On the Cryptographic Deniability of the Signal Protocol

Nihal Vatandas[1(✉)], Rosario Gennaro[1], Bertrand Ithurburn[1], and Hugo Krawczyk[2]

[1] Center for Algorithms and Interactive Scientific Software, The City College of New York, New York City, USA
nihal.vatandas@gmail.com, rosario@ccny.cuny.edu, bithurburn@gmail.com
[2] Algorand Foundation, New York City, USA
hugo@ee.technion.ac.il

Abstract. *Offline deniability* is the ability to *a posteriori* deny having participated in a particular communication session. This property has been widely assumed for the Signal messaging application, yet no formal proof has appeared in the literature. In this paper, we present what we believe is the first formal study of the offline deniability of the Signal protocol. Our analysis shows that building a deniability proof for Signal is non-trivial and requires very strong assumptions on the underlying mathematical groups where the protocol is run.

1 Introduction

The ever growing privacy concerns and vulnerabilities behind online interactions have made the *deniability of communications* a prime privacy requirement. The goal is to ensure that the transcript of an online communication between two peers cannot be used as a proof to a third party that the communication took place. This property should hold even if one of the parties is willing to deviate from the protocol, just to generate a proof that reveals the identity of the peer to a third party. A well-known fact is that communications protected by a shared symmetric key do not allow one of the peers to frame the other: Any information Bob claims to have been authenticated by Alice could have been produced by Bob himself since he also knows the key.

However, usually shared keys are generated via an online authenticated key exchange (AKE) protocol, where the parties authenticate using their own private/public keys. Then, the question is whether the shared key computed in that protocol can itself be denied. This gives rise to the notion of *deniable authenticated key exchange*. As above, the transcript of the AKE protocol should leave no proof that can convince a third party. Bob should not be able to convince a *judge* that Alice communicated with him, even if Bob is malicious and departs from the prescribed protocol just to generate such proof.

Deniability of AKE was first formally defined in [14] in terms of simulatability (following an approach set forth by [17] in the context of deniable authentication). Roughly, an AKE protocol π between peers Alice and Bob is *deniable for*

M. Conti et al. (Eds.): ACNS 2020, LNCS 12147, pp. 188–209, 2020.
https://doi.org/10.1007/978-3-030-57878-7_10

Alice if the communication transcript generated in an interaction between Alice and Bob in executions of π can be simulated by Bob without Alice's participation. That is, one requires the existence of a simulator that in interactions with Bob generates transcripts that are computationally indistinguishable from real transcripts between Alice and Bob. Here, Alice's actions are those of an honest AKE participant but Bob can deviate from the protocol in arbitrary ways (if that helps him frame Alice as discussed above). Moreover, and crucially, the simulator needs to output a session key under a distribution that is indistinguishable from that of the session key in a real protocol between Alice and (possibly malicious) Bob.

Different flavors of AKE deniability are obtained by considering the type of judge to which a proof is presented. The basic case is that of *offline deniability* that assumes a judge who is not present during the communication between the peers. In this case, the judge examines a transcript presented by Bob to decide whether Alice knowingly communicated with Bob. If we let the judge be an active participant in the protocol, ensuring deniability is harder, or even impossible. In particular, if Bob provides his private keys to the judge, the latter can interact with Alice masquerading as Bob and get convinced directly of Alice's participation. The notion of *online deniability* [42] refers to a judge who participates in the protocol but is not given Bob's private keys. The offline case is of more practical relevance since a protocol that is *not deniable* in this sense can leave proofs of communication that *anyone* can verify later (possibly with Bob's help).

Deniability has been a consideration for AKE protocols for the last 25 years (at least) since the design of the Internet Key Exchange (IKEv1) protocol [21,24], through the influential *off-the-record* protocol [4]. Yet, designing deniable AKE protocols has proven to be a delicate matter and is still an active area of research (see the related work section below), due to the popularity, and privacy concerns, of internet and messaging communications.

1.1 The Case of Signal

Deniability was the central focus in the design of the Signal protocol [9]. The latter has become the de-facto standard in the area of secure messaging protocols, having been adopted in widely deployed protocols such as WhatsApp and Facebook messenger.

However in [39,40] it has already been shown that online deniability does not hold for Signal. Still, offline deniability is widely believed to hold, and yet no formal proof has ever appeared in the literature.

In this paper, we discuss the reasons why a proof for the offline deniability of Signal has been difficult to construct, and analyze ways to overcome this problem. As far as we know, this is the first formal study of the offline deniability of the Signal protocol.

For this, we focus on the offline deniability properties of a particular family of AKE protocols, namely, *implicitly authenticated* protocols. These are characterized by the property that the transcript of the protocol is *independent of the*

private keys of the peers. That is, anyone can generate the transcript messages. Authentication is provided *implicitly* by involving the peers' private keys in the derivation of the session key. Implicitly authenticated protocols seem to be perfectly suited to provide deniability. In their minimal form, all the peers exchange are Diffie-Hellman (DH) values $X = g^x, Y = g^y$ with the key being computed as a function of X, Y, the public keys of the parties and their corresponding private keys. There is little one can learn about the participants from these transcripts, hence, *intuitively,* they "must be deniable".

The above intuition, however, has been insufficient to prove these protocols deniable, in particular due to the need to simulate the session key. Thus, the question of deniability of implicitly authenticated protocols has not been settled to this day. This is not just a challenging theoretical question, but one of practical significance. Indeed, prime examples of this family are MQV [27,31], HMQV [26], and 3DH [29] (see Sect. 3 for the description of these protocols). Implicitly authenticated protocols are preferred because of their higher efficiency (since no additional signatures or encryptions are computed/sent together with the basic DH ephemeral keys). Very importantly, 3DH is the basis of the AKE underlying the Signal protocol and a main source of "intuition" regarding Signal's deniability properties.

1.2 Our Contribution

We make progress in the study of deniability of implicitly authenticated key exchange protocols in several ways:

- We demonstrate the insufficiency of implicit authentication as a way to ensure deniability. We present settings, in terms of properties of groups and assumptions, where the original MQV protocol [31] (that does not use a random oracle for key derivation) fails deniability.
- We discuss how the counter-example built around MQV illuminates the difficulties one encounters in attempting to prove deniability for any of the other implicitly authenticated AKEs we consider, including 3DH.
- Using the above result and proof technique, we are able to *characterize the non-deniability* of MQV in terms of the feasibility of the following problem: Given a random group element $A = g^a$, sample $Y = g^y$ in G (with any chosen efficient distribution), so that it is hard to compute the Diffie-Hellman value $DH(A, Y) = g^{ay}$ while it is easy to decide correctness of $DH(A, Y)$. In other words, if such a Y can be feasibly sampled in G, then MQV is non-deniable and an adversary can always prove that he communicated with a particular honest peer.
- We formulate a general "extractability assumption", termed *Knowledge of DH Assumption*, which essentially implies that the above sampling is infeasible. Therefore, under such an assumption and other variations we define, we can prove the (offline) deniability of the three studied protocols (MQV, HMQV, 3DH) in the random oracle model. The assumption can be seen as a plausible strengthening of the related knowledge of exponent assumptions [2]. Because

our result is a characterization of deniability, this assumption, although very strong, is needed to prove deniability of the protocols. An interesting open problem is to formulate a simpler and more intuitive assumption that implies it.

– We prove a general theorem showing that any two-party protocol, whose transcript can be generated from a shared symmetric key and public information, is deniable (namely, simulatable without access to the shared key) if the symmetric key is the product of a deniable AKE. This result yields the deniability of the full Signal protocol under the assumption that its underlying AKE, 3DH, is deniable.

Due to lack of space, all the proofs are omitted and can be found in the full version of this paper [41].

1.3 Related Work

Readers are referred to [3,6,8,37] for the formal definition of secure AKE. These formal treatments provide robust justifications for the security of AKE but do not deal with the issue of deniability. Informal discussions of deniability began to appear in [4,5,13,22,25,28]. Offline Deniable AKE were defined in [14] based on the work on deniable authentication in [17]. Definitions of deniable AKE that offer composability and online deniability were presented in [16,42]. Provably offline deniable AKEs are presented in [14,43], while online deniable ones are presented in [16,39,40,42]. None of these protocols is implicitly authenticated.

There are alternative, relaxed definitions of deniability that work for less expansive purposes [10,18,44]. Each of these alternative definitions involve giving the simulator access to an oracle to perform the simulation. The oracle represent the "deniability loss" with respect to the standard strict definition of simulatability.

Other works in the context of deniability include [12,23,32], and we remark on the work by Pass [33] which stresses the important differences between a deniability simulator and a zero-knowledge one (as defined in [20]).

A full specification of the Signal protocol can be found in [38]. As mentioned above it uses the Extended Triple Diffie-Hellman (X3DH) key agreement protocol [30] (built on the 3DH AKE [29]) followed by a communication session which include a *ratcheting* mechanism to refresh keys [34]. Security analyses of these protocols that do not include deniability can be found in [1,9].

The MQV protocol was presented in [27,31]. It inspired several other protocols including HMQV by Krawczyk in [26] and the OAKE family of AKEs by Yao and Zhao [44]. No formal deniability proof of these protocols has appeared anywhere. In [44], one protocol in the OAKE family is claimed to have the weaker notion of reasonable deniability. Our full deniability results extend to the OAKE family as well. We will present those in the full version.

2 Preliminaries

In the following, we denote with G a cyclic group of prime order q generated by g. For every element $X \in G$ there exist an integer $x \in \mathbb{Z}_q$ such that $X = g^x$. We say that x is the discrete log of X with respect to g and denote it with $x = \log_g X$. Given two elements $A = g^a$ and $B = g^b$ we denote with $DH(A, B) = g^{ab} = A^b = B^a$, the Diffie-Hellman transform of A, B, [15].

With $a \leftarrow S$ we denote the process of sampling a uniformly at random in the set S.

The following definition states that computing the discrete log is hard.

Definition 1. *Let G be a cyclic group of prime order q generated by g. We say that the (T, ϵ)* Discrete Log Assumption *holds in G if for every probabilistic Turing Machine $\mathcal{A}$ running in time T we have that*

$$Prob[x \leftarrow \mathbb{Z}_q \; ; \; \mathcal{A}(g^x) = x] \leq \epsilon$$

The following definition states that computing the Diffie-Hellman transform is hard.

Definition 2. *Let G be a cyclic group of prime order q generated by g. We say that the (T, ϵ)* Computational Diffie-Hellman (CDH) Assumption *holds in G if for every probabilistic Turing Machine $\mathcal{A}$ running in time T we have that*

$$Prob[A, B \leftarrow G \; ; \; \mathcal{A}(A, B) = DH(A, B)] \leq \epsilon$$

Consider the set $G^3 = G \times G \times G$ and the following two probability distributions over it:

$$\mathcal{R}_G = \{(g^a, g^b, g^c) \text{ for } a, b, c \in_R [0..q]\}$$

and

$$\mathcal{DH}_G = \{(g^a, g^b, g^{ab}) \text{ for } a, b, \in_R [0..q]\}$$

We use these distributions in the following definition:

Definition 3. *We say that the (T, ϵ)* Decisional Diffie-Hellman (DDH) Assumption *holds over $G = \langle g \rangle$ if the two distributions $\mathcal{R}_G$ and $\mathcal{DH}_G$ are (T, ϵ)-indistinguishable.*

For Definition 3, we use Goldwasser and Micali's classical definition of computational indistinguishability [19].

Definition 4. *Let $\mathcal{X}, \mathcal{Y}$ be two probability distributions over A. Given a circuit D, the distinguisher, consider the following quantities*

$$\delta_{D,\mathcal{X}} = Prob_{x \in \mathcal{X}}[D(x) = 1]$$

$$\delta_{D,\mathcal{Y}} = Prob_{y \in \mathcal{Y}}[D(y) = 1]$$

We say that the probability distributions $\mathcal{X}$ and $\mathcal{Y}$ are (T, ϵ)-indistinguishable *if for every probabilistic Turing Machine D running in time T we have that $|\delta_{D,\mathcal{X}} - \delta_{D,\mathcal{Y}}| \leq \epsilon$.*

3 Implicitly Authenticated Key Exchange

In this section, we denote the peers as Alice and Bob. They each hold long-term public keys $A = g^a, B = g^b \leftarrow G$ respectively. The values $a, b \leftarrow \mathbb{Z}_q$ are their respective long-term private keys. In all the protocols we describe in this Section, Alice is the initiator and sends an ephemeral value $X = g^x \leftarrow G$ while keeping $x \leftarrow \mathbb{Z}_q$ secret. Bob is the responder and he answers with $Y = g^y \leftarrow G$ while keeping $y \leftarrow \mathbb{Z}_q$ secret. The protocols differ on how the resulting session key is computed.

MQV and HMQV: The MQV [27,31] and HMQV [26] protocols are described in Fig. 1. Note that in MQV the session key is defined as the group element $\tilde{K}$ (with the use of a hash function left as optional), while HMQV mandates the use of a hash function H to derive the session key from the secret value $\tilde{K}$ shared by Alice and Bob.

Triple Diffie-Hellman. Figure 2 describes the 3DH protocol [29]. The "three Diffie-Hellman" part refers to three separate Diffie-Hellman operators concatenated together and then passed to a hash function to derive a key.

We postpone discussions about how 3DH is used inside Signal to Sect. 9. Here, we simply describe 3DH as a basic key exchange protocol where both parties are alive and communicating with each other.

Key Registration. In the following, MQV [resp.HMQV or X3DH] with Key Registration denotes the MQV [resp. HMQV or X3DH] protocol where participants are required to prove knowledge of their long-term secret keys via an extractable

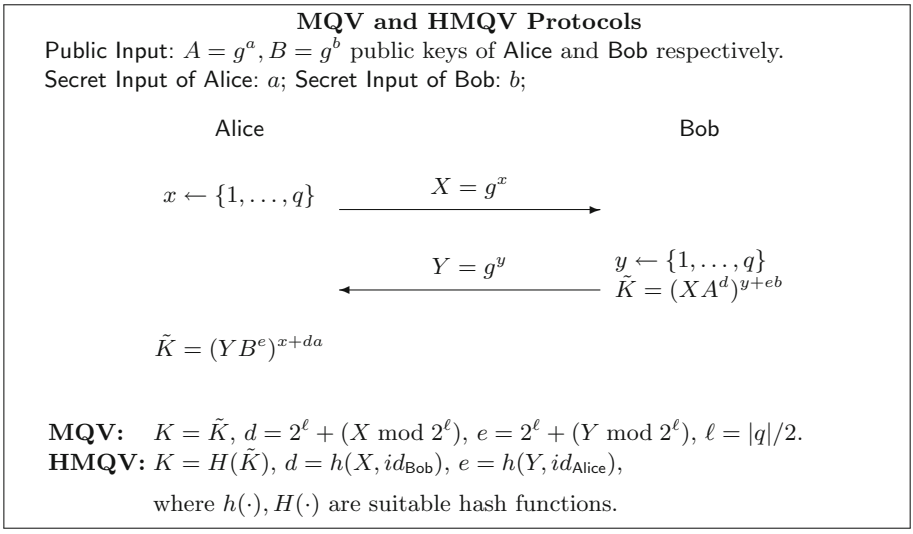

Fig. 1. Computation of the session key K in MQV and HMQV protocols

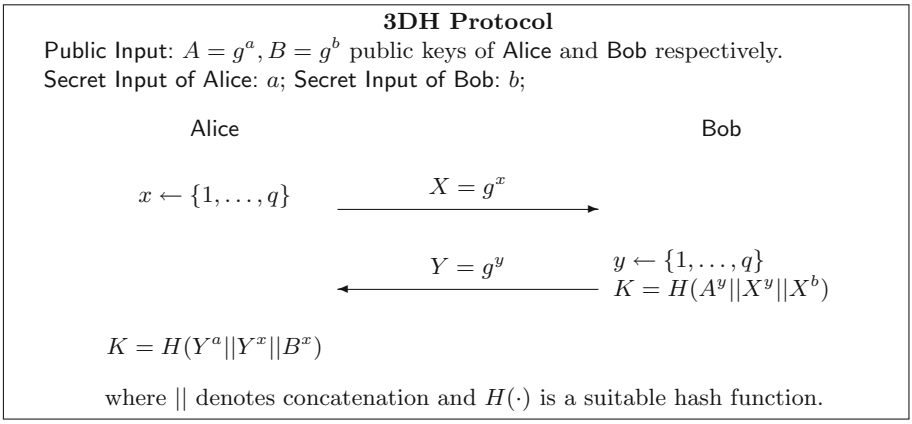

Fig. 2. Triple Diffie-Hellman key exchange protocol

proof of knowledge. This step is usually performed when players register their keys with a Public Key Infrastructure (PKI).

For the case of MQV/HMQV/X3DH, this step can be achieved by running the classic Schnorr proof [35].

4 Deniable Key Exchange

We recall the definition of deniable Authenticated Key Exchange (AKE in the rest) from [14,16,42].

An AKE protocol works with two parties, A and B, who want to agree on a secret key K. Each party has a long-term public/secret key pair which is associated to the party through a trusted registry. These key pairs are generated via a key generation phase. For notation purposes, A has public key $\mathsf{pk_A}$ and secret key $\mathsf{sk_A}$ – B has $\mathsf{pk_B}$ and $\mathsf{sk_B}$.

One of A and B acts as the initiator and the other acts as the responder. The protocol results in session key K. Informally, security for AKE requires that if an honest party A outputs key K and associates it to an honest party B, then no party other than B may know anything about K. Additional security properties can be enforced, such as perfect forward secrecy (past session keys remain secure, even if long-term keys are compromised), security against state compromise (ephemeral values are revealed to the adversary), etc. A formal treatment of AKE security can be found in [3,6,8,37].

Informally, we say that an AKE is *deniable* if it prevents A or B from proving to a third party (which we will call the *judge*) that an AKE occurred with a particular party. A weaker goal is to be able to deny just the contents of communication protected by the AKE's resulting key.

Recall that a KE protocol involves two communicating parties: the initiator, who sends the first message, and the responder. Let Σ denote an AKE protocol

with key generation algorithm KG and interactive machines Σ_I and Σ_R, which respectively denote the roles of the initiator and responder. Both Σ_I and Σ_R take as input their own secret and public keys. In most cases, they also take in the identity and public key of their peer, but other AKE protocols specify that the parties learn this information during the session [7]. The term *session* denotes an individual run of a KE protocol. When the protocol finishes, it outputs either an error or a session key.

Adversary. Let $\mathcal{M}$ denote an adversary that runs on an arbitrary number of randomly chosen public keys $\mathbf{pk} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_l)$ generated by KG. The algorithm associates the keys to honest users in the network. $\mathcal{M}$'s input also includes some arbitrary auxiliary information $aux \in AUX$. The adversary runs Σ with an arbitrary number of honest users. Sometimes $\mathcal{M}$ acts as the initiator, and other times $\mathcal{M}$ acts as the responder. The individual sessions run in a concurrent setting, so $\mathcal{M}$ may schedule and interleave them arbitrarily.

View. We define the view of the adversary $\mathcal{M}$ as its internal coin tosses, the transcript of the entire interaction, and the session keys from each session that $\mathcal{M}$ participates either as an initiator or responder. Sessions that do not produce keys result in a key defined by an error value. We denote this view by $\mathsf{View}_{\mathcal{M}}(\mathbf{pk}, aux)$.

Simulator. In order to demonstrate deniability with respect to initiator (*resp.*, responder), the simulator takes the role of the initiator I (*resp.*, responder R) and imitates I (*resp.*, R) without having the long-term secret key sk_I (*resp.*, sk_R).

As input, the simulator receives some random coins, the public keys $\mathbf{pk}$ of all parties and any auxiliary input aux available to the adversary. It generates $\mathsf{Sim}_{\mathcal{M}}(\mathbf{pk}, aux)$ by interacting with the adversary $\mathcal{M}$ as its peer. $\mathsf{Sim}_{\mathcal{M}}(\mathbf{pk}, aux)$ includes the transcript and the resulting shared key of the session.

The simulator provides the inputs to the adversary $\mathcal{M}$ prior to the protocol execution and observes all communication $\mathcal{M}$ has with its environment (such as AKE sessions $\mathcal{M}$ holds with other honest parties and the random oracle queries). The random oracle (RO) queries made by the adversary are visible to the simulator. However, the simulator might not be able to freely tamper with the RO input-output pairs (program the RO), because the judge is granted access to the random oracles, too. Therefore the RO queries involved in the simulation are expected to be consistent with the possible queries made by the judge and other honest parties running a session with the adversary.

Definition 5. *[14] An AKE protocol* $(\mathsf{KG}, \Sigma_I, \Sigma_R)$ *is* $(T_{\mathcal{M}}, T_{\mathcal{S}}, T_{\mathcal{D}}, \epsilon)$ concurrently deniable *with respect to the class of auxiliary Inputs AUX if for any adversary* $\mathcal{M}$ *running in time* $T_{\mathcal{M}}$*, on input honest parties' public keys* $\mathbf{pk} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_l)$ *generated by* KG *and any auxiliary input* $aux \in AUX$*, there exists a simulator* SIM *running in time* $T_{\mathcal{S}}$ *on the same inputs as* $\mathcal{M}$ *which produces a simulated view* $\mathcal{S}im(\mathsf{pk}, aux)$ *such that the following two distributions are* $(T_{\mathcal{D}}, \epsilon)$*-indistinguishable:*

$$\mathcal{R}eal(aux) = [(\mathsf{sk}, \mathsf{pk})_{I,R} \leftarrow \mathsf{KG}; (aux, \mathbf{pk}, \mathsf{View}_{\mathcal{M}}(\mathbf{pk}, aux))]$$

$$\mathcal{S}im(aux) = [(\mathsf{sk}, \mathsf{pk})_{I,R} \leftarrow \mathsf{KG}; (aux, \mathbf{pk}, \mathsf{Sim}_{\mathcal{M}}(\mathbf{pk}, aux))]$$

The definition follows the usual simulation paradigm which guarantees that the judge cannot decide if anything that the adversary presents (the view) is the result of an interaction with a real party or the product of a simulation. As pointed out by Pass in [33], the simulation requirements for deniability are much stronger than for Zero-Knowledge simulation. Indeed while a ZK simulator can be interpreted as a "thought experiment", a deniability simulator is a real algorithm that must exist in real life, to give a plausible explanation of the adversary view which does not include an honest party. In particular this means that standard simulation techniques for ZK, such as rewinding, random oracle programmability or common parameters are not available to a deniability simulator.

Why the session key is included in the view: Note how we include the session key in the view of the adversary. This guarantees that the contents of the session authenticated by the key can also be denied. Intuitively, since we do not know how the key will be used in the session following the AKE, we must include it in the view. Theorem 1 formalizes this intuition.

5 Deniable Sessions

As we noted above the definition of deniability of an AKE explicitly includes the session key in the view in order for us to claim that any deniability is preserved in any subsequent use of the key in the session that follows the AKE. We now formally prove this statement[1]. First we define deniability for an arbitrary interactive protocol between two parties, and then we show that any communication structured as an AKE, followed by messages where the two parties only use the session key (but not their long-term secret keys) is deniable.

Consider an adversary $\mathcal{M}$ that on input $\mathbf{pk}$ interacts with the parties holding the public keys. The adversary also may have auxiliary input aux drawn from distribution AUX.

$\mathcal{M}$ initiates several concurrent interactions with the parties and we define the adversary's *view* of the interaction as $\mathcal{M}$'s coin tosses together with the transcript of the full interactions. We denote the view as $\mathsf{View}(\mathbf{pk}, aux)$.

Definition 6. *We say that an interactive protocol* $\mathcal{P}$ $(\mathsf{KG}, \mathsf{I}, \mathsf{R})$ *is* $(T_{\mathcal{M}}, T_{\mathcal{S}}, T_{\mathcal{D}}, \epsilon)$-concurrently deniable *with respect to the class of auxiliary inputs AUX if for any adversary* $\mathcal{M}$ *running in time* $T_{\mathcal{M}}$*, on input honest parties' public keys* $\mathbf{pk} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_l)$ *generated by* KG *and any auxiliary input* $aux \in AUX$*,*

[1] This was claimed informally and without proof in [14]; here, we use this result in essential way in Sect. 9 to show how deniability of 3DH carries to deniability of the whole Signal protocol.

there exists a simulator $SIM_{\mathcal{M}}$ running in time $T_{\mathcal{S}}$ on the same inputs as $\mathcal{M}$, such that the following two distributions are $(T_{\mathcal{D}}, \epsilon)$-indistinguishable

$$\mathcal{R}eal(aux) = [(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KG}; (aux, \mathbf{pk}, \mathsf{View}(\mathbf{pk}, aux))]$$

$$\mathcal{S}im(aux) = [(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KG}; (aux, \mathbf{pk}, SIM(\mathbf{pk}, aux))]$$

We now define a *session*. Consider two parties Alice and Bob with associated public keys pk_A, pk_B. They also hold private keys sk_A, sk_B respectively, and also additional inputs x_A, x_B. We say that an interactive protocol P between Alice and Bob is a *session* if

- $P = [P_1, P_2]$ where P_1 is an AKE. Let K be the session key resulting from the execution of P_1
- every message sent by Alice (resp. Bob) in P_2 is a function only of the transcript so far, the private input x_A (resp. x_B), and the session key K, but *not* of the private keys sk_A (resp. sk_B)

Theorem 1. *Let $P = [P_1, P_2]$ be a session, where P_1 is a deniable authenticated key exchange according to Definition 5, which includes the session key in the view. Then P is deniable according to Definition 6.*

6 Negative Examples

In this section, we present a strategy that (on certain groups) allows an adversary to prove that an interaction took place for the MQV protocol. This negative result will then point the way to what type of assumption about the underlying group we need to make to guarantee deniability in a provable way.

Consider the MQV protocol in Fig. 1 and let us try to prove that the protocol is deniable for Alice. In other words we need to construct a simulator SIM who plays the role of Alice while talking to Bob. SIM is given the public key of Alice, $A = g^a$, but not the corresponding secret key, a.

SIM runs Bob to simulate the conversation and observes all of Bob's communication in his environment. SIM starts by providing the random coins r, and, if available, the auxiliary information to Bob. SIM then chooses a random x and sends out $X = g^x$ to Bob. In return it receives a group member $Y \in G$ from Bob. SIM's final output must be indistinguishable from Bob's view (r, X, Y, K) and the only thing that SIM does not know is K. Recall that in MQV

$$K = g^{xy} g^{ayd} g^{xbe} g^{abde}$$

where e, d are values computed from the transcript.

When Bob is honestly executing the protocol, the simulator is easy to construct. If Bob follows the protocol and computes Y as g^y for $y \in_R \mathbb{Z}_q$, the simulator can do exactly the same thing and compute the session key $K = g^{xy} A^{yd} (XA^d)^{be}$. Here, we assume that the key generation phase requires

an additional key *registration* step where parties prove in ZK knowledge of their secret key[2]. This allows the simulator to have knowledge of b.

However, a malicious Bob can deviate from the protocol at will and having Bob's random coins provides SIM no information about how Y is actually sampled. In the simulation above, SIM can compute two of the DH values $g^{xy} = Y^x$ and $(XA^d)^{be}$ since b and x are known. But $DH(A, Y) = g^{ay}$ cannot be computed because neither a (secret key of Alice) nor $y = \log_g Y$ is known to SIM(maybe not even to Bob).

The only option for SIM would be putting a random string as the simulated key, hoping that a random value is indistinguishable from the actual key. Such strategy would work if we could invoke the DDH assumption to claim the two distributions are indistinguishable. However, a random string does not necessarily substitute for g^{ay}, because though a is uniformly selected, DDH does not apply for an adversarially chosen Y.

6.1 When MQV Is Provably *Non-deniable*

The discussion above shows that an adversarially chosen Y is a barrier to prove deniability for MQV. We now prove that over some groups, it is actually impossible to prove that MQV is deniable, because there is a strategy for Bob to prove that he interacted with Alice.

Assume we are running the protocol over a cyclic group G setting where:

1. The DDH problem is easy
2. The following experiment succeed with negligible probability for every efficient adversary Adv and any efficiently samplable distribution $\mathcal{Y}$
   - Adv runs on input $A, B \in G$ chosen uniformly at random and outputs $X \in G$
   - Adv receives $Y \in G$ chosen according to $\mathcal{Y}$
   - Adv succeeds if it outputs $K_A = (XA^d)^y(XA^d)^{be} = DH(XA^d, Y)DH(XA^d, B^e)$ where d, e are defined as in the MQV protocol

We note that point (2) is necessary for the security of the MQV protocol: in fact a successful adversary Adv could easily impersonate Alice. Point (1) can be guaranteed if for example the group G supports a bilinear map.

On assuming that the DDH is easy. Before we proceed with our counter-example, let us discuss our assumption that the DDH problem is easy in G. Note that the MQV protocol stops being provably secure in a model where security is defined as the indistinguishability of the session key from a random value (i.e., from a random group element in the case of MQV). So it would seem that we are proving a deniability counter-example for an insecure protocol, and one could wonder what's the value in that. There are several answers to this point:

[2] For showing the failure of (proofs of) deniability, assuming a key registration step makes our negative result stronger as it implies that *even if* we allow key registration we do not know how to simulate, and in some cases simulation is actually impossible.

- First of all, one could consider a weaker security notion for key exchange to say that the session key should have "enough" computational entropy rather than being indistinguishable from random. This would be a much weaker but still useful security definition as a full entropy key could be derived via a randomness extractor. It is conceivable that MQV is a secure protocol under this definition (one would have to prove it) but provably non-deniable because of our counter-example.
- Deniability is an orthogonal property to that of security. Our counter-example is designed to illuminate the difficulties of proving deniability for MQV and similar protocols, and demonstrating the failure of the intuition that the sole lack of explicit authentication methods (such as digital signatures) is sufficient to assume that deniability holds.
- Additionally, we point out that there are so-called *trapdoor DDH* groups [11, 36], where the DDH problem is conjectured to be hard unless one is in possession of a trapdoor. In this case, the protocol is secure for anybody who does not possess the trapdoor but non-deniable for a judge who holds the trapdoor.

The counter-example. We now show a strategy that incriminates Alice. A malicious Bob samples Y uniformly at random in the group but in a way in which he can demonstrate that he does not know $y = \log_g Y$, for example by hashing a publicly known string (e.g. today's copy of the NY Times) into a group element via a sufficiently complicated hash function (which could be modeled as a random oracle[3]).

We now prove by contradiction that there cannot be a simulator. If there is a simulator SIM, let K_S be the key provided by the simulator, while $K = (XA^d)^y(XA^e)^b = DH(XA^d, Y)DH(XA^e, B)$ is the real key. We assume that Bob is willing to reveal b to the judge in order to prove that an interaction took place.

The knowledge of b allows the judge to compute $z = DH(XA^d, B^e) = (XA^d)^{be}$. Since the DDH is easy, the judge can decide if $K = K_S$ by checking if $K_S \cdot z^{-1} = DH(XA^d, Y)$. Therefore, anything other than the authentic key is detected by the judge. So the only possible successful simulator is the one that outputs $K_S = K$. But that means that SIM contradicts assumption (2) above.

3DH without hashing. It is not hard to see that a similar reasoning applies to an "unhashed" version of 3DH where the session key is set as $K = DH(A, Y)||DH(X, Y)||DH(X, B)$. Therefore such a version of 3DH would also be provably non-deniable under the above conditions.

[3] It is not necessary to model this hash as a random oracle, as long as we assume that computing g^{ay} is hard when $A = g^a$ is sampled uniformly at random and $Y = g^y$ is sampled according to the procedure used by Bob.

6.2 Does the Random Oracle Help?

In HMQV and 3DH the session key is computed by hashing the secret shared value, i.e.

$$K = H[DH(XA^d, Y)DH(XA^d, B^e)]$$

in HMQV and

$$K = H[DH(A, Y)||DH(X, Y)||DH(X, B)]$$

in 3DH. If we model H as a random oracle, would this help in solving the problems described in the previous section?

The question is still how can the simulator provide a session key which is indistinguishable from the real one. In this case, one would hope that the use of the random oracle will allow the simulator to identify the key. Assume Bob is the malicious party and can deviate from the honest execution of the protocol. Every time Bob makes a random oracle query, SIM sees it (even though it is not allowed to choose the answer for it [33]).

If Bob computes the real session key K that matches A, B, X, Y, then he must have queried $DH(XA^d, Y)DH(XA^d, B^e)$ (resp. $DH(A, Y)||DH(X, Y)||DH(X, B)$ for 3DH) to the random oracle.

First of all there is no guarantee that Bob will actually query the random oracle on those points. But even if he did, it is not clear how the simulator can identify the correct query that corresponds to the computation of the session key. Indeed, the simulator SIM is able to compute g^{bx} and g^{xy}, but cannot compute g^{ay} since a and y are not known. If Y is uniformly distributed and the DDH holds, SIM cannot *provably* detect which query corresponds to the session key[4].

The only option is to choose a random key, but this is distinguishable from the real view if Bob presents to the judge the correct input to the random oracle (e.g. Bob knows $y = \log_g Y$ and can convince the judge that the session key was computed using the correct input).

Note that if this is the case, Bob still cannot convince the judge that he spoke to Alice. In fact if Bob knows y, then the entire transcript could be his own creation without ever interacting with Alice.

In other words, we have one of two possible cases: either Bob does not know y (and the input to the random oracle) in which case the simulator should be able to put a random key in the view, or Bob knows y (and the correct input to the random oracle) in which case the simulator should be able to identify the correct key from the knowledge of Bob. The problem is that we do not know which case we are in, and therefore we cannot complete the simulation successfully.

The way out of this problem is described in the next section and relies on an appropriate "knowledge extraction" from Bob, which will address also the issues related to the previous counter-example.

[4] One could use a Gap-DDH Assumption, which states that the CDH Assumption holds even in the presence of an oracle that decides the DDH. Then such oracle could be provided to the simulator to detect the query. Yet this simulator would not be a legitimate *deniability* simulator unless the oracle could be implemented in real-life.

7 A Characterization for Non-deniability

In this section, we show that the sampling strategy shown above to make MQV non-deniable is essentially the *only* one. More specifically we prove that if an adversary is able to "frame" one of the parties in the MQV protocol and prove that an interaction took place, then we have a way to sample a group element Y in G in a way that it is hard to compute $DH(A, Y)$ for a fixed group element A, but it must be easy to detect that $DH(A, Y)$ is correct.

The consequence is that if we assume that such a task is computationally infeasible then we can prove (albeit non-constructively) that the MQV protocol is deniable. Details follows.

Non-deniable AKE. First we define what a non-deniable or *incriminating* AKE is, as the logical negation of deniability.

We call a key exchange protocol (KG, I, R) as $(T_{\mathcal{M}}, T_{\mathsf{SIM}}, T_{\mathsf{J}}, \varepsilon_{\mathsf{J}})$-*incriminating* if there is an adversary $\mathcal{M}$ running in time $T_{\mathcal{M}}$ for which, all simulators SIM running in time T_{SIM}, there exists a judge J running in time T_{J} which distinguishes the uniformly selected samples from the following distributions with probability at least ε_{J}.

$$\mathcal{R}eal = \{\mathsf{View}_{\mathcal{M}}(pk_i)\}_{(sk_i,pk_i)\leftarrow KG}$$
$$\mathcal{S}im = \{\mathsf{SIM}_{\mathcal{M}}(pk_i)\}_{(sk_i,pk_i)\leftarrow KG}$$

$$|Pr_{x\in\mathcal{R}eal}[\mathsf{J}(x) = 1] - Pr_{x\in\mathcal{S}im}[\mathsf{J}(x) = 1]| \geq \varepsilon_{\mathsf{J}}$$

$\mathsf{View}_{\mathcal{M}}$ includes the public keys, the transcript, the session key and random coins r given to $\mathcal{M}$. (sk_i, pk_i) denotes long-term key pairs of parties for $i \in \{I, R\}$ (I for initializer, R for responder).

7.1 Bad Sampler

We now define a particular type of sampling algorithm for G which we call a *Bad Sampler*. We will prove that the existence of a bad sampler is equivalent to MQV being incriminating.

We say that a sampling algorithm for G is $(T_{\mathsf{Samp}}, T_{\mathsf{Solv}}, T_{\mathsf{D}}, \varepsilon_{\mathsf{Solv}}, \varepsilon_{\mathsf{D}})$-$\mathsf{Bad}$ if the following conditions are satisfied:

There exists a sampling algorithm Sample which satisfies the following

1. Sample takes as input A (uniformly picked from G) and the random coins r to generate $Y = \mathsf{Sample}(A, r)$ running in time $\leq T_{\mathsf{Samp}}$.

2. $\forall$ Solve running in T_{Solv}

$$Pr_{\mathsf{Solve},\ A,\ r}[\mathsf{Solve}(A, Y, r) = g^{ay} \mid Y = \mathsf{Sample}(A, r)] \leq \varepsilon_{\mathsf{Solv}}$$

Probability is over the randomness of Solve, uniform choice of $A = g^a$ and random coins r.

3. There exists a distinguisher D running in time $\leq T_{\mathsf{D}}$ which tells apart g^{ay} from a random group member $\hat{g} \leftarrow G$ for a uniformly chosen A and random coins r.

$$|Pr_{\mathsf{D},\,A,\,r}[\mathsf{D}(A, Y, r, g^{ay}) = 1 \mid Y = \mathsf{Sample}(A, r)] - Pr_{\mathsf{D},\,A,\,r}[\mathsf{D}(A, Y, r, \hat{g}) = 1 \mid Y = \mathsf{Sample}(A, r)]| \geq \varepsilon_{\mathsf{D}}$$

7.2 Equivalence Between Bad Sampling and Incrimination

In the following Theorem, if T is the running time of an algorithm then the notation $\tilde{T}$ means T plus a constant number of exponentiations in G.

Theorem 2. *If there is a* $(T_{\mathsf{Samp}}, T_{\mathsf{Solv}}, T_{\mathsf{D}}, \varepsilon_{\mathsf{Solv}}, \varepsilon_{\mathsf{D}})$*-bad sampler in* G *then the MQV protocol is* $(\tilde{T}_{\mathcal{M}}, \tilde{T}_{\mathsf{SIM}}, \tilde{T}_{\mathsf{J}}, \varepsilon_{\mathsf{J}})$*-incriminating with* $\varepsilon_{\mathsf{J}} = \varepsilon_{\mathsf{D}}(1 - \varepsilon_{\mathsf{Solv}})$*.*

Conversely if the MQV protocol run over G *is* $(T_{\mathcal{M}}, T_{\mathsf{SIM}}, T_{\mathsf{J}}, \varepsilon_{\mathsf{J}})$*-incriminating then there exists a* $(\tilde{T}_{\mathsf{Samp}}, \tilde{T}_{\mathsf{Solv}}, \tilde{T}_{\mathsf{D}}, \varepsilon_{\mathsf{Solv}}, \varepsilon_{\mathsf{D}})$*-bad sampler for* G*, with* $\varepsilon_{\mathsf{Solv}} = (1 - \varepsilon_{\mathsf{J}})$ *and* $\varepsilon_{\mathsf{D}} = \varepsilon_{\mathsf{J}}$*.*

Malicious Initiator. The theorem above proves the equivalence of bad sampling with the non-deniability of MQV for the initiator when interacting with a malicious responder. It is also not hard to see that a similar theorem holds for the case of a malicious initiator who is trying to incriminate the responder. In this case also, the only possible strategy for a malicious initiator will be to run a bad sampler.

Theorem Interpretation. The above theorem characterizes the strategy that the adversary needs to follow to be able to incriminate one of the parties: the only way to do it is to be able to sample elements Y in G such that for every element $A \leftarrow G$ it is easy to decide if $DH(A, Y)$ is correct while it is still hard to compute it. If we assume that such "bad" sampling is infeasible, then we immediately have a proof that the protocols are deniable. Yet such proof is a 'non-constructive' one, as we are not able to show how the simulator works, but just that it must exist. The significance of such a statement in real-life is not clear, as plausible deniability requires the judge to actually run a simulator to counter Bob's statement that he spoke to Alice. In the absence of such real simulator, there is no way to argue that the conversation was not generated by Alice, even if we assume that bad sampling is impossible.

As before we are stuck on the fact that when we are trying to simulate a malicious Bob we do not know if he did sample $Y = g^y$ with or without knowledge of y (or more precisely with or without knowledge of $DH(A, Y)$). The above theorem says that if bad sampling is impossible then either Bob must know $DH(A, Y)$ or the value is indistinguishable from random: in either case we would be able to successfully complete the simulation if we knew which case we were in (and in the former be able to "extract" $DH(A, Y)$). But the mere assumption that bad sampling is impossible does not give us this knowledge, and therefore leaves us stuck in the construction of a simulator.

The next section shows how to define an appropriate "knowledge extractor" for Bob, that will allow us to build a simulator.

8 Deniability Proof

As we discussed in the previous section, the roadblock in the construction of a deniability simulator is that the simulator does not know if a malicious Bob knows the value $DH(A, Y)$ or not, at the moment he sends Y out. We also showed that the only way a malicious Bob can frame Alice is if he samples a Y for which he does *not* know $DH(A, Y)$, but such value can be efficiently recognized as correct (i.e. distinguished from a random value).

8.1 The Case of MQV

The above discussion therefore points out to the natural definition of a "knowledge extractor" which allows us to build a simulator for MQV. If we assume that given a malicious responder Bob, we can either (i) extract the value $DH(A, Y)$ or (ii) assume that $DH(A, Y)$ is indistinguishable from random, then the simulator immediately follows as the output of the extractor will be the simulated key.

We call this the Strong Knowledge of DH (SKDH) Assumption and it is defined below. In the next section we define a weaker version of this assumption which will be sufficient to prove HMQV and 3DH.

Definition 7. *Let G be a cyclic group and AUX a class of auxiliary inputs. Let M be a probabilistic Turing Machine running in time T_M on input (U, aux) where $U \in_R G$, and $aux \in AUX$, and outputs $Z \in G$; we denote with $Z = M(U, aux, r)$ the output of running M on input U, aux with coin tosses r. We say that the $(T_M, T_{\hat{M}}, T_D, \varepsilon_D)$-SKDH Assumption holds over group G and class AUX, if for every such M, there exists a companion probabilistic Turing Machine $\hat{M}$ (called the extractor for M) such that: $\hat{M}$ runs on input U, aux, r in time $T_{\hat{M}}$ and outputs $\hat{Z} \in G$ such that the distributions*

$$[U, aux, r, DH(U, Z)] \quad \textit{and} \quad [U, aux, r, \hat{Z}]$$

are (T_D, ε_D)-indistinguishable.

Remark: Basically the assumption says that for every sampler M of a value Z, there is an extractor that either computes $DH(U, Z)$ or produces an output distribution that is computationally indistinguishable from $DH(U, Z)$ even when given the internal coin tosses of M. The assumption is written generically: when Bob [resp. Alice] is the adversary $U = A$ [resp. $U = B$] the peer's long-term public key, and $Z = Y$ [resp. $Z = X$] the adversary's ephemeral value.

Theorem 3. *Under the $(T_M, T_{\hat{M}}, T_D, \varepsilon_D)$ SKDH Assumption, MQV with Key Registration is a $(\tilde{T}_M, \tilde{T}_{\hat{M}}, \tilde{T}_D, \varepsilon_D)$ deniable AKE.*

8.2 The Case of HMQV and 3DH

For HMQV and 3DH we can use a weaker assumption. In this case, when the extractor fails to produce $DH(A, Y)$ we do not need to establish that $DH(A, Y)$ is indistinguishable from random, but simply that it cannot be computed by anybody. This is sufficient because the session key (in both HMQV and 3DH) is the result of a random oracle call over a function of $DH(A, Y)$. If nobody (except Alice) knows the input to the random oracle call that produces the session key, then the session key can be simulated with a random value.

Definition 8. *Let G be a cyclic group and AUX a class of auxiliary inputs. Let M be a probabilistic Turing Machine running in time T_M which runs on input (U, aux) where $U \in_R G$, and $aux \in AUX$, and outputs $Z \in G$; we denote with $Z = M(U, aux, r)$ the output of running M on input U, aux with coin tosses r.*

We say that the $(T_M, T_{\hat{M}}, T_C, \varepsilon_C)$ Knowledge of DH (KDH) Assumption *holds over group G and class AUX, if for every such M, there exists a companion probabilistic Turing Machine $\hat{M}$ (called the extractor for M) such that: $\hat{M}$ runs on on input U, aux, r in time $T_{\hat{M}}$ and outputs $\hat{Z} \in G$ or $\hat{Z} = \perp$ such that*

- *If $\hat{M}(U, aux, r) = \hat{Z} \neq \perp$ then $\hat{Z} = DH(U, Z)$*
- *If $\hat{M}(U, aux, r) = \perp$ then for every probabilistic Turing Machine C running in time T_C we have that*

$$Prob[C(U, Z, r, aux) = DH(U, Z)] \leq \varepsilon_C$$

where the probabilities are taken over $U \leftarrow G$, r and the coin tosses of $\hat{M}$ and C, conditioned on $\hat{M}(U, aux, r) = \perp$.

The first condition says that if the extractor outputs a group element, this element must be $DH(U, Z)$, otherwise no algorithm C can compute the value $DH(U, Z)$ even if C is given the coin tosses of M.

Remark. Another way to look at the KDH Assumption is as follows. The adversary M samples an element Z in G. For this particular sampling algorithm the extractor $\hat{M}$ tells us if computing $DH(U, Z)$, given the random coins of M, is feasible or not, and if it is feasible it actually gives us the value $DH(U, Z)$.

Theorem 4. *Under the $(T_M, T_{\hat{M}}, T_C, \varepsilon_C)$ KDHA, HMQV with Key Registration is a $(\tilde{T}_M, \tilde{T}_{\hat{M}}, \tilde{T}_C, \varepsilon_C)$ deniable AKE in the random oracle model.*

For the case of 3DH the Key Registration step is not necessary since the value g^{ab} (the Diffie-Hellman transform of the long-term secret keys) is not included in the session key.

Theorem 5. *Under the $(T_M, T_{\hat{M}}, T_C, \varepsilon_C)$ KDHA, 3DH is a $(\tilde{T}_M, \tilde{T}_{\hat{M}}, \tilde{T}_C, \varepsilon_C)$ deniable AKE in the random oracle model.*

9 3DH vs Signal

We refer the reader to [9] for a full description of the Signal protocol and its security analysis. Informally, we can describe Signal as an initial AKE which establishes a *root key*, followed by a secure session where messages are exchanged. However each message exchange is accompanied by a *ratcheting* step, which generates new session key. These sequence of keys, creates a *key chain* where keys are authenticated by their predecessor in the chain. In a *symmetric ratcheting* step the current chain key K is fed to a KDF function to generate two keys, the new chain key K_1 and the key K_2 used to encrypt/authenticate the message at this round. In a *asymmetric* ratcheting the parties perform a new Diffie-Hellman exchange over two ephemeral keys and feed the result to a KDF together with the current chain key, also outputting K_1, K_2 as above.

Note how, in the above description, after the initial AKE which establishes a session key K, the messages exchanged in the protocol do *not* use the long-term secret keys of the parties. Therefore, if the initial AKE is deniable, we can apply Theorem 1 and claim the deniability of Signal.

The X3DH Protocol. If the initial AKE protocol in Signal were 3DH, we would be done. However to enable asynchronous communication (where Bob, the responder, could be offline at the moment in which Alice, the initiator, sends him a message), the Signal protocol uses the X3DH variant of 3DH. This variant allows Bob to load his ephemeral key Y onto a key distribution server (a *pre-key* in Signal jargon). To prevent impersonation attacks by the server, Bob will *sign* Y with his long term secret key. When Alice wants to talk to Bob she queries the key distribution server for Bob's ephemeral key and runs the 3DH protocol to establish a root chain key K_1 and a message key K_2 used to secure the first message she sends to Bob. At this point Alice and Bob will continue with the ratcheting mechanism described above. We now move to establish the deniability of X3DH.

It is not hard to see that the proof of deniability of 3DH extends to X3DH in the case of the initiator. Indeed, the deniability argument for Alice in X3DH is the same as for the responder in 3DH since here Alice acts on the ephemeral value Y chosen by Bob. In contrast, deniability with respect to Bob in X3DH is complicated by the fact that Bob signs the value Y. But note that Bob places Y and its signature on a public server that anyone can access. Thus, Y is not bound to any specific peer, and cannot be used as a proof that Bob communicated with anyone.

Formally, we can consider Y and its signature as "auxiliary information" that an adversarial Alice has when initiating the protocol, and can therefore be provided to the simulator as well. While this is the intuition behind the simulation argument, there is another technical twist at this point. In the 3DH simulation of Bob against a malicious Alice, the simulator is allowed to choose $y \leftarrow \mathbb{Z}_q$ and set $Y = g^y$; the knowledge of y helps the simulator in the computation of the correct key. In the X3DH simulation, however, Y is part of the auxiliary input and the simulator has no access to y. Intuitively, because Bob signs Y, the latter

can be seen as another public key associated with him and the simulator cannot be given its secret key.

The problem boils down to the computation of g^{xy}. In the 3DH simulation, we simply computed it through the knowledge of y. Here, we need to extract it from Alice, and this requires an additional assumption that says we can extract *both* g^{xy} and g^{bx}.

Definition 9. *Let G be a cyclic group and AUX a class of auxiliary inputs. Let M be a probabilistic Turing Machine running in time T_M which runs on input (U, W, aux) where $U, W \in_R G$, and $aux \in AUX$, and outputs $Z \in G$; we denote with $Z = M(U, W, aux, r)$ the output of running M on input U, W, aux with coin tosses r.*

We say that the $(T_M, T_{\hat{M}}, T_C, \varepsilon_C)$ Knowledge of 2DH (K2DH) Assumption *holds over group G and class AUX, if for every such M, there exists a companion probabilistic Turing Machine $\hat{M}$ (called the extractor for M) such that: $\hat{M}$ runs on input U, W, aux, r in time $T_{\hat{M}}$ and outputs $\hat{Z}_1, \hat{Z}_2 \in G$ or $\perp$ such that*

- *If $\hat{M}(U, aux, r) \neq \perp$ then $\hat{Z}_1 = DH(U, Z)$ and $\hat{Z}_2 = DH(W, Z)$*
- *If $\hat{M}(U, aux, r) = \perp$ then for every probabilistic Turing Machine C running in time T_C we have that*

$$Prob[C(U, Z, r, aux) \in \{DH(U, Z), DH(W, Z)\}] \leq \varepsilon_C$$

where the probabilities are taken over $U \leftarrow G$, W, r and the coin tosses of $\hat{M}$ and C, conditioned on $\hat{M}(U, aux, r) = \perp$.

Theorem 6. *Under the $(T_M, T_{\hat{M}}, T_C, \varepsilon_C)$ K2DHA, X3DH with Key Registration is a $(\tilde{T}_M, \tilde{T}_{\hat{M}}, \tilde{T}_C, \varepsilon_C)$ deniable AKE in the random oracle model.*

10 On the Need to Extract the Long-Term Private Keys

The simulation arguments of Theorems 4 and 6 assume the ability to extract the incriminating party's (Bob in our examples) private key, for example via key registration. This simplified the proofs and intuition. We note, however, that such extraction is not needed. Instead, we can generalize our extraction assumptions to prevent Bob from sampling either B or Y in a way that he does not know the discrete logs and yet *both* g^{ab} *and* g^{ay} are distinguishable from random. Indeed, what happens (in either HMQV, 3DH and X3DH) is that Bob will be able to incriminate Alice if (and only if) he is able to sample either B or Y under the above conditions. Formally, we achieve this by adding one extra "knowledge" assumption about the way parties generate their long-term keys; arguably, this additional assumption is not essentially stronger than the previous ones. Details follow.

Definition 10. *Let G be a cyclic group and AUX a class of auxiliary inputs. Let M be a probabilistic Turing Machine running in time T_M which runs on input*

$aux \in AUX$, and outputs $Z \in G$; we denote with $Z = M(aux, r)$ the output of running M on input aux with coin tosses r.

We say that the $(T_M, T_{\hat{M}}, T_C, \varepsilon_C)$ Extended Knowledge of DH (EKDH) Assumption *holds over group G and class AUX, if for every such M, there exists a companion probabilistic Turing Machine $\hat{M}$ (called the extractor for M) such that: $\hat{M}$ runs on on input aux, r and an additional input $U \in G$, in time $T_{\hat{M}}$ and outputs $\hat{Z}$ or $\perp$ such that*

- *If $\hat{M}(U, aux, r) = \hat{Z} \neq \perp$ then $\hat{Z} = DH(U, Z)$*
- *If $\hat{M}(U, aux, r) = \perp$ then for every probabilistic Turing Machine C running in time T_C we have that*

$$Prob[C(U, Z, r, aux) = DH(U, Z)] \leq \varepsilon_C$$

where the probabilities are taken over r and the coin tosses of $\hat{M}$ and C, conditioned on $\hat{M}(U, aux, r) = \perp$.

Note the difference between the KDH and the EKDH Assumption. In the latter, the group element $U \in G$ is not known to the machine M, but is fed to the extractor as input. This is because we need to model a machine M that generates Z as its long-term public key *before* seeing any of the keys of the parties it will interact with.

Theorem 7. *Under the $(T_M, T_{\hat{M}}, T_C, \varepsilon_C)$ EKDHA, protocols HMQV is $(\tilde{T}_M, \tilde{T}_{\hat{M}}, \tilde{T}_C, \varepsilon_C)$ deniable AKE in the random oracle model, even without registration of long-term public keys.*

A similar theorem holds for X3DH.

Acknowledgment. The authors thank the anonymous reviewer whose excellent comments greatly improved the presentation of this paper.

References

1. Alwen, J., Coretti, S., Dodis, Y.: The double ratchet: security notions, proofs, and modularization for the signal protocol. IACR Cryptology ePrint Archive **2018**, 1037 (2018)
2. Bellare, M., Palacio, A.: The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 273–289. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_17
3. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_21
4. Borisov, N., Goldberg, I., Brewer, E.: Off-the-record communication, or, why not to use PGP. In Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society WPES 2004, pp. 77–84. ACM, New York (2004)

5. Boyd, C., Mao, W., Paterson, K.G.: Key agreement using statically keyed authenticators. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 248–262. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24852-1_18
6. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44987-6_28
7. Canetti, R., Krawczyk, H.: Security analysis of IKE's signature-based key-exchange protocol. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 143–161. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_10
8. Canetti, R., Krawczyk, H.: Universally composable notions of key exchange and secure channels. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 337–351. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_22
9. Cohn-Gordon, K., Cremers, C., Dowling, B., Garratt, L., Stebila, D.: A formal security analysis of the signal messaging protocol. In: 2017 IEEE European Symposium on Security and Privacy (EuroS P), pp. 451–466, April 2017
10. Cremers, C., Feltz, M.: One-round strongly secure key exchange with perfect forward secrecy and deniability. Cryptology ePrint Archive, Report 2011/300 (2011). https://eprint.iacr.org/2011/300
11. Dent, A.W., Galbraith, S.D.: Hidden pairings and trapdoor DDH groups. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 436–451. Springer, Heidelberg (2006). https://doi.org/10.1007/11792086_31
12. Di Raimondo, M., Gennaro, R.: New approaches for deniable authentication. J. Cryptol. **22**(4), 572–615 (2009). https://doi.org/10.1007/s00145-009-9044-3
13. Di Raimondo, M., Gennaro, R., Krawczyk, H.: Secure off-the-record messaging. In: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society WPES 2005, pp. 81–89. ACM, New York (2005)
14. Di Raimondo, M., Gennaro, R., Krawczyk, H.: Deniable authentication and key exchange. In: Proceedings of the 13th ACM Conference on Computer and Communications Security CCS 2006, pp. 400–409. ACM, New York (2006)
15. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theor. **22**(6), 644–654 (2006)
16. Dodis, Y., Katz, J., Smith, A., Walfish, S.: Composability and on-line deniability of authentication. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 146–162. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_10
17. Dwork, C., Naor, M., Sahai, A.: Concurrent zero-knowledge. In: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing STOC 1998, pp. 409–418. ACM, New York (1998)
18. Fischlin, M., Mazaheri, S.: Notions of deniable message authentication. In: Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society WPES 2015, pp. 55–64. ACM, New York (2015)
19. Goldwasser, S., Micali, S.: Probabilistic encryption. JCSS **28**(2), 270–299 (1984)
20. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM J. Comput. **18**(1), 186–208 (1989)
21. Harkins, D., Carrel, D.: The internet key exchange (IKE). RFC 2409, RFC Editor, November 1998
22. Harkins, D., Carrel, D.: The internet key exchange (IKE) (1998)
23. Katz, J.: Efficient and non-malleable proofs of plaintext knowledge and applications. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 211–228. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_13

24. Kaufman, C.: Internet key exchange (IKEv2) protocol. RFC 4306, RFC Editor, December 2005
25. Krawczyk, H.: Skeme: a versatile secure key exchange mechanism for internet. In: Proceedings of Internet Society Symposium on Network and Distributed Systems Security, pp. 114–127, February 1996
26. Krawczyk, H.: HMQV: a high-performance secure Diffie-Hellman protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_33
27. Law, L., Menezes, A., Qu, M., Solinas, J., Vanstone, S.: An efficient protocol for authenticated key agreement. Des. Codes Crypt. **28**(2), 119–134 (2003)
28. Mao, W., Paterson, K.: On the plausible deniability feature of internet protocols. Manuscript (2002)
29. Marlinspike, M.: Simplifying OTR deniability (2013). https://signal.org/blog/simplifying-otr-deniability/
30. Marlinspike, M., Perrin, T.: The x3dh key agreement protocol, Rev. 1, November 2016
31. Menezes, A., Qu, M., Vanstone, S.: Some new key agreement protocols providing implicit authentication. In: Workshop on Selected Area in Cryptography (SAC 1995), pp. 22–32 (1995)
32. Naor, M.: Deniable ring authentication. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 481–498. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_31
33. Pass, R.: On deniability in the common reference string and random oracle model. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 316–337. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_19
34. Perrin, T., Marlinspike, M.: The double ratchet algorithm, Rev. 1, November 2016
35. Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991). https://doi.org/10.1007/BF00196725
36. Seurin, Y.: New constructions and applications of trapdoor DDH groups. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 443–460. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36362-7_27
37. Shoup, V.: On formal models for secure key exchange. Technical report RZ 3120, IBM, April 1999
38. Signal technical information. https://signal.org/docs/
39. Unger, N., Goldberg, I.: Deniable key exchanges for secure messaging. In: Proceedings on 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 1211–1223 (2015)
40. Unger, N., Goldberg, I.: Improved strongly deniable authenticated key exchanges for secure messaging. Proc. Priv. Enhancing Technol. **2018**(1), 21–66 (2018)
41. Vatandas, N., Gennaro, R., Ithurburn, B., Krawczyk, H.: On the deniability of signal communications. Cryptology ePrint Archive (2020). https://eprint.iacr.org/
42. Walfish, S.: Enhanced security models for network protocols. Ph.D thesis (2008)
43. Yao, A.C., Zhao, Y.: Deniable internet key exchange. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 329–348. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13708-2_20
44. Yao, A.C., Zhao, Y., OAKE: a new family of implicitly authenticated diffie-hellman protocols. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security CCS 2013, pp. 1113–1128. ACM, New York (2013)