Practical Adaptive Oblivious Transfer from Simple Assumptions

Matthew Green* — Johns Hopkins University
Susan Hohenberger† — Johns Hopkins University

January 7, 2011

Abstract

In an adaptive oblivious transfer (OT) protocol, a sender commits to a database of messages and then repeatedly interacts with a receiver in such a way that the receiver obtains one message per interaction of his choice (and nothing more) while the sender learns nothing about any of the choices. Recently, there has been significant effort to design practical adaptive OT schemes and to use these protocols as a building block for larger database applications. To be well suited for these applications, the underlying OT protocol should: (1) support an efficient initialization phase where *one* commitment can support an arbitrary number of receivers who are guaranteed of having the same view of the database, (2) execute transfers in time independent of the size of the database, and (3) satisfy a strong notion of security under a simple assumption in the standard model.

We present the first adaptive OT protocol simultaneously satisfying these requirements. The sole complexity assumption required is that given (g, g^a, g^b, g^c, Q), where g generates a bilinear group of prime order p and a, b, c are selected randomly from $\mathbb{Z}_p$, it is hard to decide if $Q = g^{abc}$. All prior protocols in the standard model either do not meet our efficiency requirements or require dynamic "q-based" assumptions.

Our construction makes an important change to the established "assisted decryption" technique for designing adaptive OT. As in prior works, the sender commits to a database of n messages by publishing an encryption of each message and a signature on each encryption. Then, each transfer phase can be executed in time *independent* of n as the receiver blinds one of the encryptions and proves knowledge of the blinding factors and a signature on this encryption, after which the sender helps the receiver decrypt the chosen ciphertext. One of the main obstacles to designing an adaptive OT scheme from a simple assumption is realizing a suitable signature for this purpose (i.e., enabling signatures on group elements in a manner that later allows for efficient proofs.) We make the observation that a secure signature scheme is not necessary for this paradigm, provided that signatures can only be forged in certain ways. We then show how to efficiently integrate an insecure signature into a secure adaptive OT construction.

1 Introduction

Oblivious transfer OT [40, 44] is a two-party protocol, where a Sender with messages $M_1, \ldots, M_N$ and a Receiver with indices $\sigma_1, \ldots, \sigma_k \in [1, N]$ interact in such a way that at the end the Receiver obtains $M_{\sigma_1}, \ldots, M_{\sigma_k}$ without learning anything about the other messages and the Sender does

*Supported by NSF CNS-1010928. This publication was also made possible by Grant Number HHS 90TR0003/01. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the HHS.

†Supported by NSF CAREER CNS-1053886, a Microsoft New Faculty Fellowship and a Google Research Award.

not learn anything about the choices $\sigma_1, \ldots, \sigma_k$. In the *adaptive* OT setting [37], the Receiver may obtain $M_{\sigma_{i-1}}$ before deciding on σ_i [37].

Our Goals. Adaptive OT is an interesting primitive. Like non-adaptive OT, it is a key building block for secure multi-party computation [45, 20, 30]. More practically, it captures the way an oblivious medical, financial or patent database would be accessed. Recently, there has been a focus on designing practical, privacy-preserving databases with access controls [15, 8] or pricing mechanisms [41] based on adaptive OT. Unfortunately, researchers trying to design more-complex systems on top of current adaptive OT protocols do not have any ideal choices. For a database with N messages supporting U Receivers with security parameter λ, such a protocol must be:

1. Extremely efficient, even when N, the database size, is large. In particular, the cost to transfer one message to one Receiver should depend only on the security parameter and not on N. I.e., a Receiver should not have to do work proportional to the size of the database to download one file. (This rules out a number of naive approaches as discussed below.)

2. Furthermore, since few databases serve only one user, it should be possible to extend the protocol to the case where there are *many* Receivers, each of whom receives a consistent view of the database. In particular, the ideal situation, which we achieve in this work, is to have a *non-interactive* initialization phase, where the Sender can do $O(\lambda N)$ work to form a commitment that can then be used for an arbitrary number of receivers. Several prior works (e.g., [10, 23, 29, 41]) support a relatively efficient initialization phase with $O(\lambda(N+U))$ total work. By adding a CRS and making some modifications, this can likely be reduced to $O(\lambda N)$ (although the complexity assumptions will still be an issue.) What one wishes to avoid, however, is an initialization phase that requires $O(\lambda NU)$ total work. I.e., the sender should not have to set up a *unique* database containing *all* of its files for *each* of its users. (This also rules out some basic approaches.)

3. Finally, since this protocol is designed to be a building block of larger applications, it is critical that it be a solid one. In particular, it should satisfy a strong notion of security (i.e., full-simulatability or UC) under a mild complexity assumption in the standard model. Unfortunately, while sufficiently practical protocols exist, they either require random oracles [10, 23], dynamic[1] assumptions [10, 23, 29, 41] or interactive assumptions [43].

Thus, a new construction based on new techniques is needed.

From Non-Adaptive to Adaptive OT for Single Receivers. Since it is known how to build non-adaptive OT protocols based on simple assumptions [22, 36, 39] such as Decisional Diffie-Hellman and Quadratic Residuosity, it is natural to ask why constructing adaptive protocols has proven so difficult. Given any fully-simulatable 1-out-of-N non-adaptive OT protocol, one can build a fully-simulatable k-out-of-N adaptive OT protocol for a *single* Receiver by sequentially executing k instances of the non-adaptive protocol and, before each execution, having the sender

[1]These are also called *parametric* or *q-based* assumptions. An example is q-Strong Diffie-Hellman [3] (q-SDH): given $(g, g^x, g^{x^2}, g^{x^3}, \ldots, g^{x^q})$, where g generates a group of prime order p and x is a random value in $\mathbb{Z}_p$, it is hard to compute $(g^{1/(x+c)}, c)$ for any $c \in \mathbb{Z}_p^*$. Typically, when q-SDH is used as the foundation of an adaptive OT scheme, q must dynamically adjust to the number of files in the database. Thus, the assumption required actually changes based on how the protocol is used.

Protocol	Initialization Cost	Transfer Cost	Assumption	Security Defn
Folklore	$\cdot$	$O(\lambda N)$	general assumptions	Full Sim
KN [32]	$O(\lambda(N+U))$	$O(\lambda N)$	Decisional nth Residuosity/DDH	Full Sim
NP [37]	$\cdot$	$O(\lambda \lg(N))$	DDH + OT_1^2	Half Sim
KNP [33]	$O(\lambda NU)$	$O(\lambda)$	DDH	Full Sim*
CNS [10]	$O(\lambda(N+U))$	$O(\lambda)$	q-Power DDH + q-Strong DH	Full Sim
GH [23]	$O(\lambda(N+U))$	$O(\lambda)$	Decision Linear + q-Hidden LRSW	UC
JL [29]	$O(\lambda(N+U))$	$O(\lambda)$	Comp. Dec. Residuosity + q-DDHI	Full Sim
RKP [41]	$O(\lambda(N+U))$	$O(\lambda)$	DLIN + q-Hidden SDH + q-TDH	UC
§3.2	$O(\lambda(N+U))$	$O(\lambda)$	Decision 3-Party DH	Full Sim
§4	$O(\lambda N)$	$O(\lambda)$	Decision 3-Party DH + DLIN	Full Sim

Figure 1: Survey of adaptive k-out-of-N Oblivious Transfer protocols secure in the standard model. Let λ be the security parameter, N the size of the database and U the number of receivers. The horizontal lines separate the schemes into efficiency categories, which improve as one scans down the table. While the least efficient categories can be realized using assumptions such as DDH, all prior attempts to achieve practicality have required a dynamic q-based complexity assumption. A $*$ denotes the construction meets a strictly weaker notion than the standard used in the other works.

prove in zero-knowledge that the sequence of N messages used in execution i is the same as the sequence of N messages used in execution $i-1$ [10]. Unfortunately, for security parameter λ, this protocol requires a total of $O(Nk\lambda)$ work to transfer k messages for (only) *one* Receiver and is thus impractical for any application involving large databases.

Thus, when Camenisch, Neven and shelat [10] began to reinvestigate this problem in 2007, they stressed that the real challenge was to build an OT scheme where the sender makes an initial commitment to the database (which is assumed to be broadcast to all receivers), and then the two parties only exchange a *constant number* of group elements per transfer.

Our Contributions. We present an efficient, adaptive oblivious transfer protocol which is fully-simulatable under a simple, static assumption. The sole complexity assumption required is that given (g, g^a, g^b, g^c, Q), where g generates a bilinear group of prime order p and a, b, c are selected randomly from $\mathbb{Z}_p$, it is hard to decide if $Q = g^{abc}$. This assumption called *Decisional 3-Party Diffie-Hellman* has been used in prior works [34, 5, 27]. Our protocol is practical, although more costly than the very efficient Camenisch et al. protocol [10] by a constant factor. The database commitment in our scheme requires roughly $(9 + 7N)$ group elements, whereas the commitment in [10] required roughly $(3 + 2N)$ group elements. By including the mild Decision Linear assumption [4], we can efficiently make this initialization phase *non-interactive* as we discuss in Section 4.

Our construction introduces a twist on the *assisted decryption* approach to OT design, where the underlying signatures need not be existentially unforgeable provided that certain forgeries are not permitted. As we discuss, these techniques may be useful in simplifying the complexity assumptions in schemes beyond OT such as F-signatures and anonymous credentials [1].

Intuition behind our $\mathsf{OT}_{k\times 1}^N$ Construction. As with most previous $\mathsf{OT}_{k\times 1}^N$ constructions, our construction uses a technique known as *assisted decryption*. For $i = 1$ to N, the Sender commits to his database by encrypting each message as $C_i = \text{Enc}(M_i)$, and publishes a public key and ciphertexts $(pk, C_1, \ldots, C_N)$. The Receiver then checks that each ciphertext is well-formed. To obtain a message, the Sender and Receiver engage in a *blind* decryption protocol, i.e., an interactive protocol in which the Sender does not view the ciphertext he decrypts, but where the Receiver is convinced that decryption was done correctly.

The difficulty here is to prevent the Receiver from abusing the decryption protocol, e.g., by requesting decryptions of ciphertexts which were either not produced by the Sender or have been mauled. The folklore solution is to have the Receiver provide a proof that his request corresponds to $C_1 \vee C_2 \vee ... \vee C_N$. Of course, the cost of each transfer is now dependent on the total database size and thus this solution is no (asymptotically) better than the trivial solution mentioned above.

In Eurocrypt 2007, Camenisch, Neven and shelat [10] were the first to propose a method for executing "assisted decryption" efficiently. The sender signed each ciphertext value. The receiver was required to prove knowledge of a corresponding signature before the sender would assist him in decrypting a ciphertext. This clever approach reduced the $O(N\lambda)$ work per transfer required above, to only $O(\lambda)$ work, where λ is a security parameter.

More specifically, Camenisch, Neven and shelat [10] used a deterministic encryption scheme and a signature with a particular structure: for $pk = (g, g^x, H = e(g, h))$ and $sk = h$, let $C_i = \left(g^{\frac{1}{x+i}}, M_i \cdot e(g, h)^{\frac{1}{x+i}}\right)$. Recall that $g^{1/(x+i)}$ is a weak Boneh-Boyen signature [3] on i under g^x, and here only a polynomial number of "messages" (1 to N) are signed. While this scheme supports an elegant and efficient blind decryption protocol, it also requires strong q-based assumptions for both the indistinguishability of the ciphertexts as well as the unforgeability of the weak Boneh-Boyen signature. It is based on the q-Strong Diffie-Hellman and the q-Power Decisional Diffie-Hellman assumptions. The latter assumption states that given $(g, g^x, g^{x^2}, \ldots, g^{x^q}, H)$, where $g \in \mathbb{G}$ and $H \in \mathbb{G}_T$, it is hard to distinguish the vector of elements $(H^x, H^{x^2}, \ldots, H^{x^q})$ from a vector of random elements in $\mathbb{G}_T$. In essence, the rigid structure of this (and all prior) constructions appear to require a similarly structured complexity assumption, which grows with the database size.

To move past this, we will "loosen" the structure of the ciphertext and signature enough to break the dependence on a structured assumption, but not so much as to ruin our ability to perform efficient proofs. Finding this balance proved highly non-trivial.

We now turn to how our construction works. We will encrypt using the Boneh-Boyen IBE [2], which has a public key $pk = (g, g_1 = g^a, g_2, h)$ and encrypts M as $(g^r, (g_1^i h)^r, e(g_1, g_2)^r M)$ for identity i and randomness $r \in \mathbb{Z}_p$. Then we will sign r. To do this, we need a standard model signature scheme from a simple assumption (which is itself somewhat rare.) We choose the *stateful* signatures of Hohenberger-Waters [28], which has a public key $pk = (g, g^b, u, v, d, w, z, h)$ and signs M as $(\sigma_1, \sigma_2, s, i)$ for state i and randomness $s, t \in \mathbb{Z}_p$, where $\sigma_1 = g^t$, $\sigma_2 = (u^M v^s d)^b (w^{\lceil \lg(i) \rceil} z^i h)^t$.

Attempt 1. Now, consider the construction obtained by combining the BB IBE, secure under Decisional Bilinear Diffie-Hellman, with the HW signature, secure under the Computational Diffie-Hellman assumption. Here we will encrypt the ith message using identity i (in the BB IBE) and state i (in the HW signature), with an extra u^r term to allow the Receiver to verify well-formedness:

$$g^r, \quad (g_1^i h)^r, \quad e(g_1, g_2)^r M, \quad g^t, \quad (u^r v^s d)^b (w^{\lceil \lg(i) \rceil} z^i h)^t, \quad u^r, \quad s$$

The Receiver can verify the well-formedness of the ith ciphertext $(c_1, \ldots, c_7)$ by checking that $e((g_1^i h), c_1) = e(g, c_2)$, $e(g, c_6) = e(c_1, u)$ and

$$e(g, c_5) = e(c_6 v^{c_7} d, g^b) e(w^{\lceil \lg(i) \rceil} z^i h, c_4).$$

It is important that the Receiver can verify the well-formedness of the ciphertext-signature pair, so that the simulator can properly extract the messages from a cheating Sender during the proof of security. It is a nice additional feature that our verification is public and non-interactive.

Attempt 2. However, the above construction still has a lot of problems. Recall that we want the Receiver to ask for a blind decryption of a given ciphertext by (somehow) sending in blinded portions of the ciphertext, proving that these portions are linked to r and proving that he knows a signature on r. Unfortunately, efficiently proving knowledge of the HW signature is problematic due to the $\lceil \lg(i) \rceil$ exponent. We could do this using a range proof [13, 9, 6, 7], however, this would require that we introduce stronger assumptions such as Strong RSA or q-Strong Diffie-Hellman. We could instead do a bit-by-bit proof, but this would severely hurt our efficiency. Instead, our solution is to drop this term entirely from the HW signature to obtain the ciphertext:

$$g^r, \quad (g_1^i h)^r, \quad e(g_1, g_2)^r M, \quad g^t, \quad (u^r v^s d)^b (z^i h)^t, \quad u^r, \quad s$$

One major issue is that dropping this term breaks the unforgeability of the signature scheme. Indeed, it is now possible for anyone to efficiently compute a signature on any index over a certain polynomial threshold as set in the proof of security. However, we specifically chose to encrypt with the Boneh-Boyen IBE for this purpose. We will set our parameters so that an adversary is free to forge signatures with states of $N+1$ and higher, where N is the size of our database. The key idea is that asking for decryptions on *different identities* will not help a malicious Receiver obtain information about the database messages; indeed, we could even hand him the secret key for those identities. This makes our proof much more efficient, however, there is still a large problem.

Attempt 3. To argue, in the proof of security, that no malicious Receiver can forge signatures on a state $i \in [1, N]$, we must *extract* this signature and its forgery message from the proof of knowledge. However, we cannot extract the "message" r from a cheating Receiver, because an honest Receiver will not know the randomness used in the ciphertexts created by the Sender. The most we can ask a Receiver to prove knowledge of is the signature on r comprised of (c_4, c_5, c_6, c_7) and the value g^r. Thus, we cannot extract from the Receiver a valid forgery of the HW signatures.

Moreover, we need a stronger security guarantee than HW signatures gave us (i.e., existential unforgeability under adaptive chosen message attack [21].) We need that: it is not only the case that an adversary cannot produce a pair (m, σ) for a new m; now the adversary cannot even produce the pair (g^m, σ) for a new m, where σ is a signature on m. Do such powerful signatures exist?

Indeed, this security notion was formalized as F-signatures by Belenkiy, Chase, Kohlweiss and Lysyanskaya [1], where they also required q-based complexity assumptions for their construction. Fortunately, we are able to show that the HW signatures (and our mangled version of them without the $w^{\lceil \lg(i) \rceil}$ term) remain F-unforgeable for $F(m) = g^m$ under a simple static assumption. (See Appendix D for the full details on HW; the mangled version is proven as part of the OT system in Section 3.3.) We tie both this version of the signature scheme and the Boneh-Boyen IBE together under a single assumption: given (g, g^a, g^b, g^c), it is hard to decide if $Q = g^{abc}$.

Comparison to Prior Work. Let us briefly compare our approach to prior works; see Figure 1 for more. As we mention above, Camenisch, Neven and shelat [10] gave the first efficient, fully-simulatable construction for adaptive (and non-adaptive) OT. It is secure in the standard model under the q-Strong Diffie-Hellman and the q-Power Decisional Diffie-Hellman assumptions. They also provided a scheme in the random oracle model from unique blind signatures.

Green and Hohenberger [22] provided an adaptive OT construction in the random oracle model based on the Decisional Bilinear Diffie-Hellman assumption, namely, that given (g, g^a, g^b, g^c, Q), it is hard to decide if $Q = e(g, g)^{abc}$. In their construction, the Sender encrypted each message i under identity i using a IBE system. Then they provided a blind key extraction protocol, where the Receiver could blindly obtain a secret key for any identity of her choice.

In the assisted decryption setting, Green and Hohenberger [23] took an approach similar to [10] to achieve UC security. It was based on the Decision Linear and q-Hidden LRSW assumptions, in the asymmetric setting. The latter assumption implies that DDH must hold in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

Jarecki and Liu [29] took an alternative view: for $pk = g^x$, let $C_i = M_i \cdot g^{1/(x+i)}$. Recall that $g^{1/(x+i)}$ is also the Dodis-Yampolskiy pseudorandom function on input i [18]. This essentially simplifies the Camenisch et al. construction and allows a fully-simulatable scheme based on the Composite Decisional Residuosity and q-Decisional Diffie-Hellman Inversion assumptions. The blind decryption protocol involves obliviously evaluating the PRF on input i, which requires some costly zero knowledge proofs. However, this protocol is interesting as the only efficient and fully-simulatable protocol that does not require bilinear groups.

Rial, Kohlweiss and Preneel [41] presented a *priced* version of UC-secure adaptive OT using the assisted decryption approach. In priced OT, the obliviousness property must hold, even though the items being sold may have unique prices. The scheme is secure in the standard model under the Decision Linear, q-Triple Diffie-Hellman, and q-Hidden Strong Diffie-Hellman assumptions.

Unfortunately, all of these constructions have a rigid structure and seem to require a structured complexity assumption. We show that this structure can be enforced, not on the message itself, but rather through the *identity* of the encryption and the *state* of the signature. This provides us with enough glue to keep the security of the scheme together without overdoing it.

Recently, Kurosawa and Nojima [32] and Chen, Chou and Hou [14] gave adaptive OT constructions which purported to improve the underlying complexity assumptions of the schemes above, but which actually resorted to $O(N\lambda)$ transfer cost. It was already known how to achieve this level of (in)efficiency from *general* assumptions, including those of [32, 14], by following the folklore method for building adaptive OT from any non-adaptive OT system, as described in [17, 10] and the opening of our introduction. Moreover, [14] is set in the random oracle model.

Very recently[2], Kurosawa, Nojima and Phong [33] gave an adaptive OT construction from DDH with $O(\lambda)$ transfers. However, their work has several technical issues. First, their construction does not satisfy the standard full simulation definition used in [10, 22, 23, 29, 41] and this work. In [33], if a receiver ever requests the same file twice (say, she downloads a patent one day, deletes it, then downloads it again a month later), then this can be detected by the sender. This is at odds with the full simulation definition where the adversarial sender is only told by the ideal functionality that a file has been requested and thus cannot detect a repeated download. Second, it is not obvious how to modify their construction to satisfy the full simulation notion. One approach might be to make the receiver stateful and store every file she ever requests. This has the obvious drawback of requiring permanent storage of the decrypted messages, which may not be practical and is not a requirement in other works. Moreover, subtle technical issues arise as to what the receiver sends during a repeated query round. Third, their construction requires a very expensive initialization procedure where the sender must transmit, then receive back and store $O(N\lambda)$ bits for *each* receiver. In contrast, all prior practical work [10, 22, 23, 29, 41] and our results only require that the sender publish and store *one* $O(N\lambda)$ bit database for *all* receivers.

[2]The work of [33] appeared after the initial posting of this work [24].

Thus, we build on this body of prior work to present the first efficient scheme satisfying the standard notion of full simulation from a simple assumption in the standard model.

2 Technical Preliminaries

Bilinear Groups. Let $\mathsf{BMsetup}$ be an algorithm that, on input 1^κ, outputs the parameters for a bilinear mapping as $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e)$, where g generates $\mathbb{G}$, the groups $\mathbb{G}, \mathbb{G}_T$ have prime order $p \in \Theta(2^\kappa)$, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Two algebraic properties required are that: (1) if g generates $\mathbb{G}$, then $e(g, g) \neq 1$ and (2) for all $a, b \in \mathbb{Z}_p$, it holds that $e(g^a, g^b) = e(g, g)^{ab}$.

Assumption 2.1 (Decisional 3-Party Diffie-Hellman (3DDH) [34, 5, 27]) *Let $\mathbb{G}$ be a group of prime order $p \in \Theta(2^\lambda)$. For all p.p.t. adversaries $\mathcal{A}$, the following probability is $1/2$ plus an amount negligible in λ:*

$$\Pr[g, z_0 \leftarrow \mathbb{G}; a, b, c \leftarrow \mathbb{Z}_p; z_1 \leftarrow g^{abc}; d \leftarrow \{0,1\}; d' \leftarrow \mathcal{A}(g, g^a, g^b, g^c, z_d) : d = d'].$$

Proofs of Knowledge. We use known zero-knowledge and witness indistinguishable techniques for proving statements about discrete logarithms and their natural extensions to proving statements about bilinear groups, such as (1) proof of knowledge of a discrete logarithm modulo a prime [42] and (2) proof of the disjunction or conjunction of any two statements [16]. These are typically interactive, 4-round protocols. We discuss further implementation details in Appendix C.

When referring to the proofs above, we will use the notation of Camenisch and Stadler [11]. For instance, $ZKPoK\{(x, h) : y = g^x \wedge H = e(y, h) \wedge (1 \leq x \leq n)\}$ denotes a zero-knowledge proof of knowledge of an integer x and a group element $h \in \mathbb{G}$ such that $y = g^x$ and $H = e(y, h)$ holds and $1 \leq x \leq n$. All values not enclosed in ()'s are assumed to be known to the verifier.

3 Adaptive Oblivious Transfer from a Simple Assumption

Adaptive Oblivious Transfer ($\mathsf{OT}^N_{k\times 1}$) is traditionally defined as a protocol conducted by a Sender and a single Receiver. In the following section we will formally define the protocol and its security requirements. As noted above, a primary application of $\mathsf{OT}^N_{k\times 1}$ is the construction of multi-user oblivious databases, and thus we must also consider the implications of a protocol involving $U \geq 1$ distinct Receivers. In Appendix B, we present an alternative definition that captures this notion and describes the security and consistency properties involved in such an interaction.[3]

3.1 Definition of Adaptive k-out-of-N Oblivious Transfer ($\mathsf{OT}^N_{k\times 1}$) [37, 10]

An oblivious transfer scheme is a tuple of algorithms $(\mathsf{S_I}, \mathsf{R_I}, \mathsf{S_T}, \mathsf{R_T})$. During the initialization phase, the Sender and the Receiver conduct an interactive protocol, where the Sender runs $\mathsf{S_I}(M_1, \ldots, M_N)$ to obtain state value S_0, and the Receiver runs $\mathsf{R_I}()$ to obtain state value R_0. Next, for $1 \leq i \leq k$, the i^{th} transfer proceeds as follows: the Sender runs $\mathsf{S_T}(S_{i-1})$ to obtain state value S_i, and the Receiver runs $\mathsf{R_T}(R_{i-1}, \sigma_i)$ where $1 \leq \sigma_i \leq N$ is the index of the message to be received. The receiver obtains state information R_i and the message M'_{σ_i} or $\perp$ indicating failure.

[3] Indeed, a multi-receiver definition is necessary to achieve consistency with the oblivious access control schemes of Coull et al. [15] and Camenisch et al. [8].

Definition 3.1 (Full Simulation Security.) Consider the following experiments.[4]

Real experiment. In experiment $\mathbf{Real}_{\hat{\mathsf{S}},\hat{\mathsf{R}}}(N, k, M_1, \ldots, M_N, \Sigma)$, the possibly cheating sender $\hat{\mathsf{S}}$ is given messages $(M_1, \ldots, M_N)$ as input and interacts with the possibly cheating receiver $\hat{\mathsf{R}}(\Sigma)$, where Σ is a selection algorithm that on input the full collection of messages thus far received, outputs the index σ_i of the next message to be queried. At the beginning of the experiment, both $\hat{\mathsf{S}}$ and $\hat{\mathsf{R}}$ output initial states (S_0, R_0). In the transfer phase, for $1 \le i \le k$ the sender computes $S_i \leftarrow \hat{\mathsf{S}}(S_{i-1})$, and the receiver computes $(R_i, M'_i) \leftarrow \hat{\mathsf{R}}(R_{i-1})$, where M'_i may or may not be equal to M_i. At the end of the k^{th} transfer the output of the experiment is (S_k, R_k).

We define the *honest* Sender S as one that runs $\mathsf{S_I}(M_1, \ldots, M_N)$ in the first phase, during each transfer runs $\mathsf{S_T}()$ and outputs $S_k = \varepsilon$ as its final output. The *honest* Receiver R runs $\mathsf{R_I}$ in the first phase, and $\mathsf{R_T}(R_{i-1}, \sigma_i)$ at the i^{th} transfer, and outputs $R_k = (M'_{\sigma_1}, \ldots, M'_{\sigma_k})$ as its final output.

Ideal experiment. In experiment $\mathbf{Ideal}_{\hat{\mathsf{S}}',\hat{\mathsf{R}}'}(N, k, M_1, \ldots, M_N, \Sigma)$ the possibly cheating sender algorithm $\hat{\mathsf{S}}'$ generates messages $(M^*_1, \ldots, M^*_N)$ and transmits them to a trusted party T. In the i^{th} round $\hat{\mathsf{S}}'$ sends a bit b_i to T; the possibly cheating receiver $\hat{\mathsf{R}}'(\Sigma)$ transmits σ^*_i to T. If $b_i = 1$ and $\sigma^*_i \in \{1, \ldots, N\}$ then T hands $M^*_{\sigma_i{}^*}$ to $\hat{\mathsf{R}}'$. If $b_i = 0$ then T hands $\perp$ to $\hat{\mathsf{R}}'$. After the k^{th} transfer the output of the experiment is (S_k, R_k).

Let $\ell(\cdot)$ be a polynomially-bounded function. We now define Sender and Receiver security.

Sender Security. An $\mathsf{OT}^N_{k\times 1}$ provides Sender security if for every real-world p.p.t. receiver $\hat{\mathsf{R}}$ there exists a p.p.t. ideal-world receiver $\hat{\mathsf{R}}'$ such that $\forall N = \ell(\kappa)$, $k \in [1, N]$, $(M_1, \ldots, M_N)$, Σ, and every p.p.t. distinguisher:

$$\mathbf{Real}_{\mathsf{S},\hat{\mathsf{R}}}(N, k, M_1, \ldots, M_N, \Sigma) \stackrel{c}{\approx} \mathbf{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}(N, k, M_1, \ldots, M_N, \Sigma).$$

Receiver Security. $\mathsf{OT}^N_{k\times 1}$ provides Receiver security if for every real-world p.p.t. sender $\hat{\mathsf{S}}$ there exists a p.p.t. ideal-world sender $\hat{\mathsf{S}}'$ such that $\forall N = \ell(\kappa)$, $k \in [1, N]$, $(M_1, \ldots, M_N)$, Σ, and every p.p.t. distinguisher:

$$\mathbf{Real}_{\hat{\mathsf{S}},\mathsf{R}}(N, k, M_1, \ldots, M_N, \Sigma) \stackrel{c}{\approx} \mathbf{Ideal}_{\hat{\mathsf{S}}',\mathsf{R}'}(N, k, M_1, \ldots, M_N, \Sigma).$$

3.2 The Construction

Our $\mathsf{OT}^N_{k\times 1}$ protocol appears in Figure 2. This protocol follows the *assisted (or blind) decryption* paradigm pioneered by [10, 23, 29]. The Sender begins the OT protocol by encrypting each message in the database and publishing these values to the Receiver. The Receiver then checks that each ciphertext is well-formed. For each of k transfers, the two parties co-operatively execute a protocol following which (1) the Receiver obtains the decryption of at most one ciphertext, while (2) the Sender learns nothing about *which* ciphertext was decrypted. We require that the interactive decryption protocol run in time independent of the size of the database.

[4] As in [10], we do not explicitly specify auxiliary input to the parties; this information can (and indeed must) be provided in order to achieve sequential composition.

The encryption scheme that we use is a novel combination of the Boneh-Boyen IBE scheme [2] and a (insecure) version of the Hohenberger-Waters signatures [28]. We present methods for proving knowledge of such signatures and obtaining a blind decryption. Of course, in an adaptive OT scheme, the difficulty is always in getting all elements of the fully-simulatable proof of security to work out. There are many subtle details in basing the security for any database of size N under the one simple assumption that given (g, g^a, g^b, g^c), it is hard to decide if $Q = g^{abc}$.

Ciphertext Structure. In Figure 2, we reference a $\mathsf{VerifyCiphertext}$ algorithm for verifying the well-formedness of a ciphertext. Let us explain that now. The Sender's public parameters pk include $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e)$ and generators $(g_1, g_2,\ h, g_3, g_4, u, v, d) \in \mathbb{G}^8$. For message $M \in \mathbb{G}_T$, identity $j \in \mathbb{Z}_p$, and random values $r, s, t \in \mathbb{Z}_p$ we can express a ciphertext as:

$$C = \Big(g^r,\ (g_1^j h)^r,\ M \cdot e(g_1, g_2)^r,\ g^t,\ (u^r v^s d)^b (g_3^j h)^t,\ u^r,\ s\Big)$$

Given only pk, j, the $\mathsf{VerifyCiphertext}$ function validates that the ciphertext has this structure. We define it as follows.

$\mathsf{VerifyCiphertext}(pk, C, j)$. Parse C as $(c_1, \ldots, c_7)$ and pk to obtain $g, g_1, h, g_3, g_4,\ u, v, d$. This routine outputs 1 if and only if the following equalities hold:

$$\begin{aligned} e(g_1^j h, c_1) &= e\,(g, c_2)\ \wedge \\ e\,(g, c_6) &= e\,(c_1, u)\ \wedge \\ e\,(g, c_5) &= e\,(g_4, c_6 v^{c_7} d)\, e(c_4, g_3^j h) \end{aligned}$$

This function will always output 1 when input a properly-formed ciphertext.

3.3 Security Analysis

We now show that the $\mathsf{OT}^N_{k\times 1}$ protocol above is sender-secure and receiver-secure in the full-simulation model under the Decisional 3-Party Diffie-Hellman assumption (3DDH). We will address Sender and Receiver security separately.

A note on the PoK protocols. For generality, our security proofs use the terms $\epsilon_{ZK}, \epsilon_{WI}$ to indicate the maximal advantage that every p.p.t. distinguisher has in distinguishing simulated ZKPoKs from real ones (*resp.* WI proofs on different witnesses). We additionally use ϵ_{Ext} to indicate the maximum probability that the extractor for a PoK fails (soundness). We propose to use four-round Schnorr proofs which are zero-knowledge/WI ($\epsilon_{WI} = \epsilon_{ZK} = 0$) and computationally sound under the Discrete Logarithm assumption (which is naturally implied by 3DDH). Our security proofs employ the knowledge extractors for these proofs-of-knowledge, which we will define as $\mathsf{E}_1, \mathsf{E}_2, \mathsf{E}_3$.[5]

Sender security. Given a (possibly cheating) real-world receiver $\hat{\mathsf{R}}$, we show how to construct an ideal-world receiver $\hat{\mathsf{R}}'$ such that all p.p.t. distinguishers have at most negligible advantage in distinguishing the distribution of an honest real-world sender S interacting with $\hat{\mathsf{R}}$ ($\mathbf{Real}_{\mathsf{S},\hat{\mathsf{R}}}$) from that of $\hat{\mathsf{R}}'$ interacting with the honest ideal-world sender S' ($\mathbf{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}$). Let us now describe the operation of $\hat{\mathsf{R}}'$, which runs $\hat{\mathsf{R}}$ internally, interacting with it in the role of the Sender:

[5] These correspond respectively to the proofs $ZKPoK\{(a) : g_1 = g^a\}$, $WIPoK\{(\sigma_i, x, y, z, c_4, c_5, c_6, c_7) : \ldots\}$, and $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$.

$\underline{\mathsf{S_I}(M_1,\ldots,M_N)}$

1. Select $\gamma = (p, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BMsetup}(1^\kappa)$ and $a, b \xleftarrow{\$} \mathbb{Z}_p$, choose $g_2, g_3, h, u, v, d \xleftarrow{\$} \mathbb{G}$ and set $g_1 \leftarrow g^a, g_4 \leftarrow g^b$. Let $pk = (\gamma, g_1, g_2, g_3, g_4, h, u, v, d)$ and $sk = (a, b)$.
2. For $j = 1$ to N, select $r_j, s_j, t_j \xleftarrow{\$} \mathbb{Z}_p$ and set: $C_j \leftarrow [g^{r_j},\ (g_1^j h)^{r_j},\ M_j e(g_1, g_2)^{r_j},\ g^{t_j},\ (u^{r_j} v^{s_j} d)^b (g_3^j h)^{t_j},\ u^{r_j},\ s_j]$.
3. Send $(pk, C_1, \ldots, C_N)$ to Receiver.
4. Conduct $ZKPoK\{(a) : g_1 = g^a\}$.

$\underline{\mathsf{R_I}()}$

5. Verify pk and the proof.[a] Check for $j = 1$ to N: $\mathsf{VerifyCiphertext}(pk, C_j, j) = 1$. If any check fails, output $\perp$.

Output $S_0 = (pk, sk)$.

Output $R_0 = (pk, C_1, \ldots, C_N)$.

$\underline{\mathsf{S_T}(S_{i-1})}$

$\underline{\mathsf{R_T}(R_{i-1}, \sigma_i)}$

1. Parse C_{σ_i} as $(c_1, \ldots, c_7)$, select $x, y \xleftarrow{\$} \mathbb{Z}_p$ and compute $v_1 \leftarrow g^x c_1$.
2. Send v_1 to Sender, and conduct:
$$WIPoK\{(\sigma_i, x, c_2, c_4, c_5, c_6, c_7) : e(v_1/g^x, (g_1^{\sigma_i} h)) = e(c_2, g) \wedge e(c_6, g) = e(v_1/g^x, u) \wedge e(c_5, g) = e(c_6 v^{c_7} d, g_4) e(c_4, g_3^{\sigma_i} h)\}.$$

3. Set $R \leftarrow e(v_1, g_2^a)$.
4. Send R to Receiver and conduct: $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$.

5. If the proof does not verify, output $\perp$. Else output $M'_{\sigma_i} \leftarrow \frac{c_3 \cdot e(g_1, g_2)^x}{R}$.

Output $S_i = S_{i-1}$.

Output $R_i = (R_{i-1}, M'_{\sigma_i})$.

[a] By verify pk, we mean check that γ contains parameters for a bilinear map, where p is prime and g generates $\mathbb{G}$ with order p. Also, verify that the remaining pk elements are members of $\mathbb{G}$.

Figure 2: Our adaptive $\mathsf{OT}^N_{k \times 1}$ protocol. $\mathsf{VerifyCiphertext}$ is described above.

1. To begin, $\hat{\mathsf{R}}'$ selects a random collection of messages $\bar{M}_1, \ldots, \bar{M}_N \xleftarrow{\$} \mathbb{G}_T$ and follows the $\mathsf{S_I}$ algorithm (from Figure 2) with these as input up to the point where it obtains $(pk, C_1, \ldots, C_N)$.

2. It sends $(pk, C_1, \ldots, C_N)$ to $\hat{\mathsf{R}}$ and then *simulates* the interactive proof $ZKPoK\{(a) : g_1 = g^a\}$. (Even though $\hat{\mathsf{R}}'$ knows $sk = a$, it ignores this value and simulate this proof step.)

3. For each of k transfers initiated by $\hat{\mathsf{R}}$,

   (a) $\hat{\mathsf{R}}'$ verifies the received WIPoK and uses the knowledge extractor E_2 to obtain the values

$\sigma_i, x, y, c_1, c_2, c_3, c_4$ from it. $\hat{\mathsf{R}}'$ aborts and outputs error when E_2 fails.

(b) When $\sigma_i \in [1, N]$, $\hat{\mathsf{R}}'$ queries the trusted party T to obtain M_{σ_i}, parses C_{σ_i} as $(c_1, \ldots, c_7)$ and responds with $R = \frac{c_3 e(g_1, g_2)^x}{M_{\sigma_i}}$ (if T returns $\perp$, $\hat{\mathsf{R}}'$ aborts the transfer). When $\sigma_i \notin [1, N]$, $\hat{\mathsf{R}}'$ follows the normal protocol. In both cases, $\hat{\mathsf{R}}'$ simulates $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$.

4. $\hat{\mathsf{R}}'$ uses $\hat{\mathsf{R}}$'s output as its own.

Theorem 3.2 *Let ϵ_{ZK} be the maximum advantage with which any p.p.t. algorithm distinguishes a simulated ZKPoK, and ϵ_{Ext} be the maximum probability that the extractor E_2 fails (with ϵ_{ZK} and ϵ_{Ext} both negligible in κ). If all p.p.t. algorithms have negligible advantage $\leq \epsilon$ at solving the 3DDH problem, then:*

$$\mathbf{Pr}\left[D(\mathbf{Real}_{\mathsf{S},\hat{\mathsf{R}}}(N, k, M_1, \ldots, M_N, \Sigma)) = 1\right] - \mathbf{Pr}\left[D(\mathbf{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}(N, k, M_1, \ldots, M_N, \Sigma)) = 1\right] \leq (k+1)\epsilon_{ZK} + k\epsilon_{Ext} + N\epsilon\left(1 + \frac{p}{p-1}\right).$$

A proof of Theorem 3.2 is sketched in Appendix A.1 and detailed in [24].

Receiver Security. For any real-world cheating sender $\hat{\mathsf{S}}$ we can construct an ideal-world sender $\hat{\mathsf{S}}'$ such that all p.p.t. distinguishers have negligible advantage at distinguishing the distribution of the real and ideal experiments. Let us now describe the operation of $\hat{\mathsf{S}}'$, which runs $\hat{\mathsf{S}}$ internally, interacting with it in the role of the Receiver.

1. To begin, $\hat{\mathsf{S}}'$ runs the R_I algorithm, with the following modification: when $\hat{\mathsf{S}}$ proves knowledge of a, $\hat{\mathsf{S}}'$ uses the knowledge extractor E_1 to extract a, outputting error if the extractor fails. Otherwise, it has obtained the values $(pk, C_1, \ldots, C_N)$.
2. For $i = 1$ to N, $\hat{\mathsf{S}}'$ decrypts each of $\hat{\mathsf{S}}$'s ciphertexts $C_1, \ldots, C_N$ using the value a as a decryption key,[6] and sends the resulting $M_1^*, \ldots, M_N^*$ to the trusted party T.
3. Whenever T indicates to $\hat{\mathsf{S}}'$ that a transfer has been initiated, $\hat{\mathsf{S}}'$ runs the transfer protocol with $\hat{\mathsf{S}}$ on the fixed index 1. If the transfer succeeds, $\hat{\mathsf{S}}'$ returns the bit 1 (indicating success) to T, or 0 otherwise.
4. $\hat{\mathsf{S}}'$ uses $\hat{\mathsf{S}}$'s output as its own.

Theorem 3.3 *Let ϵ_{WI} be the maximum advantage that any p.p.t. algorithm has at distinguishing a WIPoK, and let ϵ_{Ext} be the maximum probability that the extractor E_1 fails. Then $\forall$ p.p.t. D:*

$$\mathbf{Pr}\left[D(\mathbf{Real}_{\hat{\mathsf{S}},\mathsf{R}}(N, k, M_1, \ldots, M_N, \Sigma)) = 1\right] - \mathbf{Pr}\left[D(\mathbf{Ideal}_{\hat{\mathsf{S}}',\mathsf{R}'}(N, k, M_1, \ldots, M_N, \Sigma)) = 1\right] \leq (k+1)\epsilon_{Ext} + k\epsilon_{WI}.$$

A proof of Theorem 3.3 is sketched in Appendix A.2 and detailed in [24].

[6] Parse C_i as $(c_1, \ldots, c_7)$ and decrypt as $M_i^* = c_3 / e(c_1, g_2^a)$. As noted in Section 4, one can modify the protocol so that the Sender conducts a PoK of the value g_2^a.

4 Efficiently Supporting Multiple Receivers

Adaptive Oblivious Transfer ($\mathsf{OT}_{k\times 1}^{N}$) is traditionally defined as a protocol between a Sender and a single Receiver. However, the way it is typically used in practical works such as Coull et al. [15] and Camenisch et al. [8] is that $U \geq 1$ distinct Receivers all interact with a single Sender.

Extending the full simulation definition to cover this explicitly is rather straightforward. We do so in Appendix B. The main technical concern is that every Receiver should have the same view of the database. That is, if two Receivers make a request on index i and neither transfer resulted in an error, then those Receivers must have obtained the same message. In Appendix B, we explain why our construction would satisfy such a notion – namely, that all Receivers share the same database and a Receiver does not accept a message unless the Sender can prove that it correctly corresponds to this database. For simplicity, we assume secure channels for the transfer phase.

Eliminating the $O(\lambda U)$ term. Interestingly, we can also improve the efficiency of our initialization protocol when multiple Receivers are present. In the current protocol of Figure 2, the Sender must conduct the proof of knowledge $ZKPoK\{(a) : g_1 = g^a\}$ with each Receiver. This can be accomplished using a very inexpensive interactive four-round proof in the standard model.

Fortunately even this minimal per-user initialization can be eliminated by assuming a Common Reference String shared by the Sender and all Receivers and using an NIZKPoK to broadcast this proof to all Receivers. To instantiate this proof, we suggest the efficient proof system of Groth and Sahai [26], which permits proofs of pairing-based statements under the Decision Linear assumption [4]. One wrinkle in this approach is that our proof of Receiver security assumes that the simulator can extract the trapdoor $a \in \mathbb{Z}_p$ from the ZKPoK, in order to decrypt the ciphertext vector $C_1, \ldots, C_N$. However, the knowledge extractor for the Groth-Sahai proof system is limited in that it can only extract elements of the bilinear image group $\mathbb{G}$. Fortunately for our purposes, the value $g_2^a \in \mathbb{G}$ can be used as an alternative trapdoor (see Section 3.3 for details). Thus when using the Groth-Sahai system we must re-write the proof as $NIZKPoK\{(g_2^a) : e(g_1, g_2) = e(g_2^a, g)\}$.[7]

5 Conclusions and Open Problems

We presented the first efficient, adaptive oblivious transfer protocol which is fully-simulatable under simple, static assumptions. Our protocol is practical and can be used as a building block in larger database applications, such as [15, 41, 8], as a step to reducing the overall assumptions on the system.

We leave open several interesting problems. First, we use standard zero-knowledge proof and extraction techniques which require rewinding, and thus, our scheme is not UC-secure. A natural question is whether one can obtain UC-security by replacing our interactive proofs with the non-interactive Groth-Sahai proofs [26]. Unfortunately, this is not an easy substitution. Our security proofs use techniques from the Boneh-Boyen cryptosystem that depend fundamentally on the ability to extract *integers* from the Receiver's proof of knowledge during the Transfer phase. The Groth-Sahai proof system is only F-extractable, meaning that one can obtain only group elements from the extractor (even when the proof is over integer witnesses). One can easily substitute a bit-by-bit proof of each integer, but we would hope to identify a more practical approach.

[7] As mentioned by Groth and Sahai, statements of this form must be slightly re-written to enable full zero knowledge. The equivalent statement is $ZKPoK\{(g_2^a, g_1') : e(g_2^a, g)e(g_1', g_2^{-1}) = 1 \ \wedge \ e(g_1', g) = e(g_1, g)\}$.

It would be interesting to know if the observations about and manipulations of the Hohenberger-Waters signatures [28] identified in this work could be extended to applications such as anonymous credentials and electronic cash, where most efficient constructions still require random oracles or strong complexity assumptions. One of the main difficulties is that many interesting protocols require an F-signature together with an efficient range proof (i.e., method for proving in zero-knowledge that a committed value lies within a public range.) Currently, the only efficient techniques for doing the latter require either the Strong RSA assumption [13, 9, 6] or (more recently) the q-Strong Diffie-Hellman assumption [7, 12]. (Here q need only be linked to a security parameter, e.g., $q = 256$.) It would be interesting if range proofs under weaker assumptions could be devised.

References

[1] Mira Belenkiy, Melissa Chase, Markulf Kohlweiss, and Anna Lysyanskaya. P-signatures and noninteractive anonymous credentials. In *Theory of Cryptography Conference*, volume 4948 of LNCS, pages 356–374, 2008.

[2] Dan Boneh and Xavier Boyen. Efficient selective-ID secure Identity-Based Encryption without random oracles. In *EUROCRYPT '04*, volume 3027 of LNCS, pages 223–238, 2004.

[3] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *EUROCRYPT '04*, volume 3027 of LNCS, pages 382–400, 2004.

[4] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *CRYPTO '04*, volume 3152 of LNCS, pages 45–55, 2004.

[5] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Advances in Cryptology – EUROCRYPT '06*, volume 4004 of LNCS, pages 573–592, 2006.

[6] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT '00*, volume 1807 of LNCS, pages 431–444, 2000.

[7] Jan Camenisch, Rafik Chaabouni, and Abhi Shelat. Efficient protocols for set membership and range proofs. In *ASIACRYPT '08*, volume 5350 of LNCS, pages 234–252. Springer, 2008.

[8] Jan Camenisch, Maria Dubovitskaya, and Gregory Neven. Oblivious transfer with access controls. In *ACM CCS '09*, pages 131–140, 2009.

[9] Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number n is the product of two safe primes. In *EUROCRYPT '99*, volume 1592 of LNCS, pages 107–122, 1999.

[10] Jan Camenisch, Gregory Neven, and abhi shelat. Simulatable adaptive oblivious transfer. In *EUROCRYPT '07*, volume 4515 of LNCS, pages 573–590, 2007.

[11] Jan Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *CRYPTO '97*, volume 1296 of LNCS, pages 410–424, 1997.

[12] Rafik Chaabouni, Helger Lipmaa, and Abhi Shelat. Additive combinatorics and discrete logarithm based range protocols, 2009. Cryptology ePrint Archive: 2009/469. Available at `http://eprint.iacr.org/2009/469.pdf`.

[13] Agnes Chan, Yair Frankel, and Yiannis Tsiounis. Easy come – easy go divisible cash. In *EUROCRYPT '98*, volume 1403 of LNCS, pages 561–575, 1998.

[14] Yalin Chen, Jue-Sam Chou, and Xian-Wu Hou. A novel k-out-of-n oblivious transfer protocols based on bilinear pairings, 2010. Cryptology ePrint Archive: Report 2010/027.

[15] Scott Coull, Matthew Green, and Susan Hohenberger. Controlling access to an oblivious database using stateful anonymous credentials. In *Public Key Cryptography*, volume 5443 of LNCS, pages 501–520, 2009.

[16] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO '94*, volume 839 of LNCS, pages 174–187, 1994.

[17] Yevgeniy Dodis, Shai Halevi, and Tal Rabin. A cryptographic solution to a game theoretic problem. In *CRYPTO '00*, volume 1880 of LNCS, pages 112–130, 2000.

[18] Yevgeniy Dodis and Aleksandr Yampolskiy. A Verifiable Random Function with Short Proofs an Keys. In *Public Key Cryptography '05*, volume 3386 of LNCS, pages 416–431, 2005.

[19] Georg Fuchsbauer. Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. Cryptology ePrint Archive, Report 2009/320, 2009. `http://eprint.iacr.org/`.

[20] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC '87*, pages 218–229, 1987.

[21] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing*, 17(2), 1988.

[22] Matthew Green and Susan Hohenberger. Blind identity-based encryption and simulatable oblivious transfer. In *ASIACRYPT '07*, volume 4833 of LNCS, pages 265–282, 2007.

[23] Matthew Green and Susan Hohenberger. Universally composable adaptive oblivious transfer. In *ASIACRYPT '08*, volume 5350 of LNCS, pages 179–197, 2008.

[24] Matthew Green and Susan Hohenberger. Practical adaptive oblivious transfer from simple assumptions, 2010. The full version of this work appears in the Cryptology ePrint Archive: Report 2010/109.

[25] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT '06*, volume 4284 of LNCS, pages 444–459. Springer, 2006.

[26] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT '08*, volume 4965 of LNCS, pages 415–432. Springer, 2008.

[27] Susan Hohenberger, Guy N. Rothblum, abhi shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. In *TCC '07*, volume 4392 of LNCS, pages 233–252, 2007.

[28] Susan Hohenberger and Brent Waters. Realizing hash-and-sign signatures under standard assumptions. In *EUROCRYPT '09*, volume 5479 of LNCS, pages 333–350, 2009.

[29] Stanislaw Jarecki and Xiaomin Liu. Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection. In *TCC '09*, volume 5444 of LNCS, pages 577–594, 2009.

[30] Joe Kilian. Founding cryptography on oblivious transfer. In *STOC '88*, pages 20–31, 1988.

[31] Sébastien Kunz-Jacques and David Pointcheval. About the security of MTI/C0 and MQV. In *SCN '06*, volume 4116 of LNCS, pages 156–172. Springer, 2006.

[32] Kaoru Kurosawa and Ryo Nojima. Simple adaptive oblivious transfer without random oracle. In *ASIACRYPT '09*, volume 5912 of LNCS, pages 334–346, 2009.

[33] Kaoru Kurosawa, Ryo Nojima, and Le Trieu Phong. Efficiency-improved fully simulatable adaptive OT under the DDH assumption. In *SCN '10*, volume 6280 of LNCS, pages 172–181, 2010.

[34] Fabien Laguillaumie, Pascal Paillier, and Damien Vergnau. Universally convertible directed signature. In *ASIACRYPT '05*, volume 3788 of LNCS, pages 682–701, 2005.

[35] Benoît Libert and Damien Vergnaud. Multi-use unidirectional proxy re-signatures. In *CCS '08*, pages 511–520. ACM, 2008.

[36] Yehuda Lindell. Efficient fully-simulatable oblivious transfer. In *CT-RSA '08*, volume 4964 of LNCS, pages 52–70, 2008.

[37] Moni Naor and Benny Pinkas. Oblivious transfer with adaptive queries. In *CRYPTO '99*, volume 1666 of LNCS, pages 573–590, 1999.

[38] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO '92*, volume 576 of LNCS, pages 129–140, 1992.

[39] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO '08*, volume 5157, pages 554–571, 2008.

[40] Michael Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Aiken Computation Laboratory, Harvard University, 1981.

[41] Alfredo Rial, Markulf Kohlweiss, and Bart Preneel. Universally composable adaptive priced oblivious transfer. In *Pairing 2009*, volume 5671 of LNCS, pages 231–247, 2009.

[42] Claus-Peter Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):239–252, 1991.

[43] Hung-Min Sun, Yalin Chen, and Jue-Sam Chou. An efficient secure oblivious transfer, 2009. Cryptology ePrint Archive: Report 2009/521.

[44] S. Wiesner. Conjugate coding. *SIGACT News*, 15:7888, 1983.

[45] Andrew Yao. How to generate and exchange secrets. In *FOCS*, pages 162–167, 1986.

A Proof of Sender and Receiver Security

A.1 Proof of Sender Security (Theorem 3.2)

Proof. We will begin with $\mathbf{Real}_{\mathsf{S},\hat{\mathsf{R}}}$, then modify the distribution via a series of hybrid games until we arrive at a distribution identical to that of $\mathbf{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}$. Let us define these hybrids as follows:

> **Game 0**. The real-world experiment conducted between S and $\hat{\mathsf{R}}$ ($\mathbf{Real}_{\mathsf{S},\hat{\mathsf{R}}}$).
>
> **Game 1**. This game modifies **Game 0** as follows: (1) each of S's ZKPoK executions is replaced with a *simulated* proof of the same statement,[8] and (2) the knowledge extractor E_2 is used to obtain the values $(\sigma_i, x, y, z, \bar{c}_4, \bar{c}_5, \bar{c}_6, \bar{c}_7)$ from each of $\hat{\mathsf{R}}$'s transfer queries. Whenever the extractor fails, S terminates the experiment and outputs the distinguished symbol error.
>
> **Game 2**. This game modifies **Game 1** such that, whenever the extracted value $\sigma_i \in [1, N]$, S's response R is computed using the following approach: parse $C_{\sigma_i} = (c_1, \ldots, c_7)$ and set $R = \frac{c_3 e(g_1,g_2)^x}{M_{\sigma_i}}$. When $\sigma_i \notin [1, N]$, the response is computed using the normal protocol.
>
> **Game 3**. This game modifies **Game 2** by replacing the input to S_I with a dummy vector of random messages $\bar{M}_1, \ldots, \bar{M}_N \in \mathbb{G}_T$. However when S computes a response value using the technique of **Game 2**, the response is based on the original message vector $M_1, \ldots, M_N$. We claim that the distribution of this game is equivalent to that of $\mathbf{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'}$.

Let us now consider the following Lemmas. For notational convenience, define:

$$\mathbf{Adv}\,[\,\mathbf{Game\ i}\,] = \mathbf{Pr}\,[\,D(\mathbf{Game\ i}) = 1\,] - \mathbf{Pr}\,[\,D(\mathbf{Game\ 0}) = 1\,].$$

Lemma A.1 *If all p.p.t. algorithms D distinguish a simulated ZKPoK with advantage at most ϵ_{ZK} and the extractor E_2 fails with probability at most ϵ_{Ext}, then* $\mathbf{Adv}\,[\,\mathbf{Game\ 1}\,] \leq (k+1)\cdot\epsilon_{ZK} + k\cdot\epsilon_{Ext}$.

Proof. We approach the proof via a hybrid argument. Consider a first hybrid in which all ZKPoK instances are conducted normally (non-simulated). In the first hybrid we replace the proof $ZKPoK\{(a) : g_1 = g^a\}$ with a simulated proof, and in each of k subsequent hybrids we simulate one instance of $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$. If there exists a p.p.t. D capable of distinguishing any two consecutive hybrids with advantage $> \epsilon_{ZK}$ then we can use D as an oracle for distinguishing real and simulated proofs with identical advantage. Clearly this contradicts our assumption. By applying this argument over $k + 1$ hybrids, we obtain that no p.p.t. algorithm D can distinguish the first and last hybrids with advantage $> (k + 1) \cdot \epsilon_{ZK}$.

To complete the proof, we must consider the probability that S outputs error. Clearly when no instance of extractor E_2 fails, this event will not occur (the distribution of **Game 1** is identical to that of **Game 0**). It remains only to calculate the maximal probability that one run of E_2 fails, which is $k \cdot \epsilon_{Ext}$. Summing all of these values we obtain the bound $\mathbf{Adv}\,[\,\mathbf{Game\ 1}\,] \leq (k + 1) \cdot \epsilon_{ZK} + k \cdot \epsilon_{Ext}$. □

Lemma A.2 *If no p.p.t. algorithm has advantage $> \epsilon$ in solving the 3DDH problem, then*

$$\mathbf{Adv}\,[\,\mathbf{Game\ 2}\,] - \mathbf{Adv}\,[\,\mathbf{Game\ 1}\,] \leq \frac{Np}{p-1} \cdot \epsilon.$$

[8] This includes at most $k + 1$ PoK executions, including the initial $ZKPoK\{(a) : g_1 = g^a\}$ and each subsequent proof $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$.

Proof. Recall that S has extracted $(\sigma_i, x, y, z, \bar{c}_4, \bar{c}_5, \bar{c}_6, \bar{c}_7)$ from each of $\hat{\mathsf{R}}$'s transfer queries. For every query where $\sigma_i \notin [1, N]$, S calculates the response R as in the normal protocol, and thus the distribution of R is identical to **Game 1**. Thus for this proof we need only consider queries where $\sigma_i \in [1, N]$.

Given a transfer request containing v_1, let us implicitly define $g^{r'} = v_1/g^x$ for some $r' \in \mathbb{Z}_p$. Express the σ_i^{th} ciphertext in the database as $C_{\sigma_i} = (c_1, \ldots, c_7)$. If $g^{r'} = c_1$ then our computed response R will have the same distribution as in the normal protocol. To show this, let $c_1 = g^{r_{\sigma_i}}$ for some $r_{\sigma_i} \in \mathbb{Z}_p$ and $c_3/M_{\sigma_i} = e(g_1, g_2)^{r_{\sigma_i}}$. We can now write the normal calculation of R as:

$$R = e(c_1 g^x, g_2^a) = e(g^{r_{\sigma_i}} g^x, g_2^a) = e(g_1, g_2)^{r_{\sigma_i}} e(g_1, g_2)^x = \frac{c_3 e(g_1, g_2)^x}{M_{\sigma_i}}$$

It remains only to consider the case where $g^{r'} \neq c_1$. We will refer to this as a *forged query* and argue that $\hat{\mathsf{R}}$ cannot issue such a query except with negligible probability under the 3DDH assumption in $\mathbb{G}$. Specifically, if $\hat{\mathsf{R}}$ submits a forged query with non-negligible probability, then we can construct a solver $\mathcal{B}$ for 3DDH that succeeds with non-negligible advantage. Intuitively, our proof revolves around the structure of $\bar{c}_4, \ldots, \bar{c}_7$, which can be viewed collectively as a stateful Hohenberger-Waters signature on message r' under state σ_i. The core of our proof is to show that if $\hat{\mathsf{R}}$ can submit a valid request where $g^{r'} \neq c_1$, then $\mathcal{B}$ can succeed *even if* (as will normally be the case) $\mathcal{B}$ cannot actually compute the forged message r'.[9]

We now describe the solver $\mathcal{B}$. $\mathcal{B}$ takes as input a 3DDH tuple $(g, g^\tau, g^\psi, g^\omega, Z)$, where $Z = g^{\tau\psi\omega}$ or is random, and each value τ, ψ, ω was chosen at random from $\mathbb{Z}_p$. It will simulate S's interaction with $\hat{\mathsf{R}}$ via the following simulation.

Simulation Setup. $\mathcal{B}$ first picks $j^* \xleftarrow{\$} [1, N]$ and $a, y_v, y_d, x_v, x_d, x_h, x_z, r_j, t_j \xleftarrow{\$} \mathbb{Z}_p$. It sets $u = g^\psi$, $v = g^{\psi x_v} g^{y_v}$, $d = g^{-\psi x_d} g^{y_d}$, $h = g^{-\psi j^*} g^{x_h}$, $g_3 = g^\psi g^{x_z}$, $g_4 = g^\tau$. Thus, we implicitly have $b = \tau$. The remaining components of pk are chosen as in the real protocol.

For $j = 1$ to N, $\mathcal{B}$ generates each correctly-distributed ciphertext $C_j = (c_1, \ldots c_7)$ as follows:

> **The simulation for $j = j^*$.** Pick $r_j, t_j \xleftarrow{\$} \mathbb{Z}_p$. Set $s_j = (x_d - r_j)/x_v$. Note that r_j, t_j are chosen as they ought to be. Each generator is uniquely randomized in the setup, with an extra degree of freedom left to set s_j. The remaining ciphertext elements are set as:
>
> $$(c_4, c_5, c_6, c_7) = \left(g^{t_j}, (g^\tau)^{y_v s_j + y_d} \cdot (g_3{}^j h)^{t_j}, u^{r_j}, s_j\right)$$
>
> where the additional components (c_1, c_2, c_3) are set identically to the real construction.
>
> By comparing the simulated components to the reals ones, it is clear that the only difference appears in c_5, which requires that we show the equality $(g^\tau)^{y_v s_j + y_d} = (u^{r_j} v^{s_j} d)^\tau$.

[9] In this case, we use the term forgery loosely to include both a signature on a new message, or a re-assignment of a given signature to a different state. Our proof implicitly covers both conditions.

We have $(r_j + s_j x_v - x_d) = 0$ by the setting of s_j. This gives us:

$$\begin{aligned}
(g^\tau)^{y_v s_j + y_d} &= (g^{\tau\psi})^0 (g^\tau)^{y_v s_j + y_d} \\
&= (g^{\tau\psi})^{r_j + s_j x_v - x_d} (g^\tau)^{y_v s_j + y_d} \\
&= (g^{\psi r_j} g^{\psi s_j x_v} g^{-\psi x_d} g^{y_v s_j} g^{y_d})^\tau \\
&= (g^{\psi r_j} g^{(\psi x_v + y_v) s_j} g^{-\psi x_d + y_d})^\tau \\
&= (u^{r_j} v^{s_j} d)^\tau
\end{aligned}$$

The simulation for $j \neq j^*$. Pick $r_j, s_j, t'_j \xleftarrow{\$} \mathbb{Z}_p$. Set $Y = g^{t'_j} / (g^\tau)^{(r_j + x_v s_j - x_d)/(j - j^*)}$ and:

$$(c_4, c_5, c_6, c_7) = \Big(Y, (g^\tau)^{y_v s_j + y_d} \cdot Y^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j - j^*)}, u^{r_j}, s_j\Big)$$

where the additional components (c_1, c_2, c_3) are set identically to the real construction.

Let us define $Y = g^{t_j}$ and thus implicitly $t_j = t'_j - \tau(r_j + x_v s_j - x_d)/(j - j^*)$, which is randomly distributed in $\mathbb{Z}_p$. Just by inspection, it's clear that all of the elements except c_5 are correctly distributed. Thus it remains to show that:

$$(g^\tau)^{y_v s_j + y_d} \cdot Y^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j - j^*)} = (u^{r_j} v^{s_j} d)^\tau (g_3^j h)^{t_j}$$

We do this as follows:

$$\begin{aligned}
c_5 &= (g^\tau)^{y_v s_j + y_d} \cdot Y^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j - j^*)} \\
&= (g^\tau)^{y_v s_j + y_d} \cdot (g^{t_j})^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j - j^*)} \\
&= (g^{\tau\psi})^{(r_j + x_v s_j - x_d)} \cdot (g^\tau)^{y_v s_j + y_d} \cdot (g^{t_j})^{x_z j + x_h} \cdot (g^\psi)^{t'_j (j - j^*)} \cdot (g^{-\tau\psi})^{(r_j + x_v s_j - x_d)} \\
&= (g^{\psi(r_j + x_v s_j - x_d)})^\tau (g^{y_v s_j + y_d})^\tau \cdot (g^{x_z j} g^{x_h})^{t_j} \cdot (g^\psi)^{t'_j (j - j^*)} (g^{-\tau\psi})^{(r_j + x_v s_j - x_d)} \\
&= ((g^\psi)^{r_j} (g^{\psi x_v + y_v})^{s_j} (g^{-\psi x_d + y_d}))^\tau \cdot (g^{x_z j} g^{x_h})^{t_j} \cdot (g^\psi)^{t'_j (j - j^*)} (g^{-\tau\psi})^{(r_j + x_v s_j - x_d)} \\
&= (u^{r_j} v^{s_j} d)^\tau \cdot (g^{x_z j} g^{x_h})^{t_j} \cdot (g^\psi)^{t'_j (j - j^*)} (g^{-\tau\psi})^{(r_j + x_v s_j - x_d)} \\
&= (u^{r_j} v^{s_j} d)^\tau \cdot (g^{x_z j} g^{x_h})^{t_j} \cdot (g^{\psi(j - j^*)})^{t'_j - \tau(r_j + x_v s_j - x_d)/(j - j^*)} \\
&= (u^{r_j} v^{s_j} d)^\tau \cdot (g^{x_z j} g^{x_h})^{t_j} \cdot (g^{\psi(j - j^*)})^{t_j} \\
&= (u^{r_j} v^{s_j} d)^\tau \cdot ((g^{\psi + x_z})^j g^{-\psi j^* + x_h})^{t_j} \\
&= (u^{r_j} v^{s_j} d)^\tau \cdot (g_3^j h)^{t_j}
\end{aligned}$$

Regardless of the value j, the elements (c_1, c_2, c_3) are computed as in the real protocol. Note that pk and the ciphertext database are correctly distributed.

Answering Queries. Upon receiving a query from $\hat{\mathsf{R}}$, $\mathcal{B}$ verifies the accompanying WIPoK and extracts $(\sigma_i, x, y, z, \bar{c}_4, \bar{c}_5, \bar{c}_6, \bar{c}_7)$ and the value v_1. S can answer correctly-formed queries normally since it knows the secret key a. Recall that $\hat{\mathsf{R}}$ must issue at least one forged query where v_1/g^x is not equal to the first element of C_{σ_i}. When this occurs, if $\sigma_i \neq j^*$ then $\mathcal{B}$ aborts and outputs a random bit.

Otherwise let us consider the distribution of $\hat{\mathsf{R}}$'s query. For some $t, r' \in \mathbb{Z}_p$ the soundness of the WIPoK ensures that $(v_1/g^x, \bar{c}_6) = \left(g^{r'}, u^{r'}\right)$ and $(\bar{c}_4, \bar{c}_5) = \left(g^t, (u^{r'} v^{\bar{c}_7} d)^b (g_3^{\sigma_i} h)^t\right)$. By substitution we obtain:

$$\begin{aligned} \bar{c}_5 &= (g^{\psi r'} g^{(\psi x_v + y_v)\bar{c}_7} g^{(-\psi x_d) + y_d})^{\tau} (g^{(\psi + x_z) j^*} g^{-\psi j^*} g^{x_h})^t \\ &= g^{\tau\psi(r' + x_v \bar{c}_7 - x_d)} g^{\tau(y_v \bar{c}_7 + y_d)} g^{t(x_z j^* + x_h)} \end{aligned}$$

Let us implicitly define the value $h' = (v_1/g^x) g^{x_v \bar{c}_7 - x_d} = g^{r' + x_v \bar{c}_7 - x_d}$. $\mathcal{B}$ can obtain $h'^{\tau\psi}$ by computing $\bar{c}_5 / (g^{\tau(y_v \bar{c}_7 + y_d)} \bar{c}_4{}^{x_z j^* + x_h})$. Provided that $h' \neq 1$, $\mathcal{B}$ can now compute a solution to the 3DDH problem by comparing $e\left(h'^{\tau\psi}, g^{\omega}\right) \stackrel{?}{=} e\left(Z, h'\right)$. If $h' = 1$ then $\mathcal{B}$ aborts and outputs a random bit.

Probability of abort. There are two conditions in which $\mathcal{B}$ aborts: (1) when $\hat{\mathsf{R}}$ does not issue a forgery for $\sigma_i = j^*$, and (2) when $\sigma_i = j^*$ but $((v_1/g^x) g^{x_v \bar{c}_7 - x_d}) = 1$. Since j^*, x_v, x_d are outside of $\hat{\mathsf{R}}$'s view and our base assumption is that $\hat{\mathsf{R}}$ that makes at least one request on $\sigma_i \in [1, N]$, the probability that $\mathcal{B}$ does *not* abort is $\geq \frac{p-1}{p} \cdot \frac{1}{N}$. Thus, if no p.p.t. algorithm solves 3DDH with probability $> \epsilon$, then $\mathbf{Adv}\left[\,\mathbf{Game\ 2}\,\right] - \mathbf{Adv}\left[\,\mathbf{Game\ 1}\,\right] \leq \frac{Np}{p-1} \cdot \epsilon$

$\square$

Lemma A.3 *If no p.p.t adversary has advantage $> \epsilon$ at solving the 3DDH problem, then*

$$\mathbf{Adv}\left[\,\mathbf{Game\ 3}\,\right] - \mathbf{Adv}\left[\,\mathbf{Game\ 2}\,\right] \leq N \cdot \epsilon.$$

Proof. We show, via a series of hybrids, that when D distinguishes **Game 3** from **Game 2** with probability $> N \cdot \epsilon$ we obtain a solver for the 3DDH problem that succeeds with probability $> \epsilon$. In each hybrid we will process a message vector $(\bar{M}_1, \ldots, \bar{M}_N)$, which is initially set to $(M_1, \ldots, M_N)$. In each of hybrids 1 through N we will replace one message with a random value $\in \mathbb{G}_T$, until we arrive at the distribution of **Game 3**. For each $j \in [1, N]$ we show that when D that differentiates Hybrid j from Hybrid $j-1$ with probability $> \epsilon$, this implies a solver for the 3DDH problem that succeeds with identical advantage. Thus by summation over N hybrids, we show that all distinguishers differentiate the first and last hybrids (**Game 2** and **Game 3**) with probability $\leq N \cdot \epsilon$. This proof uses techniques due to Boneh and Boyen [2].

Hybrid 0. Let $(\gamma, g, g^a, g^b, g^c, Q)$ be a candidate 3DDH tuple and and initialize i^* to be any value in $[1, N]$. Set $(\bar{M}_1, \ldots, \bar{M}_N) = (M_1, \ldots, M_N)$. In this and subsequent hybrids, generate pk as follows: select $\beta, w \xleftarrow{\$} \mathbb{Z}_p$, set $g_1 = g^a, g_2 = g^b, h = g_1^{-i^*} g^{\beta}, u = g^w$, and select the remaining elements of pk as in the normal protocol. Observe that the resulting pk has exactly the same distribution as that of **Game 2**. Finally, compute the ciphertext vector $(C_1, \ldots, C_N)$ as in the normal protocol (this does *not* require knowledge of the secret value a.)

Answering $\hat{\mathsf{R}}$'s queries. The parameters above imply that $g_2^a = g^{ab}$, which cannot be efficiently computed. However, as in **Game 2** we can extract the values (σ_i, c_2, x, y) from each of $\hat{\mathsf{R}}$'s queries; In **Game 2** we also showed that for $\sigma_i \in [1, N]$ we can correctly respond to the query without using a.

It remains to show that we can correctly answer queries when $\sigma_i \notin [1, N]$. To do this, we select $s \xleftarrow{\$} \mathbb{Z}_p$ and compute $R \leftarrow e(v_1/g^x, g_2^{\frac{-\beta}{\sigma_i - i^*}} (g_1^i h)^s) / e(c_2, g_2^{\frac{-1}{\sigma_i - i^*}} g^s) e(g_1, g_2)^x$. Observe that these responses have the exactly the same distribution as in **Game 3**, since for $\tilde{s} = s - b/(\sigma_i - i^*)$:

$$R = \frac{e(g^r, g_2^a (g_1^{\sigma_i} h)^{\tilde{s}})}{e((g_1^{\sigma_i} h)^r, g^{\tilde{s}})} \; e(g_1, g_2)^x = e(g_1, g_2)^{r+x}$$

We summarize by noting that the distribution of **Hybrid 0** is identical to that of **Game 2**.

Hybrids 1 **through** N**.** For each hybrid $j = 1$ to N, we proceed as in **Hybrid** $(j-1)$ but we set the j^{th} ciphertext to be the encryption of a random plaintext $\bar{M}_j \in \mathbb{G}_T$. We show that any p.p.t. algorithm D that distinguishes **Hybrid** j from **Hybrid** $j-1$ with advantage $> \epsilon$ can be used to construct a solver $\mathcal{B}$ for the 3DDH problem that succeeds with advantage $> \epsilon$. We now describe $\mathcal{B}$:

First set $i^* = j$ and compute pk as in **Hybrid** 0 (this gives $h = g_1^{-j} g^{\beta}$). For $\ell \in [1, j-1]$ set $\bar{M}_\ell \xleftarrow{\$} \mathbb{G}_T$. For $\ell \in [j, N]$ set $\bar{M}_\ell = M_\ell$. For each $\ell \neq j$ compute C_ℓ by encrypting $\bar{M}_\ell$ as in the normal protocol. To obtain the j^{th} ciphertext, select $t_j, s_j \xleftarrow{\$} \mathbb{Z}_p$ and compute:

$$C_j = \left(g^c, g^{c\beta}, M_j \cdot e(g, Q), g^{t_j}, (g^{cw} v^{s_j} d)^b (g_3^j h)^{t_j}, g^{cw}, s_j\right)$$

Note that when $Q = g^{abc}$ this ciphertext correctly encrypts M_j, and thus **Hybrid** j is identically distributed to **Hybrid** $(j-1)$. To show this, let $r_j = c$ and observe that C_j can be written as $(g^{r_j},\, (g_1^j h)^{r_j},\, M_j e(g_1, g_2)^{r_j},\, g^t,\, (u^{r_j} v^{s_j} d)^b (g_1^j h)^t,\, u^{r_j},\, s_j)$. Similarly, when Q is random, C_j is the encryption of a random element $\in \mathbb{G}_T$ and thus has the correct distribution for **Hybrid** j.

$\mathcal{B}$ runs D using the distribution of the simulation above. When D outputs a result, $\mathcal{B}$ simply returns this value as its output. Since the difference between the distributions depends only on Q, when D successfully distinguishes the two hybrids with advantage $> \epsilon$, then $\mathcal{B}$ solves 3DDH with advantage $> \epsilon$.

We observe that **Hybrid** N encrypts a vector of randomly-distributed plaintexts $(\bar{M}_1, \ldots, \bar{M}_N)$, and is therefore identically distributed to **Game 3**. By summation over N hybrids we bound D's advantage at distinguishing **Game 2** and **Game 3** to $\leq N\epsilon$.

□

By summing over hybrids **Game 0** to **Game 3**, we obtain $\mathbf{Adv}\left[\,\mathbf{Game\ 3}\,\right] \leq (k+1)\epsilon_{ZK} + k\epsilon_{Ext} + N\epsilon(1 + \frac{p}{p-1})$. For the Schnorr proofs we use, $\epsilon_{ZK} = 0$. This concludes the proof of Sender security. □

A.2 Proof of Receiver Security (Theorem 3.3)

Proof. We again arrive at the ideal-world sender via a series of hybrid games:

Game 0. The real-world experiment conducted between $\hat{\mathsf{S}}$ and R ($\mathbf{Real}_{\hat{\mathsf{S}},\mathsf{R}}$).

Game 1. A modification of **Game 0** in which R applies the knowledge extractor E_1 to $\hat{\mathsf{S}}$'s proof ZKPoK$\{a : g_1 = g^a\}$. If this extraction fails, R aborts and outputs $\perp$. Further, for transfers $i = 1$ through k, R uses the knowledge extractor E_3 on $\hat{\mathsf{S}}$'s proof $ZKPoK\{(a) : R = e(v_1, g_2^a) \wedge g_1 = g^a\}$ to extract the values a, aborting if the extractor fails (or returns inconsistent values).

Game 2. For transfer $i = 1$ to k, modify R's request such that $\sigma_i = 1$. The distribution of this game is identical to that of $\mathbf{Ideal}_{\hat{\mathsf{S}}',\mathsf{R}'}$.

Lemma A.4 *If the extractor* E_1 *and* E_3 *fail with probability at most* ϵ_{Ext}*, then* $\mathbf{Adv}\,[\,\mathbf{Game\ 1}\,] \leq (k+1)\epsilon_{Ext}$.

Proof. Observe that the distribution of **Game 1** differs from that of **Game 0** only when extractors E_1 or E_3 fail (or return inconsistent values). Since E_1 is used once, and E_3 at most k times, the probability of failure event is bounded by $\epsilon_{Ext} + (k \cdot \epsilon_{Ext})$. Thus we obtain our bound. □

Lemma A.5 *If the Receiver's WIPoK is distinguishable with maximum advantage* ϵ_{WI}*, then*

$$\mathbf{Adv}\,[\,\mathbf{Game\ 2}\,] - \mathbf{Adv}\,[\,\mathbf{Game\ 1}\,] =\leq k \cdot \epsilon_{WI}.$$

Proof. As in the proof of Lemma A.1 we approach this via a hybrid argument. The first hybrid is as in **Game 1**, with all transfers executed normally. With each subsequent hybrid, we alter one of the k transfers to index value 1. Since the value v_1 issued by $\hat{\mathsf{S}}'$ always has a random distribution, when a p.p.t D distinguisher differentiates the distribution of any two consecutive hybrids with advantage $> \epsilon_{WI}$ this naturally implies a distinguisher for the WIPoK with identical advantage. Summing over k transfers we obtain $\mathbf{Adv}\,[\,\mathbf{Game\ 2}\,] - \mathbf{Adv}\,[\,\mathbf{Game\ 1}\,] =\leq k\epsilon_{WI}$. □

Summing the differences, we have

$$\mathbf{Adv}\,[\,\mathbf{Game\ 2}\,] - \mathbf{Adv}\,[\,\mathbf{Game\ 0}\,] = (k+1)\epsilon_{Ext} + k\epsilon_{WI}.$$

For the Schnorr proofs we use, $\epsilon_{WI} = 0$. This concludes the proof of Receiver security. □

B Oblivious Transfer with Multiple Receivers

Traditional $\mathsf{OT}^k_{N\times 1}$ is a protocol between a Sender and a single Receiver. However, a major application for the use of $\mathsf{OT}^N_{k\times 1}$ is as a building block for constructing practical privacy-preserving databases, which seem likely to support multiple users. This raises questions about the applicability of the traditional definition, for example, with regards to consistency of each user's view of the database.

To address these issues, we now present a modified definition of adaptive Oblivious Transfer that re-casts the protocol as an interaction between a single Sender and $U \geq 1$ distinct Receivers. Our definition captures both the correctness and security properties that are required by a multi-Receiver protocol. Most importantly, it ensures that all Receivers share a consistent view of the Sender's database, which prevents a malicious Sender from targeting individual users and providing them with different messages. We will then sketch the slight proof modifications that are necessary prove our main construction secure in this model.

B.1 Definition of Adaptive Oblivious Transfer with Multiple Receivers

A multi-Receiver oblivious transfer scheme is a tuple of algorithms $(\mathsf{S_I}, \mathsf{R_I}, \mathsf{S_T}, \mathsf{R_T})$. During the initialization phase, the Sender individually conducts a (possibly interactive) protocol with the U distinct Receivers, where the Sender runs $\mathsf{S_I}(M_1, \ldots, M_N)$ to obtain state value S_0, and for $1 \le n \le U$ the n^{th} Receiver runs $\mathsf{R_I}()$ to obtain state value $R_{0,n}$. Next, for $1 \le i \le k$, the i^{th} transfer proceeds as follows: one of the U Receivers (identified by the index n) runs $\mathsf{R_T}(R_{i-1,n}, \sigma_{i,n})$ where $\sigma_{i,n}$ is the index of the message to be received, and the Sender runs $\mathsf{S_T}(S_{i-1})$ to obtain state value S_i. The receiver obtains state information $R_{i,n}$ and the message $M'_{\sigma_{i,n}}$ or $\perp$ indicating failure. Each of the remaining receivers simply retains their state from the previous transfer.

Definition B.1 (Multi-Receiver Full Simulation Security.) Consider the following experiments.

Real experiment. In experiment $\mathbf{Real}_{\hat{\mathsf{S}},\hat{\mathsf{R}}_1,\ldots,\hat{\mathsf{R}}_U}(N, k, M_1, \ldots, M_N, \Psi, \Sigma_1, \ldots, \Sigma_U)$, the possibly cheating sender $\hat{\mathsf{S}}$ is given messages $(M_1, \ldots, M_N)$ as input along with a selection strategy Ψ which determines which Receivers will speak during each transfer phase. $\hat{\mathsf{S}}$ interacts with the set of possibly cheating receivers $\hat{\mathsf{R}}_1(\Psi, \Sigma_1), \ldots, \hat{\mathsf{R}}_U(\Psi, \Sigma_U)$, where each Σ_n is a selection algorithm that on input the the full collection of messages thus far received by that Receiver, outputs the index σ_i of the next message to be queried. At the beginning of the experiment $\hat{\mathsf{S}}$ outputs initial state S_0 and for $1 \le n \le U$ $\hat{\mathsf{R}}_n$ outputs $R_{0,n}$. For $1 \le i \le k$ the sender computes $S_i \leftarrow \hat{\mathsf{S}}(S_{i-1})$, and computes $n \leftarrow \Psi(i)$. The n^{th} receiver computes $(R_{i,n}, M'_i) \leftarrow \hat{\mathsf{R}}(R_{i-1,n})$, where M'_i may or may not be equal to M_i. The non-speaking Receivers simply update their current state to be identical to the previous round's state. At the end of the k^{th} transfer the output of the experiment is $(S_k, R_{k,1}, \ldots, R_{k,U})$.

We define the *honest* Sender S as one that runs $\mathsf{S_I}(M_1, \ldots, M_N)$ in the first phase, during each transfer runs $\mathsf{S_T}()$ and outputs $S_k = \varepsilon$ as its final output. An *honest* Receiver R runs $\mathsf{R_I}$ in the first phase, and $\mathsf{R_T}(R_{i-1}, \sigma_i)$ at the i^{th} transfer, and outputs all of the received messages as its final output.

Ideal experiment. In experiment $\mathbf{Ideal}_{\hat{\mathsf{S}}',\hat{\mathsf{R}}'_1,\ldots,\hat{\mathsf{R}}'_U}(N, k, M_1, \ldots, M_N, \Psi, \Sigma_1, \ldots, \Sigma_U)$ the possibly cheating Sender algorithm $\hat{\mathsf{S}}'$ generates messages $(M_1^*, \ldots, M_N^*)$ and transmits them to a trusted party T along with Ψ. In the i^{th} round $\hat{\mathsf{S}}'$ sends a bit b_i to T. Let $n \leftarrow \Psi(i)$. The n^{th} possibly cheating Receiver $\hat{\mathsf{R}}'_n(\Psi, \Sigma_1)$ transmits $\sigma^*_{i,n}$ to T. If $b_i = 1$ and $\sigma^*_{i,n} \in \{1, \ldots, N\}$ then T hands $M^*_{\sigma^*_{i,n}}$ to $\hat{\mathsf{R}}'$. If $b_i = 0$ then T hands $\perp$ to $\hat{\mathsf{R}}'$. After the k^{th} transfer the output of the experiment is $(S_k, R_{k,1}, \ldots, R_{k,U})$. We define the honest Sender S' as one that sends $(M_1, \ldots, M_N)$ to T and outputs $S_k = \varepsilon$ as its output, and an honest Receiver $\mathsf{R}'{}_n$ as one that sends $\sigma_{i,n}$ to T and outputs the messages it received as its output.

Let $\ell(\cdot)$ be a polynomially-bounded function. We now define Sender and Receiver security.

Sender Security. Let $\mathsf{Corrupt} \cup \mathsf{Honest} = \hat{\mathsf{R}}_1, \ldots, \hat{\mathsf{R}}_U$ be a collection of real-world Receivers where $\mathsf{Corrupt}$ consists of colluding, corrupted receivers. An $\mathsf{OT}^N_{k\times 1}$ provides Sender security if for every set $\mathsf{Corrupt}$ there exists a set of ideal world Receivers such that $\forall N = \ell(\kappa)$, $k \in [1, N]$, $(M_1, \ldots, M_N)$, Ψ, $\Sigma_1, \ldots, \Sigma_U$, and every p.p.t. distinguisher:

$$\mathbf{Real}_{\mathsf{S},\hat{\mathsf{R}}_1,\ldots,\hat{\mathsf{R}}_U}(N, k, M_1, \ldots, M_N, \Psi, \Sigma_1, \ldots, \Sigma_U) \overset{c}{\approx} \mathbf{Ideal}_{\mathsf{S}',\hat{\mathsf{R}}'_1,\ldots,\hat{\mathsf{R}}'_U}(N, k, M_1, \ldots, M_N, \Psi, \Sigma_1, \ldots, \Sigma_U).$$

Receiver Security. Let $\mathsf{Corrupt} \cup \mathsf{Honest} = \hat{\mathsf{R}}_1, \ldots, \hat{\mathsf{R}}_U$ be a collection of real-world Receivers where $\mathsf{Corrupt}$ consists of colluding, corrupted receivers, and let $\hat{\mathsf{S}}$ be a corrupted Sender. $\mathsf{OT}^N_{k\times 1}$ provides Receiver security if for every real-world p.p.t. Sender $\hat{\mathsf{S}}$ and set $\mathsf{Corrupt}$ there exists a p.p.t. ideal-world sender $\hat{\mathsf{S}}'$ such that $\forall N = \ell(\kappa)$, $k \in [1, N]$, $(M_1, \ldots, M_N)$, Σ, and every p.p.t. distinguisher:

$$\mathbf{Real}_{\hat{\mathsf{S}},\hat{\mathsf{R}}_1,\ldots,\hat{\mathsf{R}}_U}(N, k, M_1, \ldots, M_N, \Psi, \Sigma_1, \ldots, \Sigma_U) \overset{c}{\approx} \mathbf{Ideal}_{\hat{\mathsf{S}}',\mathsf{R'}_1,\ldots,\mathsf{R'}_U}(N, k, M_1, \ldots, M_N, \Psi, \Sigma_1, \ldots, \Sigma_U).$$

B.2 Intuition for Multiple Receiver Security Proof

In Section 4 we discussed how to extend our main construction to support multiple Receivers. We now briefly sketch the proof modifications that are required to achieve security under the definition above.

The proof of Sender security remains roughly the same. Each time a Receiver (corrupt or honest) initiates a transfer request, our Simulator uses the extractor for the WIPoK, contacts the trusted party T to obtain a message, and simulates the correct response as in the original proof. Fundamentally it is irrelevant whether the transfer requests initiate from a single Receiver, or from multiple distinct Receivers, since the strategy for responding to them is valid in either case. We briefly observe that the corrupted Receivers gain no special information even if they are allowed to eavesdrop on communications between the honest Receivers and the Sender, since these responses consist of two information-theoretically blinded group elements and a witness indistinguishable proof.[10]

The proof of Receiver security is slightly different. In this case we construct a simulator for the corrupted real-world Sender and all of the corrupted real-world Receivers. As in the original proof, the ideal world Receiver uses the extractor for the Sender's ZKPoK to extract the secret key a, then decrypts the ciphertext database $(C_1, \ldots, C_N)$, submitting the decrypted messages to the trusted party T. Each time an honest Receiver initiates a transfer request, the simulator issues a simulated transfer request to the real-world Sender using the fixed message index 1. When a corrupted real-world Receiver initiates a transfer request, the simulator employs the proof extractor to obtain the message index, and submit this to T. The simulator then forwards communications between the corrupted real-world Sender and corrupted real-world Receiver, using its knowledge of a to determine whether the transfer succeeded or not.

C Proofs of Knowledge for Discrete Logarithms and Bilinear Groups

In this work, we use known zero-knowledge and witness indistinguishable techniques for proving statements about discrete logarithms, such as (1) proof of knowledge of a discrete logarithm modulo a prime [42] and (2) proof of the disjunction or conjunction of any two statements [16].

To facilitate these discrete-logarithm proofs, we will use the Pedersen commitment scheme [38] based on the discrete logarithm assumption, in which the public parameters are a group of prime order q, and random generators $(g_0, \ldots, g_m)$. In order to commit to the values $(v_1, \ldots, v_m) \in \mathbb{Z}_q^m$, pick a random $r \in \mathbb{Z}_q$ and set $C = g_0^r \prod_{i=1}^m g_i^{v_i}$. Schnorr's technique [42] can be used to efficiently prove knowledge of $ZKPoK\{(r, v_1, \ldots, v_m) : C = g_0^r \prod_{i=1}^m g_i^{v_i}\}$.

[10] In a formal proof we would replace these communications with simulated transcripts in which each honest Receiver requests the fixed index 1. It is easy to show that the under the WI property of the proof system, these simulated communications could not be distinguished from real communication transcripts.

These same ideas can be translated into the bilinear setting, as noted in prior works, e.g., [10]. Suppose that one wishes to prove knowledge of $ZKPoK\{(h) : H = e(g, h)\}$. Consider this honest-verifier proof of knowledge protocol under the Computational Diffie-Hellman assumption. The prover chooses a random $r \in \mathbb{Z}_p$ and sends $T = e(g, g)^r$ to the verifier. The verifier sends back a random $c \in \mathbb{Z}_p$. The prover returns the value $s = h^c g^r$. The verifier then accepts if and only if $e(s, g) = H^c T$.

In the WIPoK of Figure 2 we use a combination of techniques. The first equation involves only exponents and can therefore be conducted using the Schnorr techniques. The second equation may be conducted using the following protocol. The prover chooses random $t, t' \in \mathbb{Z}_p$ and sends $T = e(g, u)^t e(g, g)^{t'}$ to the verifier. The verifier sends back a random $c \in \mathbb{Z}_p$. The prover then sends $s = t + xc$ and $s' = c_6^c g^{t'}$. The verifier then accepts if $e(g, u)^s e(s', g) = Te(v_1, u)^c$. The third equation uses a similar approach.

D Practical F-Signatures from a Simple Assumption

We provided a direct construction of adaptive OT in the main body which is optimized for performance considerations. If we were willing to give up some efficiency and look at things in a more modular way, we can make the following observation. At a high-level, one common way to efficiently implement adaptive OT is to have the sender encrypt each message individually and then sign each encryption. When the Receiver wants help to blindly decrypt a ciphertext, she may send back a blinded ciphertext or her choice and prove in zero knowledge that she knows a signature on the underlying ciphertext.

In our construction, it is enough to only sign *part* of the ciphertext, specifically the randomness $r \in \mathbb{Z}_p$. However, the Receiver is given g^r as part of the ciphertext (and not r itself) and thus a signature on r and the value g^r is all that an honest Receiver can later prove knowledge of. Therefore, we must be sure that no malicious Receiver can forge a new, but valid pair of this form. The unforgeability property must be strengthened to capture the notion that it is difficult to produce a valid signature and *function of the message* pair for a previously unsigned message. This latter property is already known as *F-unforgeability*. Of course, in our Section 3.2 construction, we are able to relax this unforgeability requirement for better efficiency, but it is instructive to here explain the more general building block.

D.1 F-Signatures

F-Signatures were formalized by Belenkiy, Chase, Kohlweiss and Lysyanskaya [1] as a building block for non-interactive anonymous credentials. In the standard unforgeability notion for signatures [21], the adversary is not able to output a pair (m, σ), where σ is a valid signature on m, unless m was previously signed. In F-unforgeability, this notion is strengthened so that, for a given efficiently-computable bijection F, the adversary is not able to output a pair $(F(m), \sigma)$, where σ is a valid signature on m, unless m was previously signed. As an example, consider $F_g(m) = g^m$. Thus, the adversary need not *know* the message on which he forges, so long as he can produce a specified function of this message.

F-unforgeability is extremely useful. Here we highlight a relationship to adaptive oblivious transfer protocols. Belenkiy et al. [1] required F-unforgeability to combine their signatures with Groth-Sahai proofs [26] to obtain non-interactive anonymous credentials. Groth [25] implicitly uses F-unforgeability to obtain a new group signature scheme using Groth-Sahai proofs. Recall that

Groth-Sahai proofs of knowledge are only F-extractable; that is, the simulator may not be able to extract a witness w, but rather only some function of the witness $F(w)$, such as g^w. Thus, the unforgeability requirement must be strengthened to disallow forgeries on new values of w' even when the adversary need only produce $g^{w'}$ instead of w'. Combinations of F-signatures and GS proofs seem like a highly promising direction for many areas, including anonymous credentials and electronic cash. Since GS proofs are secure under static assumptions, this work could potentially provide the other required static building block.

Unfortunately, F-unforgeability is not easy to realize and typically requires much stronger complexity assumptions than the underlying signature scheme. Belenkiy et al. [1] introduced the concept with two constructions. First, they show that the weak Boneh-Boyen signatures are F-unforgeable under an interactive assumption, called Interactive Hidden SDH, where given (g, g^x, h, h^x, u) and access to an oracle $\mathcal{O}(c)$ that returns $g^{1/(x+c)}$, it is hard to compute a tuple $(g^{1/(x+c)}, h^c, u^c)$ for a $c \in \mathbb{Z}_p^*$ not queried to the oracle. Next, they present a new construction under the q-Hidden SDH and q-Triple DH assumptions. The q-Triple DH assumption states that given $(g, g^x, g^y, h, h^x, \{c_i, g^{1/(x+c_i)}\}_{i\in[1,q]})$, it is hard to compute a tuple $(h^{\mu x}, g^{\mu y}, g^{\mu xy})$, where $\mu \neq 0$. For both constructions, $F_{g,h}(m) = (g^m, h^m)$. While this work laid the foundation, its assumptions are both dynamic and very complex. Earlier, Groth [25] implicitly provided an F-signature for $F_g(m) = g^m \in \mathbb{G}$ by showing how to sign elements of $\mathbb{G}$ under the (static) Decision Linear assumption. As [25] observes, the scheme requires such huge constants that it is not practical.

In contrast, our F-signature construction based on the Hohenberger-Waters signatures [28] is very practical and requires only that given (g, g^a, g^b), it is hard to produce (h, h^{ab}) for $h \neq 1$, where $F_g(m) = g^m$. Like [28], these signatures are *stateful*, requiring that the signer keep a counter of the number of signatures issued. In our oblivious transfer protocol, we link the state of the signature to the identity of the database item. For item i, we remove the $\lceil \lg(i) \rceil$ exponent in the signature by observing that it will not matter if the adversary forges for "too large" identities, as they will be out of the range of the database items under attack.

D.2 Definition of Security

F-Signatures were formalized by Belenkiy, Chase, Kohlweiss and Lysyanskaya [1]. Their security notion extends (and implies) the standard definition [21] as follows: an adversary is given the public key and access to a signing oracle. It is successful in a forgery if, for some fixed efficiently-computable bijection F, it can output a pair $(\sigma, F(m))$ where σ is a valid signature on m and m was not queried to the oracle.

Definition D.1 (F-Secure Signature Scheme [21, 1]) *A signature scheme (G, S, V) is F-secure (against adaptive chosen message attacks) if it is correct and F-unforgeable.*

Correct. *V always accepts a signature obtained using the S algorithm.*

F-Unforgeable. *Let $F_{(\cdot)}$ be an efficiently-computable bijection. For every probabilistic polynomial time adversaries $\mathcal{A}$, there exists a negligible function μ such that*

$$\Pr[(pk, sk) \leftarrow G(1^\lambda); (y, \sigma) \leftarrow \mathcal{A}^{\mathcal{O}_{sign}(sk,\cdot)}(pk) : V(pk, F_{pk}^{-1}(y), \sigma) = 1 \land y \notin Q] \leq \mu(\lambda),$$

where Q is the set containing F_{pk} applied to all messages queried to $\mathcal{O}_{sign}$.

D.3 The Construction

We show that a modified version of the (stateful) signatures of Hohenberger and Waters [28] are also F-signatures. This requires an additional signature element and a new proof of security under a different assumption. Our bijection is $F_g(x) = g^x$, where g is a publicly known generator.

KeyGen(1^λ) The key generation algorithm selects a bilinear group $\mathbb{G}$ of prime order $p > 2^\lambda$. It next selects random values $a \in \mathbb{Z}_p$ and $g, u, v, d, w, z, h \in \mathbb{G}$. The public key is output as:

$$g, g^a, u, v, d, w, z, h.$$

The secret key $sk = a$ and the state counter is set as $i = 0$.

Sign($sk, i, M \in \mathbb{Z}_p$) The signer increments her counter i by one as $i = i + 1$. She next chooses random values $r, t \in \mathbb{Z}_p$ and then outputs the signature as:

$$\sigma_1 = (u^M v^r d)^a (w^{\lceil lg(i) \rceil} z^i h)^t, \quad \sigma_2 = g^t, \quad \sigma_3 = g^{aM}, \quad r, \quad i.$$

The message space is $\mathbb{Z}_p$, but could be enlarged using any collision resistant hash function.

Verify($pk, M, \sigma = (\sigma_1, \sigma_2, \sigma_3, r, i)$) The verification algorithm accepts if and only if $i < 2^\lambda$ and the following equations hold:

$$e(\sigma_1, g) = e(u^M v^r d, g^a) e(\sigma_2, w^{\lceil \lg(i) \rceil} z^i h), \quad e(\sigma_3, g) = e(g^M, g^a).$$

D.4 Proof of F-Unforgeability

The above F-signature is a practical scheme secure under a simple, static assumption. Specifically, we use the Flexible DH assumption formalized below. Most prior F-signatures require interactive or dynamic assumptions [1], or are impractical [25].

Assumption D.2 (Flexible Diffie-Hellman [31]) *For all p.p.t. adversaries* Adv*, there exists a negligible function* μ *such that:*
$\Pr[(p, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BMsetup}(1^\kappa); a, b \leftarrow \mathbb{Z}_p : (h, h^{ab}) \leftarrow \mathsf{Adv}(g, g^a, g^b) \wedge h \in \mathbb{G} \wedge h \neq 1] \leq \mu(1^\kappa).$

This assumption was previously described as the 2-out-of-3 CDH assumption by Kunz-Jacques and Pointcheval [31]. We adopt the name Flexible Diffie-Hellman for consistency with recent work [35, 19].

Theorem D.3 (F-Signatures Secure under Flexible DH) *Let* $F_g(x) = g^x$*, where* g *is chosen as above. The above signature scheme is* F*-secure under the Flexible Diffie-Hellman assumption.*

Messages are bound to their states. Our OT proofs actually require an additional security property from these F-signatures: that an adversary cannot produce a signature on a previously signed message with a different state than in its original signature. Indeed, this property is actually implicitly proven in [28] and the proof of Theorem D.3.

We now show that the Hohenberger-Waters signatures are F-signatures under the Flexible DH assumption, following their original proof outline [28].

Proof. Suppose we have an adversary that makes at most q signing queries, where q is polynomial in λ. We show that this adversary breaks Flexible DH. An adversary can have two types of forgeries.

Type I The adversary forges with an index i greater than $2^{\lceil \lg(q) \rceil}$.

Type II The adversary forges with an index i less than or equal to $2^{\lceil \lg(q) \rceil}$.

In Lemma D.4, we show that a type I adversary can be used to break Computational Diffie-Hellman (CDH) with a loss of a λ factor in the reduction. (CDH is clearly implied by Flexible DH.) In Lemma D.5, we show that a type II adversary can be used to break Flexible DH with a loss of a q factor in the reduction. This concludes the proof. □

D.4.1 Type I Adversary

Lemma D.4 *If a type I adversary succeeds with probability ϵ, then it can be used to solve CDH with probability ϵ/λ.*

Proof. Given a CDH challenge (g, g^a, g^b), proceed as follows. The Setup is the same as [28].

Setup The simulator begins by guessing a value k^* in the range 1 to λ. This represents a guess that the adversary will forge on index i such that $k^* = \lceil \lg(i) \rceil$. Next, choose random $y_u, y_v, y_z \in \mathbb{Z}_p$ and set $u = g^{y_u}, v = g^{y_v}, z = g^{y_z}$. Then set $d = g^b$, $w = g^b g^{x_w}$, and $h = g^{-bk^*} g^{x_h}$, for random $x_w, x_h \in \mathbb{Z}_p$. The simulator outputs the public key as $(g, g^a, u, v, d, w, z, h)$, sets the internal signing state $i = 0$, and implicitly designates the secret key as a.

Sign When the adversary asks for a signature on message $M \in \mathbb{Z}_p$, the simulator first updates its state value $i = i + 1$. If $k^* = \lceil \lg(i) \rceil$, the simulator aborts. Otherwise, it computes the signature by choosing random $r, t' \in \mathbb{Z}_p$, computing $k = \lceil \lg(i) \rceil$ and $T = g^{t'}/(g^a)^{1/(k-k^*)} = g^{t' - a/(k-k^*)}$, and outputting:

$$\sigma_1 = (g^a)^{y_u M} \cdot (g^a)^{y_v r} \cdot T^{x_w k + y_z i + x_h} \cdot (g^b)^{t'(k-k^*)} \quad , \quad \sigma_2 = T \quad , \quad \sigma_3 = g^{aM} \quad , \quad r \quad , \quad i.$$

Implicitly set the randomness $t = t' - a/(k - k^*)$ (here t' gives t the proper distribution) and we have $T = g^t$ and

$$\sigma_1 = (u^M v^r)^a \cdot (g^{x_w k} z^i g^{x_h})^t \cdot g^{bt'(k-k^*)} = (u^M v^r d)^a (w^k z^i h)^t \quad , \quad \sigma_2 = g^t.$$

Response Eventually, the type I adversary outputs a value β and a valid signature $\tilde{\sigma} = (\tilde{\sigma_1}, \tilde{\sigma_2}, \tilde{\sigma_3}, \tilde{r}, \tilde{i})$ on message $\tilde{M} = \log_g(\beta) \in \mathbb{Z}_p$ such that $\tilde{i} \geq 2q$. From the verification equation, we have that

$$e(\beta, g^a) = e(g, \sigma_3) \quad , \quad \text{implying that } \sigma_3 = g^{a\tilde{M}} \text{ and}$$
$$e(\tilde{\sigma_1}, g) = e(\sigma_4 v^{\tilde{r}} d, g^a) e(\tilde{\sigma_2}, w^{\lceil \lg(\tilde{i}) \rceil} z^{\tilde{i}} h)$$

Interpreting $\tilde{\sigma_2}$ as g^t, for some $t \in \mathbb{Z}_p$, it follows from the above equation that

$$\tilde{\sigma_1} = (u^{\tilde{M}} v^{\tilde{r}} d)^a (w^{\lceil \lg(\tilde{i}) \rceil} z^{\tilde{i}} h)^t.$$

Let $\tilde{k} = \lceil \lg(\tilde{i}) \rceil$. If $k^* \neq \tilde{k}$, the simulator aborts. If $k^* = \tilde{k}$, then the simulator guessed correctly and we know that

$$\begin{aligned}
\tilde{\sigma_1} &= (u^{\tilde{M}} v^{\tilde{r}} d)^a ((g^{b+x_w})^{\tilde{k}} (g^{y_z})^{\tilde{i}} (g^{-bk^*+x_h}))^t \\
&= (u^{\tilde{M}} v^{\tilde{r}} d)^a (g^{x_w \tilde{k}} g^{y_z \tilde{i}} g^{x_h})^t \\
&= (g^{\tilde{M} y_u} g^{\tilde{r} y_v} g^b)^a (g^{x_w \tilde{k}} g^{y_z \tilde{i}} g^{x_h})^t \\
&= g^{ab} (g^{\tilde{M} y_u} g^{\tilde{r} y_v})^a (g^{x_w \tilde{k}} g^{y_z \tilde{i}} g^{x_h})^t \\
&= g^{ab} (g^a)^{\tilde{M} y_u + \tilde{r} y_v} (g^t)^{x_w \tilde{k} + y_z \tilde{i} + x_h}
\end{aligned}$$

Thus, the simulator outputs $\tilde{\sigma_1} / (\sigma_3^{y_u} (g^a)^{\tilde{r} y_v} (g^t)^{x_w \tilde{k} + y_z \tilde{i} + x_h}) = g^{ab}$. □

D.4.2 Type II Adversary

Lemma D.5 *If a type II adversary succeeds with probability ϵ after making q signing queries, then it can be used to solve Flexible DH with probability $\epsilon/O(q)$.*

Proof. Given a Flexible DH challenge (g, g^a, g^b), proceed as follows. The Setup is the same as [28].

Setup The simulator begins by guessing an index i^* in the range 1 to $2^{\lceil \lg(q) \rceil}$. This represents a guess that the adversary will choose to forge on index i^*. Next, it chooses random $y_v, y_d, x_v, x_d \in \mathbb{Z}_p$ and sets $u = g^b$, $v = g^{bx_v} g^{y_v}$ and $d = g^{-bx_d} g^{y_d}$. Then it chooses random $y_w, x_z, x_h \in \mathbb{Z}_p$ and sets $w = g^{y_w}$, $z = g^b g^{x_z}$ and $h = g^{-bi^*} g^{x_h}$. The simulator outputs the public key as $(g, g^a, u, v, d, w, z, h)$, sets the internal signing state $i = 0$, and implicitly designates the secret key as a.

Sign When the adversary asks for a signature on message $M \in \mathbb{Z}_p$, the simulator first updates its state value $i = i + 1$. There are now two ways the simulator will proceed.

If $i = i^*$, then first compute $r = (x_d - M)/x_v$. Next, choose random $t \in \mathbb{Z}_p$ and set

$$\sigma_1 = (g^a)^{y_v r + y_d} \cdot (w^{\lceil \lg(i) \rceil} z^i h)^t \quad , \quad \sigma_2 = g^t \quad , \quad \sigma_3 = g^{aM}, \quad r \quad , \quad i.$$

To verify correctness, observe that we can rewrite σ_1 as follows given that $M + rx_v - x_d = 0$:

$$\begin{aligned}
\sigma_1 &= (g^{ab})^{M + rx_v - x_d} (g^a)^{y_v r + y_d} \cdot (w^{\lceil \lg(i) \rceil} z^i h)^t \\
&= (g^{bM} g^{(bx_v + y_v) r} g^{-bx_d + y_d})^a \cdot (w^{\lceil \lg(i) \rceil} z^i h)^t \\
&= (u^M v^r d)^a \cdot (w^{\lceil \lg(i) \rceil} z^i h)^t
\end{aligned}$$

If $i \neq i^*$, then choose random $r, t' \in \mathbb{Z}_p$, compute $T = g^{t' - a(M + x_v r - x_d)/(i - i^*)}$, and output:

$$\sigma_1 = (g^a)^{y_v r + y_d} \cdot T^{y_w \lceil \lg(i) \rceil + x_z i + x_h} \cdot (g^b)^{t'(i - i^*)} \quad , \quad \sigma_2 = T \quad , \quad \sigma_3 = g^{aM}, \quad r \quad , \quad i.$$

Let us implicitly set the randomness $t = t' - a(M + x_v r - x_d)/(i - i^*)$ (here t' gives t the proper distribution) and we have $T = g^t$ and

$$\sigma_1 = (g^{y_v r} g^{y_d})^a \cdot (w^{\lceil \lg(i) \rceil} g^{x_z i} g^{x_h})^t \cdot (g^b)^{t'(i - i^*)} \quad , \quad \sigma_2 = g^t.$$

Response Eventually, the type II adversary outputs a value β and a valid signature $\tilde{\sigma} = (\tilde{\sigma_1}, \tilde{\sigma_2}, \tilde{\sigma_3}, \tilde{r}, \tilde{i})$ on a message $\tilde{M} = \log_g(\beta) \in \mathbb{Z}_p$ such that $\tilde{i} < 2^{\lceil \lg(q) \rceil}$. Let $\tilde{\sigma_2} = g^t$ for some $t \in \mathbb{Z}_p$. Now, from the verification equation, we see that

$$\tilde{\sigma_1} = (u^{\tilde{M}} v^{\tilde{r}} d)^a (w^{\lceil \lg(\tilde{i}) \rceil} z^{\tilde{i}} h)^t.$$

If $i^* = \tilde{i}$, then the simulator guessed correctly and we know that

$$\begin{aligned}
\tilde{\sigma_1} &= ((g^b)^{\tilde{M}} (g^{bx_v + y_v})^{\tilde{r}} (g^{-bx_d + y_d}))^a ((g^{y_w})^{\lceil \lg(\tilde{i}) \rceil} (g^{b+x_z})^{\tilde{i}} (g^{-bi^* + x_h}))^t \\
&= g^{ab(\tilde{M} + x_v \tilde{r} - x_d)} g^{a(y_v \tilde{r} + y_d)} ((g^{y_w})^{\lceil \lg(\tilde{i}) \rceil} (g^{b+x_z})^{\tilde{i}} (g^{-bi^* + x_h}))^t \\
&= g^{ab(\tilde{M} + x_v \tilde{r} - x_d)} g^{a(y_v \tilde{r} + y_d)} (g^{y_w \lceil \lg(\tilde{i}) \rceil} g^{x_z \tilde{i}} g^{x_h})^t \\
&= g^{ab(\tilde{M} + x_v \tilde{r} - x_d)} g^{a(y_v \tilde{r} + y_d)} g^{t(y_w \lceil \lg(\tilde{i}) \rceil + x_z \tilde{i} + x_h)}
\end{aligned}$$

As in [28], the probability that $(\tilde{M} + x_v \tilde{r} - x_d) = 0$ is $1/p$, in which case the simulator must abort. If $(\tilde{M} + x_v \tilde{r} - x_d) \neq 0$, the simulator outputs the Flexible DH solution (h', h'^{ab}) as

$$\begin{aligned}
h' &= g^{(\tilde{M} + x_v \tilde{r} - x_d)} \\
h'^{ab} &= \frac{\tilde{\sigma_1}}{g^{a(y_v \tilde{r} + y_d)} \tilde{\sigma_2}^{(y_w \lceil \lg(\tilde{i}) \rceil + x_z \tilde{i} + x_h)}} = g^{ab(\tilde{M} + x_v \tilde{r} - x_d)}
\end{aligned}$$

□

D.5 Flexible DH is implied by 3DDH

Recall the 3DDH assumption from Section 2. We now show that when set in a symmetric bilinear group, (decisional) 3DDH implies (computational) FDH.

Lemma D.6 (3DDH implies Flexible DH) *If 3DDH holds in bilinear group $\mathbb{G}$, then Flexible DH also holds in $\mathbb{G}$.*

Proof sketch. If a Flexible DH solver $\mathcal{A}$ succeeds with probability ϵ, then we can construct an 3DDH solver $\mathcal{B}$ that succeeds with probability ϵ minus a negligible amount. On input an 3DDH instance (g, g^a, g^b, g^c, Q), $\mathcal{B}$ proceeds as follows: Give input (g, g^a, g^b) to $\mathcal{A}$ to obtain a pair (x, y). We note that if $\mathcal{A}$ is successful, then $y = x^{ab}$. Output 1 if $e(x, Q) = e(y, g^c)$ and 0 otherwise. □

Corollary D.7 (F-Signatures Secure under 3DDH) *Let $F_g(x) = g^x$, where g is chosen as above. The above signature scheme is F-secure under the 3DDH assumption.*