}_1$ on-wards). Now it is clear that if pV accepts $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and the submitted proof T, W, then experiment $\mathbf{H}_{2,i+1}$ also accepts the same. So, we next show that if $\mathbf{H}_{2,i+1}$ accepts the $(N-i)$-th call, then pV also accepts. Thus,

$$T \; = \; \mathbf{x}_1^{d+\iota e - a \cdot (d_2 + \iota e_2)} \cdot \mathbf{x}_2^{(d_2 + \iota e_2)} \textbf{ and } \mathbf{x}_1^{\Delta_1' + \iota \Delta_2'} \cdot \mathbf{x}_2^c \cdot T \; = \; W^b \tag{5}$$

holds, and suppose pV rejects, i.e. one of the three conjuncts $\mathbf{x}_2 = \mathbf{x}_1^a$ and $T = \mathbf{x}_1^{d+\iota e}$ and $W = \mathbf{x}_1^{\Delta_1 + \iota \Delta_2}$ fails. First, note that if the first two conjuncts hold and equation (5) holds, then the third

conjunct (involving W) holds. Also, note that if the first conjunct holds and equation (5) holds, then the second conjunct (involving T) holds. So, suppose the first conjunct does not hold, i.e. $\mathbf{x}_2 \neq \mathbf{x}_1^a$. Now, from equation 5, we get

$$d_2 + \iota e_2 = \frac{\log_{\mathbf{g}_2}(T/\mathbf{x}_1^{d+\iota e})}{\log_{\mathbf{g}_2}(\mathbf{x}_2/\mathbf{x}_1^a)} \tag{6}$$

But since $d_2 + \iota e_2$ is completely independent of the adversary's view till the $(N-i)$-th call and d, e, a, and hence the right-hand-side of equation (6), the probability of equation (6) holding is at most $1/q$. that completes the proof of computational indistinguishability of $\mathbf{H}_{2.i}$ and $\mathbf{H}_{2,i+1}$, and hence by induction of also $\mathbf{H}_2$ and $\mathbf{H}_3$. Note that computational restriction was only required for the collision-resistance property.

In then next experiment $\mathbf{H}_4$, the challenger $\mathcal{C}$ changes back the response to the first-oracle to as in sfS (from (9) in $\mathbf{H}_1$), i.e.: compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$; if $\beta = 1$ set $y = 0$, else choose y at random from $\mathbb{Z}_q$; Compute and output

$$T = \mathbf{x}_1^{d_1+\iota e_1} \cdot \mathbf{x}_2^{d_2+\iota e_2} \cdot \mathbf{g}_2^y , \ W = \mathbf{x}_1^{(\Delta_1'+d_1+\iota\cdot(\Delta_2'+e_1))/b} \cdot \mathbf{x}_2^{(d_2+\iota e_2+c)/b} \cdot \mathbf{g}_2^{y/b}.$$

The argument that the view of the adversary in $\mathbf{H}_4$ is identical to the view of the adversary in $\mathbf{H}_3$ (when the membership bits β are valid) is same as the argument for $\mathbf{H}_1$ and $\mathbf{H}_0$ given earlier.

In the next experiment $\mathbf{H}_5$, the challenger $\mathcal{C}$ changes the response to the second-oracle (verifier) call (from equation (4) defined in $\mathbf{H}_3$) to: compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$ and output 1 iff

$$(T = \mathbf{x}_1^{(d_1+\iota e_1)} \cdot \mathbf{x}_2^{(d_2+\iota e_2)}) \ \textbf{and} \ (\mathbf{x}_1^{\Delta_1'+\iota\Delta_2'} \cdot \mathbf{x}_2^c \cdot T = W^b),$$

It is straightforward to see that Adversary's view is identical in $\mathbf{H}_4$ and $\mathbf{H}_5$ (i.e. by the way d_1 and e_1 are defined).

In the next experiment $\mathbf{H}_6$, the challenger samples ρ, and then computes the CRS and the trapdoors exactly as in otfK$_1$. It is straightforward to see that the CRS (and the trapdoors) generated in $\mathbf{H}_6$ is identically distributed to the CRS (and the trapdoors) generated in $\mathbf{H}_5$.

At this point, the view of the adversaries $\mathcal{A}_3$ and $\mathcal{A}_4$ is exactly as in the one-time full ZK world, and that completes the proof. $\qquad\square$

3.1   Revealing the Partial-Simulation Key

In some applications, the partial-simulation trapdoor τ can be revealed to the adversary, adaptively on adversary's demand. This event will be referred to as the *Reveal* event. In this case we would like to have the property that one-time full ZK simulation is still possible under the restriction that the adversary $\mathcal{A}_4$ has access to the second oracle (i.e. the verifier pV/sfV) only *till the reveal event.* The adversary $\mathcal{A}_3$ will continue to have access to the second oracle regardless of the reveal event.

To achieve this, we highlight an interesting property of the DSS-QA-NIZK of the Diffie-Hellman language from Section 2. We show that in the one-time full ZK property, there is an *intermediate* world, which uses a slightly different semi-functional CRS generator and simulator, but uses the private-verifier pV. Let's call these CRS generator and simulator sfK$_1'$ and sfS$'$ respectively, which

we define soon. Then, we have the following:

$$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); (\rho, \sigma, \tau, \eta) \leftarrow \mathsf{sfK}_1(\lambda); (x, \beta, s) \leftarrow \mathcal{A}_3^{\mathsf{sfS}(\sigma, \tau, \cdot, \cdot),\, \mathsf{pV}(\sigma, \eta, \cdot, \cdot)}(\sigma);$$

$$\pi \leftarrow \mathsf{sfS}(\sigma, \tau, x, \beta) : \mathcal{A}_4^{\mathsf{sfS}(\sigma, \tau, \cdot, \cdot),\, \mathsf{pV}(\sigma, \eta, \cdot, \cdot)}(\pi, s) = 1] \approx$$

$$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); (\rho, \sigma, \tau, \tau_1, \eta) \leftarrow \mathsf{sfK}_1'(\lambda); (x, \beta, s) \leftarrow \mathcal{A}_3^{\mathsf{sfS}'(\sigma, \tau, \cdot, \cdot),\, \mathsf{pV}(\sigma, \eta, \cdot, \cdot)}(\sigma);$$

$$\pi \leftarrow \mathsf{otfS}(\sigma, \tau_1, x) : \mathcal{A}_4^{\mathsf{sfS}'(\sigma, \tau, \cdot, \cdot),\, \mathsf{pV}(\sigma, \eta, \cdot, \cdot)}(\pi, s) = 1] \approx \tag{7}$$

$$\Pr[\lambda \leftarrow \mathsf{K}_0(1^m); \rho \leftarrow \mathcal{D}_\lambda; (\sigma, \tau, \tau_1, \eta) \leftarrow \mathsf{otfK}_1(\lambda, \rho); (x, \beta, s) \leftarrow \mathcal{A}_3^{\mathsf{sfS}(\sigma, \tau, \cdot, \cdot),\, \mathsf{sfV}(\sigma, \eta, \cdot, \cdot)}(\sigma);$$

$$\pi \leftarrow \mathsf{otfS}(\sigma, \tau_1, x) : \mathcal{A}_4^{\mathsf{sfS}(\sigma, \tau, \cdot, \cdot),\, \mathsf{sfV}(\sigma, \eta, \cdot, \cdot)}(\pi, s) = 1], \tag{8}$$

where β is a correct L_ρ-membership bit for x, **and** all calls to sfS also have correct L_ρ-membership bits **and** $\langle x, \pi \rangle$ is not queried to sfV/pV.

We now define sfK_1' and sfS'. These actually come from the intermediate games in the proof of one-time full-ZK in Section!3. In particular, this intermediate world is just experiment $\mathbf{H}_2$. So, in particular sfK_1' is the following algorithm: It is just like sfK_1 except that instead of picking d_1, d_2 at random (and similarly e_2, e_2 at random), it picks d and d_2 randomly and independently and sets $d_1 = d - a \cdot d_2$ (and $e_1 = e - a \cdot e_2$). Rest of the CRS computation remains the same. It outputs $c, b, d, e, , \Delta_1', \Delta_2'$ as trapdoors τ (to be used by sfS'), and outputs $c, b, d_1, d_2, e_1, e_2, \Delta_1', \Delta_2'$ as trapdoors τ_1 (to be used by otfS), and outputs $a, c, b, d, e, \Delta_1, \Delta_2$ as private-verifier trapdoors η.

The intermediate semi-functional simulator sfS' (using trapdoors $c, b, d, e, , \Delta_1', \Delta_2'$) on input a potential language member $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ and a binary bit β as follows: compute $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2)$; if $\beta = 1$ set $y = 0$, else choose y at random from $\mathbb{Z}_q$; Compute and output

$$T = \mathbf{x}_1^{d+\iota e} \cdot \mathbf{g}_2^{y}, \ W = \mathbf{x}_1^{(\Delta_1' + d + \iota \cdot (\Delta_2' + e))/b} \cdot \mathbf{x}_2^{c/b} \cdot \mathbf{g}_2^{y/b}. \tag{9}$$

The proof of the above property ((7) and (8) then follows from the proof of Section 3.

We now show that the above property ((7) and (8) holds even if τ is revealed to the adversary (adaptively), as long as the following holds: (a) the second oracle (verifier) is not called after τ is revealed to the adversary, and (b) the switch between the partial-simulation world and the intermediate world above is undertaken only if x is a (Diffie-Hellman) language member (i.e. $\beta = 1$), and (c) the first oracle ($\mathsf{sfS}/\mathsf{sfS}'$) is only called with language members (i.e. $\beta = 1$) in all cases above.

To show this we need to verify the (computational or statistical) indistinguishability of various experiments in the proof of one-time full-ZK, i.e. games $\mathbf{H}_0$ through $\mathbf{H}_6$.

Between experiment $\mathbf{H}_0$ and $\mathbf{H}_1$ the CRS computation is statistically indistinguishable even with τ revealed. When $\beta = 1$ (which is the restriction (b) and (c) combined above), the generation of T and W is identical in the two games, i.e. sfS and sfS' behave identically.

In the indistinguishability of experiments $\mathbf{H}_1$ and $\mathbf{H}_2$, again restricting to $\beta = 1$ (by (b)), we notice there is no difference in the way sfS generates T, W (experiment $\mathbf{H}_1$) and how otfS (experiment $\mathbf{H}_2$) generates T, W.

In the indistinguishability of $\mathbf{H}_2$ and $\mathbf{H}_3$ where we replace pV with sfV, we go though various hybrid games $\mathbf{H}_{2,i}$. Since by restriction (a) above, no calls to the second oracle are allowed after τ, i.e. d_2, e_2 among others, is revealed to the adversary, the information-theoretic argument in equation 6 still holds.

The indistinguishability of $\mathbf{H}_3$ and $\mathbf{H}_4$ follows by noting that we restrict to $\beta = 1$ (i.e. restriction (c)). Indistinguishability of $\mathbf{H}_4, \mathbf{H}_5$ and $\mathbf{H}_6$ is straightforward.

We will see that the above properties of the DSS-QA-NIZK for the Diffie-Hellman language allows for a *keyed-homomorphic CCA2-secure encryption* scheme with CPA security maintained if the homomorphic-evaluation key is revealed (Section 4). The above properties ensure that CPA security holds along with a decryption oracle till reveal event.

3.2 Key-Reveal and CCA1-style Security

We can also extend th keyed-homomorphic scheme of Section 4 to maintain CCA1-security when the key is revealed, but the construction requires use of another group element as proof that provides CCA1-style security. Since this proof element is actually malleable it does not require additional trapdoor keys. This additional group element however provides CCA1 style-soundness of the Diffie-Hellman tuple, and hence the usual trick of switching between alternate ways of decryption can be accomplished up to the encryption query event.

To this end, we define a DSS-QA-NIZK for a new class of parameterized languages.

We consider the following class of languages parameterized by $\mathbf{g}_2, \mathbf{g}_2^a, \mathbf{g}_2^{\epsilon_1} \cdot \mathbf{g}_2^{\epsilon_2 \cdot a}, \epsilon_1, \epsilon_2 \ (\in \mathbb{G}_2^4 \times \mathbb{Z}_q^2)$. The **augmented Diffie-Hellman language** corresponding to one such parameter ρ is

$$L_\rho = \{\langle \mathbf{g}_2^r, \mathbf{g}_2^{a \cdot r}, (\mathbf{g}_2^{\epsilon_1} \cdot \mathbf{g}_2^{\epsilon_2 \cdot a})^r \rangle \mid r \in \mathbb{Z}_q\}.$$

Note that the parameters *include* ϵ_1 and ϵ_2. In this respect, the hard component of the language (Diffie-Hellman) is due to a and that is not part of the parameters. The distribution $\mathcal{D}$ under which the DSS-QA-NIZK now picks ϵ_1 and ϵ_2 at random from $\mathbb{Z}_q$ as well.

We now describe the modifications required to the construction of Section 2. First, the component T of the proof will remain as before. The component W was just the QA-NIZK for the extended Diffie-Hellman language (see (1) in Section 3), and now it will be the single group element QA-NIZK [JR14] for the augmented and extended Diffie-Hellman language (and for this QA-NIZK the parameters ϵ_1, ϵ_2 are ignored by the CRS generator, and it just uses $\mathbf{g}_2, \mathbf{g}_2^a, \mathbf{g}_2\epsilon_1 \cdot \mathbf{g}_2^{\epsilon_2 \cdot a}$). More precisely, this *doubly-augmented Diffie-Hellman* language is the following tag-based language

$$\overline{XL}_\rho = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, T, \iota \mid \exists r : \mathbf{x}_1 = \mathbf{g}_2^r, \mathbf{x}_2 = \mathbf{a}^r, \mathbf{x}_3 = (\mathbf{g}_2^{\epsilon_1} \cdot \mathbf{g}_2^{\epsilon_2 \cdot a})^r, T = (\mathbf{de}^\iota)^r\}. \tag{10}$$

Thus, the real-world CRS for the DSS-QA-NIZK is the real-world CRS for the QA-NIZK for this doubly-augmented Diffie-Hellman language $\overline{XL}_\rho$ (with ϵ_1, ϵ_2 dropped) (see [JR14]). The real-world prover P computes $\iota = \mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, l)$, and $T = (\mathbf{d} \cdot \mathbf{e}^\iota)^r$, and computes W using the prover for the QA-NIZK for $\overline{XL}_\rho$. The real-world verifier V just follows the verifier for the QA-NIZK for $\overline{XL}_\rho$.

In the partial-simulation world, sfK_1 itself generates the language parameters. However, it generates $\mathbf{g}_2^{\epsilon_1} \cdot \mathbf{g}_2^{\epsilon_2 \cdot a}$ differently by just picking an ϵ at random from $\mathbb{Z}_q$ and outputting $\mathbf{g}_2^\epsilon$. The value ϵ is made an additional part of the verifier trapdoor η, and hence the private verifier pV can directly check for validity of the language members. Crucially, ϵ is not part of the proof simulation trapdoor τ.

In the one-time full-ZK world, the CRS generator otfK_1 will make ϵ_1 and ϵ_2 as part of the verifier trapdoor η. Recall ϵ_1 and ϵ_2 are part of the parameters of the augmented Diffie-Hellman language.

Now, focusing on the proof security of the above scheme, the proof is similar to the proof of theorem 1. More importantly, now we also get a CCA1-style security even if the proof simulation

trapdoor is revealed. This is proven similarly to the proof sketch given above in Section 3.1, but now with the restriction of not calling the second (verifier) oracle after reveal *only limited* to $\mathcal{A}_4$. More precisely, we now claim that properties ((7) and (8) hold even if τ is revealed to the adversary (adaptively), as long as the following (modified restrictions) holds: (a') the second oracle (verifier) is not called by $\mathcal{A}_4$ after τ is revealed to the adversary, and (b) the switch between the partial-simulation world and the intermediate world above is undertaken only if x is a (Diffie-Hellman) language member (i.e. $\beta = 1$), and (c) the first oracle ($\mathsf{sfS}/\mathsf{sfS'}$) is only called with language members (i.e. $\beta = 1$) in all cases above.

The proof is similar to the proof sketch above in Section 3.1 except with the following change. In the indistinguishability of $\mathbf{H}_2$ and $\mathbf{H}_3$ where we replace pV with sfV, we go though various hybrid games $\mathbf{H}_{2,i}$. Since by restriction (a') above, no calls to the second oracle are allowed after τ, i.e. d_2, e_2 among others, is revealed to the adversary, the information-theoretic argument in equation 6 still holds. A similar information-theoretic argument continues to hold for $\mathcal{A}_3$'s oracle calls using β_2 as trapdoor to prove that if $\mathbf{x}_2 \neq \mathbf{x}_1^a$ holds then

$$\beta_2 = \frac{\log_{\mathbf{g}_2}(\mathbf{x}_3/\mathbf{x}_1^{\beta})}{\log_{\mathbf{g}_2}(\mathbf{x}_2/\mathbf{x}_1^a)} \tag{11}$$

where β is chosen independently by challenger $\mathcal{C}$ in experiment $\mathbf{H}_2$ and $\mathbf{H}_3$, But, the probability of this holding is negligible, and that completes the proof sketch.

4   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. To an adversary not having access to the evaluation key, the homomorphic operation should not be possible 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 with better security guarantees.

Definition 2 (KH-PKE Scheme) *A KH-PKE scheme for a message space $\mathcal{M}$ with a binary operation $\odot$ defined from $\mathcal{M}^2 \to \mathcal{M}$ is a tuple of algorithms $(KeyGen, Enc, Dec, Eval)$:*

KeyGen: *This algorithm takes 1^λ as input, and returns a public key pk, a decryption key sk_d, and a homomorphic operation key sk_h.*

Enc: *This algorithm takes pk and a message $M \in \mathcal{M}$ as inputs and returns a ciphertext C.*

Dec: *This algorithm takes sk_d and a ciphertext C as input, and returns M or $\bot$.*

Eval: *This algorithm takes sk_h and two ciphertexts C_1 and C_2 as inputs, and returns a ciphertext C or $\bot$.*

The scheme is said to be correct if (i) for Enc we have $Dec(sk_d, C) = M$ and (ii) for Eval we have $Dec(sk_d, C) = Dec(sk_d, C_1) \odot Dec(sk_d, C_2)$.

Definition 3 (KH-CCA Security) *A KH-PKE scheme $(KeyGen, Enc, Dec, Eval)$ for a message space $\mathcal{M}$ with a binary operation $\odot$ defined from $\mathcal{M}^2 \to \mathcal{M}$ is said to be KH-CCA secure if no PPT adversary $\mathcal{A}$ has non-negligible advantage in winning the following game with a challenger:*

KeyGen: *The challenger takes 1^λ as input, and computes a public key pk, a decryption key sk_d, and a homomorphic operation key sk_h. It then gives pk to $\mathcal{A}$. It also initializes a list $D := \emptyset$. It also gives oracle access to $\mathcal{A}$ to the functions $Enc(.), Eval(sk_h, .), RevHK$ and $Dec(sk_d, .)$ defined as follows:*

Enc: *This query is performed once. On $\mathcal{A}$'s inputs $m_0, m_1 \in \mathcal{M}$, the challenger randomly samples a bit b and computes $C := Enc(pk, m_b)$ and returns C.*

Eval: *On receiving ciphertexts C_1, C_2 from $\mathcal{A}$, computes $C := Eval(sk_h, C_1, C_2)$ and returns C. In addition if $C_1 \in D$ or $C_2 \in D$, then sets $D := D \cup \{C\}$.*

RevHK: *Upon receiving this request, returns sk_h.*

Dec: *This oracle is not available after $\mathcal{A}$ has both requested $RevHK$ and obtained the challenge ciphertext C^* in any order. When available, it returns $\perp$ if $C \in D$ and $Dec(sk_d, C)$ on $\mathcal{A}$'s query C.*

The adversary $\mathcal{A}$ outputs a bit b' and wins if $b' = b$. Its advantage is defined to be $|\Pr[\mathcal{A} \ wins] - 1/2|$.

Construction. We present a construction of a KH-CCA secure encryption scheme which utilizes our DSS-QA-NIZK proofs. We start with the observation that a standard El Gamal encryption scheme $(\mathbf{g}^x, m \cdot \mathbf{f}^x)$ is multiplicatively homomorphic, but is not CCA secure due to the exact same reason. Extending it to be a CCA secure encryption scheme, as in Cramer-Shoup [CS02], destroys the homomorphic property. Our construction essentially takes an El Gamal scheme and attaches a DSS-QA-NIZK to it. Although the NIZK itself is not homomorphic, we can take advantage of the corresponding Semi-Functional Simulator sfS and simulate the proof of a multiplicatively generated El Gamal encryption while computing a homomorphic evaluation. We show that keeping the sfS trapdoor hidden from the adversary enables the scheme to be CCA secure, while also giving weaker semantic security if the sfS trapdoor is revealed. The semantic security continues to hold even with the availability of a decryption oracle till the time of reveal. Using the augmented Diffie Hellman language and its DSS-QA-NIZK as described in Section 3.2, we also get CCA1-security despite the key being revealed. Here we focus on the construction for the weaker notion where the decryption oracle is available till the reveal event.

KeyGen: Generate $\mathbf{g}$ and a, k randomly. Use sfK_1 to generate CRS σ and trapdoors τ and η for the language $(\mathbf{g}^w, \mathbf{g}^{aw})$. Set $pk = (\mathbf{g}, \mathbf{g}^a, \mathbf{g}^k, \sigma), \quad sk_h := \tau, \quad sk_d := k$.

Enc: Given plaintext m, generate $w \xleftarrow{\$} \mathbb{Z}_q$ and compute

$$c := (\mathbf{g}^w, \mathbf{g}^{aw}, m \cdot \mathbf{g}^{kw}, \mathsf{P}(\sigma, (\mathbf{g}^w, \mathbf{g}^{aw}), w; \gamma)),$$

where $\gamma := m \cdot \mathbf{g}^{kw}$

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