How to Avoid Obfuscation Using Witness PRFs

Mark Zhandry(✉)

Massachusetts Institute of Technology, Cambridge, MA, USA
mzhandry@gmail.com

Abstract. We propose a new cryptographic primitive called *witness pseudorandom functions* (witness PRFs). Witness PRFs are related to witness encryption, but appear strictly stronger: we show that witness PRFs can be used for applications such as multi-party key exchange without trusted setup, polynomially-many hardcore bits for any one-way function, and several others that were previously only possible using obfuscation. Thus we improve the minimal assumptions required for these applications. Moreover, current candidate obfuscators are far from practical and typically rely on unnatural hardness assumptions about multilinear maps. We give a construction of witness PRFs from multilinear maps that is simpler and much more efficient than current obfuscation candidates, thus bringing several applications of obfuscation closer to practice. Our construction relies on new but very natural hardness assumptions about the underlying maps that appear to be resistant to a recent line of attacks.

Keywords: Witness PRFs · Multilinear maps · Multiparty key exchange

1 Introduction

Program obfuscation is the act of "scrambling" a program such that the functionality is preserved, but the inner workings of the program are completely hidden even given the scrambled code. Recently, Garg et al. [GGH+13b] proposed the first construction of a general purpose program obfuscator [GGH+13b], which has sparked significant advances in cryptographic capabilities. Obfuscation has been used to construct a plethora of surprising and powerful cryptographic applications, including functional encryption [GGH+13b], deniable encryption [SW14], multiparty non-interactive key agreement [BZ14], multiparty computation in very few rounds [GGHR14], and much more. Thus, obfuscation is a "heavy hammer" by which, it seems, most of cryptography can be built. This leads to a natural question:

To what extent is obfuscation actually needed for various applications?

M. Zhandry—Work done while the author was a graduate student at Stanford University. Supported by the DARPA PROCEED program.

This is a very important question, as using obfuscation for applications has some major drawbacks. For one, current candidate obfuscators [GGH+13b, BR14, BGK+14, PST14, AGIS14, GLSW14, SZ14, Zim15, AB15] are incredibly inefficient, to the point that they are utterly *unimplementable* for all except the simplest of functionalities. This is even despite significant improvements in efficiency obtained by several recent works. Second, we do not know how to base the security of obfuscation on any traditional assumptions, but must instead make strong new assumptions on multilinear maps.

Therefore, obfuscation is likely too general of a tool for practical protocols with reasonable underlying security assumptions. Instead, obfuscation serves as a proof of concept, showing that a particular application is plausible. Then, more application-specific tools and techniques are required to actually obtain a usable protocol.

This Work. In this work, we make progress toward answering the above question by showing that obfuscation is *not* necessary for several applications. We do this by introducing a new technical tool called *witness pseudorandom functions (witness PRFs)* that abstracts an obfuscation technique used by several recent applications. We show that witness PRFs maintains enough of the power of obfuscation that it can still be used for these tasks, which were only previously possible using obfuscation. We also give a very simple construction of witness PRFs using multilinear maps that is significantly more efficient that current obfuscators. For security, our construction relies on new assumptions on the underlying maps. Our assumptions are very simple, and we argue that they are in some ways "better" than the assumptions on which obfuscation is based.

While applications of our witness PRFs remain impractical and security is still based on relatively untested multilinear map assumptions, our work provides a significant step towards improving the efficiency of some applications of obfuscation, and potentially towards basing applications on better assumptions. Moreover, our work provides a more refined view of the cryptographic landscape by showing that weaker primitives suffice for some applications.

1.1 Motivating Example: Non-interactive Key Exchange Without Setup

We motivate the following discussion using a specific application of obfuscation: multiparty non-interactive key exchange (NIKE) without trusted setup. In such a protocol protocol, n users each generate a secret and public value and simultaneously publishes their public values to a public bulletin board. All of the users then read off the values from the bulletin board and are each able to derive the same shared key with no further interaction. Non-interaction is crucial to obtaining a *re-usable* protocol: $N \gg n$ users can each publish their public value, and then at a later point *any* subset of n of them can establish a shared secret key *without any additional interaction.* In contrast, in an interactive scheme, the protocol needs to be carried out once *for every* subset of users that wishes to derive a key.

The first key exchange protocol for $n = 2$ users is the celebrated Diffie-Hellman protocol. Joux [Jou04] shows how to use pairings to extend this to $n = 3$ users. Boneh and Silverberg [BS02] generalize Joux's work to obtain multiparty NIKE for arbitrary n from (symmetric) multilinear maps as follows. Recall that a symmetric n-linear map consists of a source group $\mathbb{G}$ with generator g of order p, a target group $\mathbb{G}_T$ with generator g_T of order p, and a multilinear "pairing" operation $e : \mathbb{G}^n \to \mathbb{G}_T$ with the property that $e(g^{a_1}, g^{a_2}, \ldots, g^{a_n}) = g_T^{a_1 a_2 \ldots a_n}$. We call n the multilinearity of the map. Ideally any operation except the group and pairing operations should be computationally infeasible. Using an n-linear map, the Boneh-Silverberg protocol for $n+1$ users is as follows: user i chooses a random $a_i \in \mathbb{Z}_p$, and publishes $h_i = g^{a_i}$. The shared secret is $K = g_T^{a_1 a_2 \ldots a_{n+1}}$. User i can compute K as $e(h_1, h_2, \ldots, h_{i-1}, h_{i+1}, \ldots, h_{n+1})^{a_i}$ by pairing the other n public values, and then exponentiating by her secret value. However, an eavesdropper that only sees the h_i would have to pair all $n + 1$ of the public values to obtain K, but the pairing operation only supports pairing n elements together. Security can be proved based on the *multilinear DDH* assumption, a natural generalization of the DDH assumption to the multilinear setting, and one of the most basic assumptions made on multilinear maps.

Garg, Gentry, and Halevi [GGH13a] give the first candidate multilinear map construction, thus giving the first multiparty NIKE protocol. However, in their construction, generating g and e requires secrets, knowledge of which completely breaks any security of the maps. The protocol is therefore only non-interactive in a *trusted* setup model, where setup must be performed by a central authority, and the authority will also be able to learn the shared key. Moreover, since g is needed for users to compute their public value, the setup must take place *before* the protocol is carried out. The need for a trusted central authority is a serious limitation of the protocol, and also for all protocols for $n > 3$ users prior to the obfuscation-based protocol we explain next.

Multiparty NIKE Without Setup. Boneh and Zhandry [BZ14] show how to use obfuscation to remove the setup phase entirely. In their protocol, each party generates a seed s_i of length λ for a pseudorandom generator G with output size 2λ, and publishes the corresponding output x_i. In addition, a designated master party (say, party 1) chooses a random key fk for a PRF F, and builds the following program P:

- On input $x_1, \ldots, x_n, s, i$, check that $\mathsf{G}(s) = x_i$.
- If the check fails, output $\perp$. Otherwise, output $\mathsf{F}(\mathsf{fk}, x_1, \ldots, x_n)$.

The master party then publishes an obfuscation of P along with their public x_i. Each party i can now compute $K = \mathsf{F}(\mathsf{fk}, x_1, \ldots, x_n)$ by feeding $x_1, \ldots, x_n, s_i, i$ into the obfuscation of P. Thus, all parties establish the same shared key K. An eavesdropper meanwhile only gets to see the obfuscation of P and the x_i, and tries to determine K. He can do so in one of two ways: either run the obfuscation of P on inputs of his choice, hoping that one of the outputs is K, or inspect the obfuscated code of P to try to learn K.

The one-wayness of G means the first approach is not viable. Boneh and Zhandry show that when using an "indistinguishability" obfuscator and "puncturable" PRF, the value of K is still hidden, even if the adversary inspects the obfuscated code for P. The proof works roughly as follows: first, all of the public values x_i are replaced with truly random strings. The security of G shows that this change is undetectable. Then, since G is expanding, with high probability, none of the x_i have pre-images under G. This means there is no input to the program P that causes it to pass the check and output $K = \mathsf{F}(\mathsf{fk}, x_1, \ldots, x_n)$. Then, using indistinguishability obfuscation and the puncturing property of F, it is possible to show the adversary learns no information about K.

Implementing the Boneh Zhandry Protocol. There are two ways to instantiate the obfuscator in the protocol above using multilinear maps:

- Directly on a "core obfuscator" for shallow circuits. The multilinearity required for the underlying map will be approximately 2^d for input circuit of depth d. This presents a serious implementation barrier, as parameters in current multilinear maps grow polynomially with the multilinearity. In an asymptotic sense, using circuits of logarithmic depth will result in polynomial-sized programs. In the case of the Boneh-Zhandry protocol, the bottleneck is clearly the PRF. While there exist puncturable PRFs that are computable in log-depth (for example, it is folklore that the Naor-Reignold PRF is puncturable), the constant term is moderate. Thus, if the depth of the PRF is, say $c \log(2n\lambda)$ ($2n\lambda$ being roughly the input size to the PRF), the resulting program requires multilinearity at least $(2n\lambda)^c$, a polynomial. However, for even moderate c, this polynomial becomes extremely large.
- By boosting the "core obfuscator" to a general obfuscator for all circuits. Depending on the conversion used, this at best requires obfuscating a low-depth PRF [App13] with the core obfuscator anyway, and at worst obfuscating the decryption function of a fully homomorphic encryption scheme [GGH+13b]. Therefore, this approach seems unlikely to yield significant improvements.

In terms of security, current obfuscators can be separated into two categories:

- Schemes with heuristic security. This includes the first candidate scheme of Garg et al. [GGH+13b] as well as several subsequent constructions [BR14, BGK+14, PST14, AGIS14, SZ14, Zim15, AB15]. Some of these schemes can be proven secure in idealized models of computation [BR14, BGK+14, AGIS14, SZ14, Zim15, AB15], but such a proof does not translate into a standard model proof under any assumptions. Thus for these constructions, the security assumption is "tautological" and basically matches the scheme. While there has been significant progress towards simplifying obfuscation, these candidates still are complicated and require several techniques (straddling sets, Kilian randomization, etc.) that yield unnatural security assumptions.
- Schemes with security proved relative to a "nice" assumption. There are basically two examples. The first is a construction due to Pass, Seth, and

Telang [PST14], based on the "semantic security" assumption on multilinear maps. Unfortunately, this conjectured assumption is an "uber assumption" that is so general that it comes close assuming the scheme itself is secure. The second is a construction of Gentry et al. [GLSW14] based on a single assumption, the multilinear subgroup elimination (MSE) assumption. While this is a significant advance in terms of basing the security of obfuscation on better assumptions, there are some notable drawbacks. First, the assumption requires introducing subgroups, which complicates the scheme and makes it less efficient. Second, the MSE assumption is a "source group" assumption on multilinear maps, which has proven very problematic on current map candidates. In particular, the MSE assumption is broken on all other multilinear maps due to a recent line of attacks [CHL+14, GHMS14, BWZ14b, CLT14][1]. Finally, the proof uses complexity leveraging, which seems inherent to basing obfuscation on simple assumptions [GGSW13]. This means that, for the proof to hold, the security parameter must be set quite large, compounding the efficiency issues above. Thus the most efficient obfuscators are likely to require complicated "tautological" security assumptions.

Thus, we pay a very steep price for eliminating the setup, both in terms of efficiency and in terms of assumptions. Using multilinearity as a proxy for efficiency, we see that the multilinearity for an n-user protocol increases from $n-1$ to $(2n\lambda)^c$ for a moderate constant c. Moreover, whereas the security of the basic multilinear map protocol is based on the very simple MDDH assumption, the setupless protocol requires somewhat more complicated assumptions. Outside of this work, all setupless key exchange protocols (even in subsequent work [HJK+14, Rao14]) require obfuscation, and therefore suffer from these weaknesses.

1.2 Our Contributions: Witness PRFs

Abstracting the Needed Functionality. We now ask, what features of obfuscation are needed for setupless key exchange? Observe that we do not need to hide the entire program P in the protocol: for example, the entire computation up until the PRF can be leaked. Thus, we do not necessarily need the full power of obfuscation. In fact, obfuscation is used in a very particular way:

– First, the input is separated into two parts. The first part, the "instance", consists of the $y_1, \ldots, y_n$. The second part, the "witness" or "token", consists of s, i.
– The program has the following structure: check some relation between the instance and witness and then apply a PRF to the instance (but not the witness) if the check passes.

[1] These assumptions are only broken on these maps if certain "re-randomization" terms are published, and it is possible to state the MSE assumption without these terms in which case the assumption may hold on *all* candidate multilinear maps. The obfuscator of [GLSW14] does not rely on such re-randomization parameters, but the security proof *does* need the parameters. Hence, the form of the assumption needed to prove security *is* broken.

– The security we desire is that if the instance has no witness, no information about the output of the PRF value at that input is revealed.

Thus, obfuscation is acting as an access control to the PRF, only allowing evaluation at a point if the user can supply a valid token.

Witness PRFs. We now define our new primitive called *witness pseudorandom functions* (witness PRFs) that captures the functionality and security properties needed above. Informally, a witness PRF for an NP language L is a PRF F such that anyone with a valid witness that $x \in L$ can compute $\mathsf{F}(x)$ without the secret key, but for all $x \notin L$, $\mathsf{F}(x)$ is computationally hidden without knowledge of the secret key. More precisely, a witness PRF consists of the following three algorithms:

– $\mathsf{Gen}(L, n)$ takes as input (a description of) an NP language L and instance length n (and implicitly a security parameter), and outputs a secret function key fk and public evaluation key ek.
– $\mathsf{F}(\mathsf{fk}, x)$ takes as input the function key fk, an instance $x \in \{0,1\}^n$, and produces an output y.
– $\mathsf{Eval}(\mathsf{ek}, x, w)$ takes the evaluation key ek, and instance x, and a witness w that $x \in L$, and outputs $\mathsf{F}(\mathsf{fk}, x)$ if w is a valid witness, $\perp$ otherwise.

For security, we require that for any $x \in \{0,1\}^n \setminus L$, the value $\mathsf{F}(\mathsf{fk}, x)$ is pseudorandom even given ek. In Sect. 3, we also consider many variants of this definition. For example, an interactive variant allows the adversary to make polynomially many PRF queries to $\mathsf{F}(\mathsf{fk}, \cdot)$, and still requires that $\mathsf{F}(\mathsf{fk}, x)$ is indistinguishable from random (conditioned, of course, on x not being one of the PRF queries). We also define an extractable variant that allows $x \in L$, but if the adversary can distinguish $\mathsf{F}(\mathsf{fk}, x)$ from random, then the adversary must "know" a witness that $x \in L$.

Witness PRFs are closely related to the concept of smooth projective hash functions (a comparison is given in Sect. 1.5), and can be seen as a generalization of constrained PRFs [BW13,KPTZ13,BGI14] to arbitrary NP languages[2].

We first show how to replace obfuscation with witness PRFs for certain applications, including a no-setup multiparty key exchange protocol. We then show how to build witness PRFs from multilinear maps. Our witness PRFs are more efficient than current obfuscation candidates, and rely on very natural, though new, assumptions about the underlying maps. We stress that all of our applications can be instantiated using obfuscation, and the applications are therefore not "new." However, instantiating the applications with witness PRFs result in significant efficiency improvements compared to obfuscation. Our witness PRFs

[2] This is not strictly true, as constrained PRFs generate the secret function key independent of any language and multiple evaluation keys can be generated for multiple languages. Witness PRFs, on the other hand, only permit one evaluation key, and the language for the key must be known when the function key is generated. In the full version [Zha14b], we discuss how to obtain *multi-relation* witness PRFs which get around these issues.

rely on assumptions that appear to be weaker than those needed for obfuscation, and are qualitatively better in several ways. Our assumptions are very natural and simple, and while they essentially match the security of a component of our scheme, that component is much simpler than current obfuscation candidates. Our assumptions are also a very restricted case of the semantic security [PST14] assumption on multilinear maps, and do not seem general enough to imply obfuscation. Lastly, our assumption is a "target group" assumption, which appear to be more resilient to recent attacks on multilinear maps, whereas all assumptions required for obfuscation are "source group" assumptions (more details below).

Therefore, our work can be seen as (1) improving the minimal assumption under which several applications are possible and (2) providing significant efficiency improvements for those applications.

Our Results. Below, we list our main results:

– We show how to realize the following primitives from witness PRFs
  - **Multiparty Non-Interactive Key Exchange (NIKE) Without a Trusted Setup** (Sect. 5.2). We give a construction closely related to the Boneh-Zhandry [BZ14] protocol, where the obfuscator is replaced with a witness PRF, and prove that security still holds.
  - **Poly-Many Hardcore Bits**. Bellare, Stepanovs, and Tessaro [BST14] construct a hardcore function of arbitrary output size for any one-way function. They require differing inputs obfuscation [BGI+01,BCP14,ABG+13], which is a form of knowledge assumption for obfuscators. In the full version [Zha14b], we show how to replace the obfuscator with a witness PRF that satisfies our extractable notion of security.
  - **Reusable Witness Encryption**. In witness encryption, messages are encrypted to instances x of some NP language L, and any user that knows a witness that $x \in L$ can decrypt the ciphertext. Security says that if $x \notin L$, the ciphertext reveals no information about the plaintext. Garg, Gentry, Sahai, and Waters [GGSW13] define and build the first witness encryption scheme from multilinear maps. Later, Garg et al. [GGH+13b] show that indistinguishability obfuscation implies witness encryption. In the full version [Zha14b], we show that witness PRFs are actually sufficient, showing that witness PRFs are essentially a generalization of witness encryption.
    We also define a notion of re-usability for witness encryption, and give a construction from witness PRFs. Our re-usable witness encryption scheme has very short ciphertexts: namely proportional to the security parameter and independent of the size of the relation. Combining with the witness encryption-to-attribute-based encryption conversion of Garg et al. [GGSW13], this allows us to build attribute-based encryption (ABE) for circuits with similarly short ciphertexts (namely independent of the size of the access policy). No other ABE construction with such succinct ciphertexts is known without using obfuscation; it is not known how to construct such an ABE scheme from the (non-reusable) witness encryption scheme of [GGSW13].

  - **Rudich Secret Sharing for mNP.** Rudich secret sharing is a generalization of secret sharing to the case where the sets of "qualified" users correspond to instances of a monotone NP (mNP) language L. In other words, n users are each given a share of a secret s. Any set $S \subseteq [n]$ of users corresponds to an instance $x \in \{0,1\}^n$, and if the users in S know a witness that $x \in L$, they can collectively reconstruct the secret using their shares. However, if $x \notin L$, the secret remains hidden. Monotonicity implies that adding users to a qualified set S does not affect the ability of S to compute the secret. Komargodski, Naor, and Yogev [KNY14] give the first construction for all of mNP using witness encryption[3]. In the full version [Zha14b], we give a related protocol using witness PRFs that is reusable, which results in much shorter shares than in [KNY14].
  - **Fully Distributed Broadcast Encryption.** In broadcast encryption, n users each have a user-specific secret key, and anyone can encrypt a message to an arbitrary subset $S \subseteq [n]$ of users. Each user in S can decrypt using their individual secret, but users outside of S, even if they all collude, learn nothing about the message. The measures of interest for broadcast encryption are the sizes of the ciphertext, user secret keys, and public broadcast key as a function of the number of users n. Boneh and Zhandry [BZ14] observe that multiparty NIKE protocols with small messages give rise to broadcast encryption with constant-size ciphertexts and secret keys, but with large public keys. The resulting scheme has the novel property of being distributed, where users generate their own secret keys. In Boneh and Zhandry's notion of distributed broadcast encryption, the large public keys are inherent because there is a component of the public key corresponding to each user. In the full version [Zha14b], we put forward a new notion of *fully distributed* broadcast encryption which does not suffer from this issue, and give a construction from our extractable notion of witness PRFs where secret keys, public keys, and ciphertexts are all poly-logarithmic in n. Our scheme even obtains the strong notion of adaptive security[4]. We note that our construction could have been instantiated using (extractable) witness encryption, but witness PRFs give a protocol with better parameters.

– Next, we show how to build witness PRFs from multilinear maps. We first define an intermediate notion of a subset-sum encoding, and construct such encodings from multilinear maps. Our construction is very simple, and we argue security based on new assumptions on multilinear maps. While our assumptions basically match the security of the subset-sum encodings, the assumptions are very simple and natural due to the simplicity of our scheme. Our full construction is given in Sect. 4.

[3] Originally, [KNY14] used obfuscation, but in a later update showed that witness encryption was sufficient.

[4] Of course, obtaining adaptive security from an interactive assumption is not that interesting. However, our construction relies only on a *non*-interactive variant. Therefore, obtaining adaptive security is non-trivial.

In the full version [Zha14b], we then show how to build witness PRFs from subset-sum encodings. The resulting construction is much more efficient that what is currently possible with obfuscation. In particular, we can build witness PRFs for arbitrary relations directly without the costly boosting step required for obfuscation. The multilinearity required for the underlying multilinear maps is roughly equal to the size of the circuit defining the relation, rather than exponential in the depth, as in current obfuscators. While implementing our construction is still impractical for all except the most basic relations, future research in improving the efficiency of multilinear maps will bring our construction closer to practice.

– Finally, in the full version [Zha14b] we discuss how to obtain a multi-language variant of witness PRFs, where multiple evaluation keys ek_{L_i} corresponding to multiple language L_i can be produced. A witness for x relative to any of the L_i can be used to evaluate the PRF on x, and if $x \notin L_i$ for any i, then the value of the PRF on x is pseudorandom. We do not need such multi-language witness PRFs for any of our applications, but we believe they are an interesting object, and may be useful in other situations.

1.3 Techniques

Secure Subset-Sum Encodings. As a first step to building witness PRFs, we construct a primitive called a subset-sum encoding. Roughly, such an encoding corresponds to a (multi-)set S of n integers, and consists of a secret encoding function which maps integers t into encodings $\hat{t}$. Additionally, there is a public evaluation function which takes as input a subset $T \subseteq S$, and can compute the encoding $\hat{t}$ of the sum of the elements in T: $t = \sum_{i \in T} i$. For security, we ask that for any t that does not correspond to a subset-sum of elements of S, the encoding $\hat{t}$ is indistinguishable from a random element.

We provide a simple candidate subset-sum encoding from asymmetric cryptographic multilinear maps. We use asymmetric maps, though it is straightforward to adapt our protocol to the symmetric setting. Recall that in an asymmetric n-linear map, instead of a single source group $\mathbb{G}$, there are n source groups $\mathbb{G}_1, \ldots, \mathbb{G}_n$ with generators $g_1, \ldots, g_n$, and the pairing operation only allows for one element from each group. That is, $e : \mathbb{G}_1 \times \cdots \times \mathbb{G}_n \to \mathbb{G}_T$ where[5]

$$e(g_1^{a_1}, g_2^{a_2}, \ldots, g_n^{a_n}) = g_T^{a_1 a_2 \ldots a_n}.$$

To generate a subset-sum encoding for a collection $S = \{v_1, \ldots, v_n\}$ of n integers, choose a random $\alpha \xleftarrow{R} \mathbb{Z}_p$, and compute $V_i = g_i^{\alpha^{v_i}}$ for $i = 1, \ldots, n$. Publish each V_i, while α is kept secret.

[5] This is the asymmetric variant of the multilinear map notion proposed by Boneh and Silverberg [BS02]. Current multilinear map candidates actually support a much richer set of operations, but our construction does not require this additional structure.

The encoding of a target integer t is $\hat{t} = g_T^{\alpha^t}$. Given the secret α it is easy to compute $\hat{t}$[6]. Moreover, if $t = \sum_{i \in T} i$ for some subset $T \subseteq S$, then given the public values V_i, it is also easy to compute $\hat{t}$ using the multilinear operation: define $V_{i,1} = V_i$ and $V_{i,0} = g_i$ so that $V_{i,b} = g_i^{\alpha^{bv_i}}$. Then set b_i to be the indicator function for $i \in T$ (so that $t = \sum_{i \in [n]} b_i v_i$) and compute

$$\hat{t} = e(V_{1,b_1}, \ldots, V_{n,b_n}) = e(g_1^{\alpha^{b_1 v_1}}, \ldots, g_n^{\alpha^{b_n v_n}}) = g_T^{\left(\alpha^{\sum_{i \in [n]} b_i v_i}\right)} = g_T^{\alpha^t}$$

However, if t cannot be represented as a subset-sum of elements in S, then the multilinear map operations do not allow for computing $\hat{t}$: there is no way to pair or multiply the V_i and g_i together so that the result is $\hat{t}$. We conjecture that in this case, $\hat{t}$ is hard to compute. This gives rise to a new complexity assumption on multilinear maps: we say that the *multilinear subset-sum Diffie-Hellman assumption* holds for a multilinear map if, for any set of integers $S = \{v_1, \ldots, v_n\}$ and any target t that cannot be represented as a subset-sum of elements in S, that $g_T^{\alpha^t}$ is indistinguishable from a random group element, even given the elements $\{g_i^{\alpha^{v_i}}\}_{i \in [n]}$[7]. In the full version [Zha14b], we show that this assumption holds in a generic model of multilinear maps, the same model that has been used to argue the security of current obfuscators [BR14, BGK+14]. We leave for future work the problem of proving security in the more refined generic model of Gentry et al. [GHMS14], which captures the recent line of "zero-izing" attacks. However, while we do not prove security in the zero-izing model, we stress that these attacks do not appear to apply to our assumptions.

Our assumption can be seen as an "uber-assumption", containing exponentially-many assumptions, one per SubsetSum instance (S, t). For example, setting S to be $\{1, 2, 3\}$ and t to be -1, our assumption states that $g_T^{\alpha^{-1}}$ is indistinguishable from random, given the elements $\{g_1^{\alpha^1}, g_2^{\alpha^2}, g_3^{\alpha^3}\}$. The assumptions in this family have the flavor of several existing assumptions on bilinear and multilinear maps, such as the Diffie-Hellman inversion and Diffie-Hellman Exponent assumptions.

Notice that the element that must be distinguished from random, namely $g_T^{\alpha^t}$, is in the target group $\mathbb{G}_T$. Therefore, our assumption is a *target-group* assumption, which appear more plausible on currently multilinear map candidates than *source-group* assumptions involving only elements in the groups $\mathbb{G}_1, \ldots, \mathbb{G}_n$. Indeed, the focus of recent attacks [CHL+14, GHMS14, BWZ14b, CLT14] is usually the source-group assumptions. For all current obfuscators, the assumption

[6] Current multilinear map candidates do not allow all users to perform exponentiation by arbitrary elements of $\mathbb{Z}_p$, which makes computing V_i and $\hat{t}$ potentially problematic. However, whomever sets up the subset-sum encoding will also set up the multilinear map, and will thus have a trapdoor that *does* allow computing V_i and $\hat{t}$. Therefore, the secret key should also include this trapdoor along with α.

[7] We can also use an even stronger assumption that also allows the adversary to adaptively ask for values $g_T^{\alpha^{t'}}$ for $t' \neq t$. This will result in a stronger security guarantee for the subset-sum encodings and our derived witness PRFs.

that the scheme itself is secure is a source-group assumption, so while the recent line of attacks does not appear to break current obfuscators, the attacks do decrease our confidence in their security. Target-group assumptions, on the other hand, appear much more resistant to attack.

Application to Witness Encryption. Recall that in a witness encryption scheme as defined by Garg et al. [GGSW13], a message m is encrypted to an instance x, which may or may not be in some NP language L. Given a witness w that $x \in L$, it is possible to decrypt the ciphertext and recover m. However, if $x \notin L$, m should be computationally hidden.

Our subset-sum encodings immediately give us witness encryption for the language L of SubsetSum instances. Let (S, t) be a SubsetSum instance. To encrypt a message m to (S, t), generate a subset-sum encoding for set S. Then, using the secret encoding algorithm, compute $\hat{t}$. The ciphertext is the public evaluation function, together with $c = \hat{t} \oplus m$. To decrypt using a witness subset $T \subseteq S$, use the public evaluation procedure on T to obtain $\hat{t}$, and then XOR with c to obtain m. If $(S, T) \notin$ SubsetSum, then the security of our subset-sum encoding implies that $\hat{t}$, and hence m, is hidden from the adversary.

Since SubsetSum is NP-complete, we can use NP reductions to obtain witness encryption for any NP language L. Our scheme may be more efficient than [GGSW13] for languages L that have simpler reductions to SubsetSum than to the ExactCover problem used by [GGSW13]. For example, the language L_{LWE} of learning-with-errors instances admits a very simple algebraic reductions to SubsetSum. Also, while our assumptions are new, they are no more or less plausible than the assumptions used in [GGSW13].

We can also obtain a special case of Rudich secret sharing. Given a SubsetSum instance (S, t), compute the elements $V_i, \hat{t}$ as above, and compute $c = \hat{t} \oplus s$ where s is the secret. Hand out share (V_i, c) to user i. Notice that a set U of users can learn s if they know a subset $T \subseteq U$ such that $\sum_{j \in T} j = t$. If no such subset exists, then our subset-sum Diffie-Hellman assumption implies that s is hidden from the group U of users.

Witness PRFs for NP. As defined above, witness PRFs are PRFs that can be evaluated on any input x for which the user knows a witness w that $x \in L$. For any $x \notin L$, the value of the PRF remains computationally hidden. Notice that subset-sum encodings *almost* give us witness PRFs for the SubsetSum problem. Indeed, the setup algorithm for a subset-sum encoding only depends on the subset S of integers, and not the target value t. Thus, a subset-sum encoding for a set S gives us a witness PRF for the language L_S of all integers t that are subset-sums of the integers in S.

To turn a subset-sum encoding into a witness PRF for an arbitrary language, we give a reduction from any NP language L to SubsetSum with the following property: the set S is independent of the instance x itself, but is instead determined entirely by the NP relation defining L (and the instance length). The instance x instead only affects the target t. Therefore, to build a witness PRF for any fixed NP relation R, run our reduction algorithm to obtain a set S_R, and then build a subset-sum encoding for S_R.

A notable feature of our resulting witness PRF is that its efficiency is comparable to that of existing witness encryption schemes for general relations R. In particular, the level of multilinearity required and the number of group elements in the evaluation key are equal to the size of the set S_R, which is roughly equal to the number of gates in R. The original witness encryption scheme of Garg et al. [GGSW13] required the level of multilinearity and the number of ciphertext group elements to roughly correspond to the ExactCover instance size, which similarly grows linearly with R. Therefore, we get the added functionality of witness PRFs essentially "for free" in terms of efficiency.

Replacing Obfuscation with Witness PRFs. We return to attention to multiparty non-interactive key exchange without setup to demonstrate how witness PRFs can be used in place of obfuscation.

We now explain how witness PRFs actually suffice for this application. As in the Boneh-Zhandry protocol, each user chooses a random seed s_i for the PRG G, and publishes the output x_i. Simultaneously, we define an NP language L consisting of all tuples $(x_1, \ldots, x_n)$ where at least one of the x_i has a pre-image under G. Instead of obfuscating a program, the master party can simply produce a witness PRF F for the language L, and publishes the corresponding evaluation key ek. All users then set the shared key to be $\mathsf{F}(\mathsf{fk}, x_1, \ldots, x_n)$, which all the honest parties can compute using ek since they know a witness.

To argue security, as in the Boneh-Zhandry protocol we replace the x_i with random elements, and rely on the security of G to show that this change is undetectable. Then with overwhelming probability none of the x_i have pre-images under G. This means that with overwhelming probability $(x_1, \ldots, x_n)$ is no longer in L. Therefore, the security of the witness PRF shows that the value $K = \mathsf{F}(\mathsf{fk}, x_1, \ldots, x_n)$ is computationally indistinguishable from a random string, as desired.

Notice that the master party does not know the instance $(x_1, \ldots, x_n)$ until *after* all parties have published their values; in particular, he does not know the instance when setting up the witness PRF. This is crucial to obtaining a non-interactive scheme. Witness encryption, on the other hand, requires knowing the instance when generating the ciphertext, and therefore appears insufficient for non-interactive key exchange.

Efficiency Comparison. Let $p(\lambda)$ be the circuit size for computing G. It is straightforward to implement a relation for L with circuits of size $8n\lambda + O(\lambda) + p(\lambda)$ (the bottleneck is the muxing operation to select one of the inputs to check). Using fast PRGs, we can take $p(\lambda) = O(\lambda)$. Thus, our witness PRF uses multilinear maps with linearity $8n\lambda + O(\lambda)$. While this is somewhat worse than the multilinearity $n - 1$ required for the direct protocol with trusted setup, it is many orders of magnitude better than the $(2n\lambda)^c$ multilinearity required for the obfuscation-based construction, and only about two orders of magnitude away from what is currently achievable [ACLL14]. We note that, using knowledge variants of obfuscation as in [ABG+13], it is possible to reduce the multilinearity required for the obfuscation construction to $(\lambda \log n)^{c'}$ for a larger constant c'. Using our knowledge variant of witness PRFs, we can similarly reduce the multilinearity of our protocol to $O(\lambda \log n)$. In either case (using knowledge assumptions or not), our witness PRFs are currently (by far) the most efficient multiparty key exchange protocols that do not require a trusted setup.

The reasons for the efficiency gains are two-fold:

- Our witness PRF construction grows polynomially with circuit size, rather than exponentially in the depth as in current obfuscators. Thus we will get immediate improvements for all except the shallowest circuits.
- For the applications discussed in this work, the original constructions required obfuscating a PRF. This translates to using the underlying multilinear map operations to simulate the evaluation of the PRF, which is quite costly. In contrast, our witness PRFs use the multilinear map elements themselves as the PRF outputs, eliminating the need for a separate PRF computation. Thus only the relation checking needs to be carried out with multilinear operations. For cases such as key exchange where the PRF evaluation is the bottleneck, this results in significant additional efficiency gains.

1.4 Directions for Future Work

Our work raises several intriguing open questions:

- We give several applications of witness PRFs that previously required the full power of obfuscation. For what other applications of obfuscation do witness PRFs suffice?
- Witness PRFs do not appear sufficient for many applications of obfuscation, including some that seem well-suited for witness PRFs on the surface. For example, obfuscation plays a similar role of gatekeeper to a PRF in the traitor tracing scheme of Boneh and Zhandry [BZ14]. However, in there scheme, the underlying relation must actually be kept secret for security to hold. In our notion of witness PRFs, the relation is not a secret, and our construction explicitly requires the relation to be public. A natural goal is to devise a stronger notion of witness PRFs that would suffice for these applications (say, by hiding some information about the relation) but yet has efficiency similar to that of witness PRFs and witness encryption.
- While our assumptions are natural, they are *instance dependent*, meaning that the assumption depends on the challenge instance. This means our scheme relies on an exponential number of assumptions, one per instance. An important goal is therefore to construct witness PRFs from simple instance *in*dependent assumptions. We note that the since witness PRFs imply witness encryption, the arguments of Garg et al. [GGSW13] indicate that such a construction would likely involve complexity leveraging.

Indistinguishability obfuscation (iO) can be used to build witness PRFs[8], and iO can in turn be based on simple assumptions following the work of Gentry et al. [GLSW14]. However, such an approach defeats the efficiency gains of building witness PRFs directly. A natural starting point to look for a construction would be the witness encryption scheme of Gentry et al. [GLW14], which is also based on instance independent assumptions.

– How do Witness PRFs relate to other advanced cryptographic primitives? For instance, are witness PRFs indeed weaker than obfuscation, and can witness encryption be used generically to build witness PRFs? In a subsequent work, Komargodski and Zhandry [KZ15] make progress in this direction by showing that witness PRFs are equivalent to a notion of secret sharing called *distributed secret sharing*. An interesting direction for future work would be to find more equivalences, or to give black box separations between witness PRFs and other primitives.

1.5 Other Related Work

Removing Obfuscation. Very recently, a few works have shown how to remove obfuscation from certain applications. Garg et al. [GGHZ14] build the first many-key functional encryption schemes that do not rely on obfuscation, though their construction is obfuscation-inspired. Boneh et al. [BLR+14] build a near-practical order revealing encryption scheme; the only other known construction requires obfuscation and is therefore far from practical.

Smooth Projective Hash Functions and Functional PRFs. Cramer and Shoup [CS02] define the notion of *smooth projective hash functions* (SPHFs), a concept similar to that of witness PRFs. Concurrently and independently of our work, Chen and Zhang [CZ14] define the notion of *publicly evaluable PRFs* (PEPRFs), which are again similar in concept to witness PRFs. The main differences between SPHFs and PEPRFs and our witness PRFs are that existing constructions of SPHFs and PEPRFs are only for certain classes of languages, such as certain group-theoretic languages. Witness PRFs on the other hand, can handle arbitrary NP languages, and such flexibility is required for the applications in this work. The trade-off is that witness PRFs are much less efficient and require much stronger assumptions. There are also minor differences in security notions.

Boyle et al. [BGI14] define *functional* PRFs, where the evaluation key corresponds to a function f, and given the evaluation key it is possible to compute $\mathsf{F}(f(x))$, but $\mathsf{F}(y)$ is pseudorandom for y not in the image of f. Functional PRFs

[8] To see this, start with any "puncturable" PRF, and obufscate the program that takes an input and a witness, checks the witness relation, and outputs the PRF evaluated on the input. The resulting obfuscated program is the evaluation key, and the PRF key is the secret key. Correctness is straightforward to verify, and the static security definition described above can be shown easily through the punctured programming technique of Sahai and Waters [SW14].

in their full generality equivalent to witness PRFs. In one direction, we can set $f(\ (x, w)\ ) = x$ if $R(x, w) = 1$ and $f(\ (x, w)\ ) = \perp$ otherwise. In the other, we can set $R(y; x) = 1$ if $f(x) = y$ and $R(y; x) = 0$ otherwise. We note, however, that [BGI14] only construct functional PRFs for very limited functions f related to prefix matching, which are insufficient for our applications. In particular, the functions f considered all correspond to languages that are in P (and so correspond exactly to constrained PRFs), where our construction supports general NP relations, as needed by our applications.

Witness Encryption. Garg et al. [GGSW13] define witness encryption and give the first candidate construction for the NP-Complete ExactCover problem, whose security is based on the *multilinear no-exact-cover problem*. Goldwasser et al. [GKP+13] define a stronger notion, called extractable witness encryption, which stipulates that anyone who can distinguish the encryption of two messages relative to an instance x must actually be able to produce a witness for x. Our extractable notion for witness PRFs can be seen as a generalization of extractable witness encryption. Subsequently, Garg et al. [GGHW14] cast doubt on the plausibility of the most general forms of extractable witness encryption (and thus extractable witness PRFs), though their results do not apply to most potential applications of the primitives.

Hard-Core Bits. The Goldreich-Levin theorem [GL89] shows how to build a single hard-core bit for any one-way function. This result can be extended to logarithmically-many bits, and polynomially-many hard-core bits have been constructed for *specific* one-way functions [CGH01]. Bellare et al. [BST14] give poly-many hard-core bits for *any* one-way function using obfuscation, which is the only construction prior to this work.

Broadcast Encryption. There has been an enormous body of work on broadcast encryption, and we only mention a few specific works. Boneh et al. [BGW05] use bilinear maps to give a broadcast scheme with short ciphertexts and secret keys, though public broadcast keys grew linearly with the number of users. Some subsequent schemes based on bilinear maps were able to achieve adaptive security [GW09], but the public parameters always grew linearly with the number of recipients. Boneh and Zhandry [BZ14] give a broadcast scheme from indistinguishability obfuscation which achieves similarly short ciphertexts and secret keys. Their broadcast scheme has the novel property of being distributed, where every user chooses their own secret key. However, their public keys are obfuscated programs, and are quite large (namely, linear in the number of users), and security is proved in a weaker *static* model. Ananth et al. [ABG+13] show how to shrink the public key (while maintaining secret key and ciphertext size), though they lose the distributed property. Boneh et al. [BWZ14a] give several broadcast schemes whose concrete parameter sizes are much better directly from multilinear maps, and very recently Zhandry [Zha14a] gives a variant that is adaptively secure. However, these schemes are not distributed.

Secret Sharing. The first secret sharing schemes due to Blakely [Bla79] and Shamir [Sha79] are for the *threshold* access structure, where any set of users of size at least some threshold t can recover the secret, and no set of size less than t can learn anything about the secret. In an unpublished work, Yao shows how to perform (computational) secret sharing where the allowable sets are decided by a polynomial-sized monotone circuit. Komargodski, Naor and Yogev [KNY14] use witness encryption to build the first protocol for arbitrary NP access structures, answering a question of Rudich.

2 Preliminaries

2.1 Subset-Sum

Let $\mathbf{A} \in \mathbb{Z}^{m\times n}$ be an integer matrix, and $\mathbf{t} \in \mathbb{Z}^m$ be an integer vector. The *subset-sum* search problem is to find an $\mathbf{w} \in \{0,1\}^n$ such that $\mathbf{t} = \mathbf{A}\cdot\mathbf{w}$. The decision problem is to decide if such an $\mathbf{w}$ exists.

We define several quantities related to a subset-sum instance. Given a matrix $\mathbf{A} \in \mathbb{Z}^{m\times n}$, let $\mathsf{SubSums}(\mathbf{A})$ be the set of all subset-sums of columns of $\mathbf{A}$. That is, $\mathsf{SubSums}(\mathbf{A}) = \{\mathbf{A}\cdot\mathbf{w} : \mathbf{w} \in \{0,1\}^n\}$. Define $\mathsf{Span}(\mathbf{A})$ as the convex hull of $\mathsf{SubSums}(\mathbf{A})$. Equivalently, $\mathsf{Span}(\mathbf{A}) = \{\mathbf{A}\cdot\mathbf{w} : \mathbf{w} \in [0,1]^n\}$. We define the integer range of $\mathbf{A}$, or $\mathsf{IntRange}(\mathbf{A})$, as $\mathsf{Span}(\mathbf{A})\bigcap\mathbb{Z}^m$. We note that given an instance $(\mathbf{A},\mathbf{t})$ of the subset-sum problem, it is efficiently decidable whether $\mathbf{t} \in \mathsf{IntRange}(\mathbf{A})$. Moreover, $\mathbf{t} \notin \mathsf{IntRange}(\mathbf{A})$ implies that $(\mathbf{A},\mathbf{t})$ is unsatisfiable. The only "interesting" instances of the subset sum problem therefore have $\mathbf{t} \in \mathsf{IntRange}(\mathbf{A})$. From this point forward, we only consider $(\mathbf{A},\mathbf{t})$ a valid subset sum instance if $\mathbf{t} \in \mathsf{IntRange}(\mathbf{A})$.

2.2 Multilinear Maps

An asymmetric multilinear map [BS02] is defined by an algorithm Setup which takes as input a security parameter λ, a multilinearity n, and a minimum group order p_{min}[9]. It outputs (the description of) $n+1$ groups $\mathbb{G}_1,\dots,\mathbb{G}_n,\mathbb{G}_T$ of prime order $p \geq \max(2^\lambda, p_{min})$, corresponding generators $g_1,\dots,g_n,g_T$, and a map $e : \mathbb{G}_1\times\cdots\times\mathbb{G}_n \to \mathbb{G}_T$ satisfying

$$e(g_1^{a_1},\dots,g_n^{a_n}) = g_T^{a_1\cdots a_n}$$

Cryptographic multilinear maps are multilinear maps where certain computations not expressly allowed by the map are computationally difficult. For example, it should at a minimum be computationally infeasible to compute $a \in \mathbb{Z}_p$ given g_i^a for a random a. An example of the type of computational assumption we make in this work is that the following problem is hard: given $g_i^{ab^i}$ for $i \in [n]$, distinguish g_T^a from a random element of $\mathbb{G}_T$.

Another requirement we make on multilinear maps is that a random element of $\mathbb{G}_T$ is statistically indistinguishable from a uniform random bit string.

[9] It is easy to adapt multilinear map constructions [GGH13a,CLT13] to allow setting a minimum group order.

Approximate Multilinear Maps. Current candidate multilinear maps [GGH13a, CLT13] are only *approximate* and do not satisfy the ideal model outlined above. In particular, the maps are noisy, resulting in several implications. First, representations of group elements are not unique. Current map candidates provide an extraction procedure that takes a representation of a group element in the the target group $\mathbb{G}_T$ and outputs a canonical representation. This allows multiple users with different representations of the same element to arrive at the same value. The extraction procedure satisfies the requirement that, when applied to a random element of the target group, the result is statistically close to a uniform random bit string.

A more significant limitation is that noise grows with the number of multiplications and pairing operations. If the noise term grows too large, then there will be errors in the sense that the extraction procedure above will fail to output the canonical representation. In our application, the number of multiplications is equal to the multilinearity, which current candidates natively support without needing to adjust the parameter settings[10].

Lastly, and most importantly for our use, current map candidates do not allow regular users to compute g_i^α for any $\alpha \in \mathbb{Z}_p$ of the user's choice. Instead, the user computes a "level-0 encoding" of a random (unknown) $\alpha \in \mathbb{Z}_p$, and then pairs the "level-0 encoding" with g_i, which amounts computing the exponentiation g_i^α. To compute terms like $g_i^{\alpha^k}$ would require repeating this operation k times, resulting in a large blowup in the error. Thus, for large k, computing terms like $g_i^{\alpha^k}$ is infeasible for regular users. However, whomever sets up the map knows secret parameters about the map and *can* compute g_i^α for any $\alpha \in \mathbb{Z}_p$ without blowing up the error. Thus, the user who sets up the map can pick α, compute α^k in $\mathbb{Z}_p$, and then compute $g_i^{\alpha^k}$ using the map secrets. This will be critical for our construction.

3 Witness PRFs

Informally, a witness PRF is a generalization of constrained PRFs [BW13, KPTZ13, BGI14] to arbitrary NP relations. That is, for an NP language L, a user can evaluate the function F at an instance x only if $x \in L$ *and* the user can provide a witness w that $x \in L$. More formally, a witness PRF is the following:

Definition 1. *A witness PRF is a triple of algorithms* $(\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ *such that:*

– Gen *is a randomized algorithm that takes as input a security parameter* λ *and a circuit* $R : \mathcal{X} \times \mathcal{W} \to \{0,1\}$[11]*, and produces a secret function key* fk *and a public evaluation key* ek.

[10] In fact, the parameters can be set more aggressively since our application does not need to support re-randomization. Re-randomizing elements adds significant noise in current encodings, and the native parameter settings support this noise growth.

[11] By accepting relations as circuits, our notion of witness PRFs only handles instances of a fixed size. It is also possible to consider witness PRFs for instances of arbitrary size, in which case R would be a Turing machine.

- F *is a deterministic algorithm that takes as input the function key* fk *and an input* $x \in \mathcal{X}$*, and produces some output* $y \in \mathcal{Y}$ *for some set* $\mathcal{Y}$*.*
- Eval *is a deterministic algorithm that takes as input the evaluation key* ek *and input* $x \in \mathcal{X}$*, and a witness* $w \in \mathcal{W}$*, and produces an output* $y \in \mathcal{Y}$ *or* $\perp$*.*
- *For correctness, we require* $\mathsf{Eval}(\mathsf{ek}, x, w) = \begin{cases} \mathsf{F}(\mathsf{fk}, x) & \text{if } R(x,w)=1 \\ \perp & \text{if } R(x,w)=0 \end{cases}$ *for all* $x \in \mathcal{X}, w \in \mathcal{W}$*.*

We note one significant way in which our notion of witness PRFs is *weaker* than constrained PRFs: our notion only allows a single evaluation key ek for a relation R that must be chosen at setup time. In contrast, constrained PRFs allow arbitrarily-many ek for different circuits, and the circuits can be chosen after setup. This limitation will be inherent to our construction: the function defined by $\mathsf{F}(\mathsf{fk}, \cdot)$ will depend on the relation R. Nonetheless, this definition will be sufficient for our applications. In the full version [Zha14b], we define a multi-relation variant, discuss a possible approach to building such enhanced primitives.

3.1 Security

The simplest and most natural security notion we consider is a direct generalization of the security notion for constrained PRFs, which we call adaptive instance interactive security. Consider the following experiment $\mathtt{EXP}_{\mathcal{A}}^{R}(b, \lambda)$ between an adversary $\mathcal{A}$ and challenger, parameterized by a relation $R : \mathcal{X} \times \mathcal{W} \to \{0,1\}$, a bit b and security parameter λ.

- Run $(\mathsf{fk}, \mathsf{ek}) \xleftarrow{R} \mathsf{Gen}(\lambda, R)$ and give ek to $\mathcal{A}$.
- $\mathcal{A}$ can adaptively make queries on instances $x_i \in \mathcal{X}$, to which the challenger response with $\mathsf{F}(\mathsf{fk}, x_i)$.
- $\mathcal{A}$ can make a single challenge query on an instance $x^* \in \mathcal{X}$. The challenger computes $y_0 \leftarrow \mathsf{F}(\mathsf{fk}, x^*)$ and $y_1 \xleftarrow{R} \mathcal{Y}$, and responds with y_b.
- After making additional F queries, $\mathcal{A}$ produces a bit b'. The challenger checks that $x^* \notin \{x_i\}$, and that there is no witness $w \in \mathcal{W}$ such that $R(x,w) = 1$ (in other words, $x \notin L$)[12]. If either check fails, the challenger outputs a random bit. Otherwise, it outputs b'.

Define W_b as the event the challenger outputs 1 in experiment b. Let

$$\mathtt{WPRF.Adv}_{\mathcal{A}}^{R}(\lambda) = |\Pr[W_0] - \Pr[W_1]|$$

Definition 2. $\mathsf{WPRF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ *is* adaptive instance interactively secure *for a relation* R *if, for all PPT adversaries* $\mathcal{A}$*, there is a negligible function* negl *such that.*

[12] This check in general cannot be implemented in polynomial time, meaning our challenger is not efficient.

We can also define a weaker notion of *static instance* security where $\mathcal{A}$ commits to x^* before seeing ek or making any F queries. Independently, we can also define *non-interactive* security where the adversary is not allowed any F queries. In the full version [Zha14b], we also consider more fine-grained security notions, similar to the obfuscation-based notions of [BST14]. In the full version, we also consider *extractability* notions of witness PRFs, where pseudorandomness holds even for x^* in the language, as long as the adversary does not "know" a witness for x.

4 An Abstraction: Subset-Sum Encoding

Now that we have seen many applications of witness PRFs, we begin our construction. In this section, we give an abstraction of functionality we need from multilinear maps. Our abstraction is called a *subset-sum* encoding. Roughly, a subset sum encoding is a way to encode vectors $\mathbf{t}$ such that (1) the encoding of $\mathbf{t} = \mathbf{A} \cdot \mathbf{w}$ for $\mathbf{w} \in \{0,1\}^n$ is efficiently computable given $\mathbf{w}$ and (2) the encoding of $\mathbf{t} \notin \mathsf{SubSums}(\mathbf{A})$ is indistinguishable from a random string. More formally, a subset-sum encoding is the following:

Definition 3. *A* subset-sum encoding *is a triple of efficient algorithms* (Gen, $\mathsf{Encode}, \mathsf{Eval}$) *where:*

- Gen *takes as input a security parameter* λ *and an integer matrix* $\mathbf{A} \in \mathbb{Z}^{m\times n}$*, and outputs an encoding key* sk *and an evaluation key* ek.
- Encode *takes as input the secret key* sk *and a vector* $\mathbf{t} \in \mathbb{Z}^m$*, and produces an encoding* $\hat{\mathbf{t}} \in \mathcal{Y}$*.* Encode *is deterministic.*
- Eval *takes as input the encoding key* ek *and a bit vector* $\mathbf{w} \in \{0,1\}^n$*, and outputs a value* $\hat{\mathbf{t}}$ *satisfying* $\hat{\mathbf{t}} = \mathsf{Encode}(\mathsf{sk}, \mathbf{t})$ *where* $\mathbf{t} = \mathbf{A} \cdot \mathbf{w}$.

Security Notions. The security notions we define for subset-sum encodings are very similar to those for witness PRFs. Consider the following experiment $\mathtt{EXP}^{\mathbf{A}}_{\mathcal{A}}(b, \lambda)$ between an adversar $\mathcal{A}$ and challenger, parameterized by a matrix $\mathbf{A} \in \mathbb{Z}^{m\times n}$, a bit b, and a security parameters λ:

- Run $(\mathsf{sk}, \mathsf{ek}) \overset{R}{\leftarrow} \mathsf{Gen}(\lambda, \mathbf{A})$, and give ek to $\mathcal{A}$
- $\mathcal{A}$ can adaptively make queries on targets $\mathbf{t}_i \in \{0,1\}^m$, to which the challenger responds with $\hat{\mathbf{t}_i} \leftarrow \mathsf{Encode}(\mathsf{sk}, \mathbf{t}_i) \in \mathcal{Y}$.
- $\mathcal{A}$ can make a single challenge query on a target $\mathbf{t}^*$. The challenger computes $y_0 = \hat{\mathbf{t}^*} \leftarrow \mathsf{Encode}(\mathsf{sk}, \mathbf{t}^*)$ and $y_1 \overset{R}{\leftarrow} \mathcal{Y}$, and responds with y_b.
- After making additional Encode queries, $\mathcal{A}$ produces a bit b'. The challenger checks that $\mathbf{t}^* \notin \{\mathbf{t}_i\}$ and $\mathbf{t}^* \notin \mathsf{SubSums}(\mathbf{A})$. If either check fails, the challenger outputs a random bit. Otherwise, it outputs b'.

Define W_b as the event the challenger outputs 1 in experiment b. Let

$$\mathtt{SS.Adv}^{\mathbf{A}}_{\mathcal{A}}(\lambda) = |\Pr[W_0] - \Pr[W_1]|$$

Definition 4. $(\mathsf{Gen}, \mathsf{Encode}, \mathsf{Eval})$ *is* adaptive target interactively secure *for a matrix* $\mathbf{A}$ *if, for all adversaries* $\mathcal{A}$*, there is a negligible function* negl *such that* $\mathtt{SS.Adv}^{\mathbf{A}}_{\mathcal{A}}(\lambda) < \mathsf{negl}(\lambda)$*.*

We can also define a weaker notion of *static target* security where $\mathcal{A}$ commits to $\mathbf{t}^*$ before seeing ek or making any Encode queries. Independently, we can also define *non-interactive* security where the adversary is not allowed to make any Encode queries.

4.1 A Simple Instantiation from Multilinear Maps

We now construct subset-sum encodings from asymmetric multilinear maps.

Construction 1. *Let* Setup *be the generation algorithm for an asymmetric multilinear map. We build the following subset-sum encoding:*

- $\mathsf{Gen}(\lambda, \mathbf{A})$*: on input a matrix* $\mathbf{A} \in \mathbb{Z}^{m\times n}$*, let* $B = \|\mathbf{A}\|_\infty$*, and* $p_{min} = 2nB+1$*. Run* $\mathsf{params} \xleftarrow{R} \mathsf{Setup}(\lambda, n, p_{min})$ *to get the description of a multilinear map* $e : \mathbb{G}_1 \times \cdots \times \mathbb{G}_n \to \mathbb{G}_T$ *on groups of prime order* p*, together with generators* $g_1, \ldots, g_m, g_T$*. Choose random* $\boldsymbol{\alpha} \in (\mathbb{Z}_p^*)^m$*. Denote by* $\boldsymbol{\alpha}^{\mathbf{v}}$ *the product* $\prod_{i\in[m]} \alpha_i^{v_i}$ *(since each component of* $\boldsymbol{\alpha}$ *is non-zero, this operation is well-defined for all integer vectors* $\mathbf{v}_i$*). Let* $V_i = g_i^{\boldsymbol{\alpha}^{\mathbf{v}_i}}$ *where* $\mathbf{v}_i$ *are the columns of* $\mathbf{A}$*. Publish* $\mathsf{ek} = (\mathsf{params}, \{V_i\}_{i\in[n]})$ *as the public parameters and* $\mathsf{sk} = \boldsymbol{\alpha}$
- $\mathsf{Encode}(\mathsf{sk}, \mathbf{t}) = g_T^{\boldsymbol{\alpha}^{\mathbf{t}}}$*, where* $\mathbf{t} \in \mathsf{IntRange}(\mathbf{A})$*.*
- $\mathsf{Eval}(\mathsf{ek}, \mathbf{w})$*: define* $V_{i,1} = V_i$ *and* $V_{i,0} = g_i$*. Then output*

$$e(V_{1,w_1}, V_{2,w_2}, \ldots, V_{n,w_n})$$

For correctness, observe that $V_{i,w_i} = g_i^{\boldsymbol{\alpha}^{\mathbf{v}_i w_i}}$, and therefore

$$\begin{aligned} e(V_{1,w_1}, V_{2,w_2}, \ldots, V_{n,w_n}) &= e(g_1^{\boldsymbol{\alpha}^{\mathbf{v}_1 w_1}}, \ldots, g_n^{\boldsymbol{\alpha}^{\mathbf{v}_n w_n}}) = g_T^{\boldsymbol{\alpha}^{\sum_{i\in[n]} \mathbf{v}_i w_i}} = g_T^{\boldsymbol{\alpha}^{\mathbf{A}\cdot\mathbf{w}}} \\ &= \mathsf{Encode}(\mathsf{sk}, \mathbf{A}\cdot\mathbf{w}) \end{aligned}$$

Security. We assume the security of our subset-sum encodings, which translates to a new security assumption on multilinear maps, which we call the *(adaptive target interactive) multilinear subset-sum Diffie Hellman assumption.* For completeness, we formally define the assumption as follows. Let $\mathtt{EXP}^{\mathbf{A}}_{\mathcal{A}}(b, \lambda)$ be the following experiment between an adversary $\mathcal{A}$ and challenger, parameterized by a matrix $\mathbf{A} \in \mathbb{Z}^{m\times n}$, a bit b, and a security parameter λ:

- Let $B = \|\mathbf{A}\|_\infty$, and $p_{min} = 2nB + 1$. Run $\mathsf{params} \xleftarrow{R} \mathsf{Setup}(\lambda, n, p_{min})$.
- Choose a random $\boldsymbol{\alpha} \in \mathbb{Z}_p^m$, and let $V_i = g_i^{\boldsymbol{\alpha}^{\mathbf{v}_i}}$ where $\mathbf{v}_i$ are the columns of $\mathbf{A}$. Give $(\mathsf{params}, \{V_i\}_{i\in[n]})$ to $\mathcal{A}$.
- $\mathcal{A}$ can make oracle queries on targets $\mathbf{t}_i \in \mathsf{IntRange}(\mathbf{A})$, to which the challenger responds with $g_T^{\boldsymbol{\alpha}^{\mathbf{t}_i}}$.

- $\mathcal{A}$ can make a single challenge query on a target $\mathbf{t}^* \in \mathsf{IntRange}(\mathbf{A})$. The challenger computes $y_0 = g_T^{\boldsymbol{\alpha}^{\mathbf{t}^*}}$ and $y_1 = g_T^r$ for a random $r \overset{R}{\leftarrow} \mathbb{Z}_p$, and responds with y_b.
- After making additional Encode queries, $\mathcal{A}$ produces a bit b'. The challenger checks that $\mathbf{t}^* \notin \{t_i\}$ and $t^* \notin \mathsf{SubSums}(\mathbf{A})$. If either check fails, the challenger outputs a random bit. Otherwise, it outputs b'.

Define W_b as the event that the challenger outputs 1 in experiment b. Let $\mathtt{SSDH.Adv}_{\mathcal{A}}^{\mathbf{A}}(\lambda) = |\Pr[W_0] - \Pr[W_1]|$.

Definition 5. *The adaptive target interactive multilinear subset-sum Diffie Hellman (SSDH) assumption holds relative to* Setup *if, for all adversaries* $\mathcal{A}$*, there is a negligible function* negl *such that* $\mathtt{SSDH.Adv}_{\mathcal{A}}^{\mathbf{A}}(\lambda) < \mathsf{negl}(\lambda)$.

Security of our subset-sum encodings immediately follows from the assumption:

Fact 2. *If the adaptive target interactive multilinear SSDH assumptions holds for* Setup*, the Construction 1 is an adaptive target interactively secure subset-sum encoding.*

Flattening the Encodings. We can convert any subset-sum encoding for $m = 1$ into a subsetsum encoding for any m. Let $\mathbf{A} \in \mathbb{Z}^{m \times n}$ and define $B = \|\mathbf{A}\|_\infty$. Then, for any $\mathbf{w} \in \{0,1\}^n$, $\|\mathbf{A} \cdot \mathbf{w}\|_\infty \leq nB$. Therefore, we can let $\mathbf{A}' = (1, nB + 1, (nB+1)^2, \ldots, (nB+1)^{m-1}) \cdot \mathbf{A}$ be a single row, and run $\mathsf{Gen}(\lambda, \mathbf{A}')$ to get $(\mathsf{sk}, \mathsf{ek})$. To encode an element $\mathbf{t}$, compute $\mathbf{t}' = (1, nB, (nB)^2, \ldots, (nB)^{m-1}) \cdot \mathbf{t}$, and encode $\mathbf{t}'$. Finally, to evaluate on vector $\mathbf{w}$, simply run $\mathsf{Eval}(\mathsf{ek}, \mathbf{w})$.

Security translates since left-multiplying by $(1, nB, (nB)^2, \ldots, (nB)^{m-1})$ does not introduce any collisions. Therefore, we can always rely on subset-sum encodings, and thus the subset-sum Diffie-Hellman assumption, for $m = 1$. However, we recommend *not* using this conversion for two reasons:

- To prevent the exponent from "wrapping" mod $p-1$, $p-1$ needs to be larger than the maximum L_1-norm of the rows of $\mathbf{A}$. In this conversion, we are multiplying rows by exponential factors, meaning p needs to correspondingly be set much larger.
- In the full version [Zha14b], we prove the security of our encodings in the generic multilinear map model. Generic security is only guaranteed if $\|\mathbf{A}\|_\infty / p$ is negligible. This means for security, p will have to be substantially larger after applying the conversion.

4.2 Witness PRFs from Subset-Sum Encodings

We note that subset-sum encodings immediately give us witness PRFs for restricted classes. In particular, for a matrix $\mathbf{A}$, a subset-sum encoding is a witness PRF for the language $\mathsf{SubSums}(\mathbf{A})$. The various security notions for subset-sum encodings correspond exactly to the security notions for witness PRFs. In the full version [Zha14b], we show how to extend this to witness PRFs for any NP language, obtaining the following theorem:

Theorem 3. *If adaptive/static target interactively/non-interactively secure subset-sum encodings exist, then adaptive/static instance interactively/non-interactively secure witness PRFs exist.*

Roughly, we prove this Theorem 3 by providing a reduction from an instance x of any NP language L to subset-sum instance $(\mathbf{A}, \mathbf{t})$, where the matrix $\mathbf{A}$ is determined entirely by the language L, and is independent of x (except for its length). Thus, $\mathsf{SubSums}(\mathbf{A})$ corresponds exactly with L.

Our witness PRF for a language L is then a subset-sum encoding for the corresponding matrix $\mathbf{A}$. The value of the PRF on instance x is the encoding of the corresponding target $\mathbf{t}$. Given a witness w for x, the reduction gives a corresponding subset S of columns of $\mathbf{A}$ that sum to $\mathbf{t}$. This allows anyone with a witness to evaluate the PRF at x.

5 Applications

In this section, we show that for several applications of obfuscation, the obfuscator can be replaced with witness PRFs.

5.1 CCA-secure Public Key Encryption

We demonstrate that witness PRFs give a simple construction of CCA-secure public key encryption that is similar to the obfuscation-based construction of Sahai and Waters [SW14]. Given the similarities of witness PRFs to smooth projective hash functions (SPHFs) [CS02], and that the original motivation for SPHFs was CCA-secure public key encryption, this result is not surprising. Instead, we present the construction as a warm-up for the more interesting applications that follow.

Construction 4. *Let* $\mathsf{WPRF} = (\mathsf{WPRF.Gen}, \mathsf{F}, \mathsf{Eval})$ *be a witness PRF, and let* $\mathsf{G} : \mathcal{S} \to \mathcal{Z}$ *be a pseudorandom generator with* $|\mathcal{S}|/|\mathcal{Z}| < \mathsf{negl}$. *Build the following key encapsulation mechanism* $(\mathsf{Enc.Gen}, \mathsf{Enc}, \mathsf{Dec})$*:*

- $\mathsf{Enc.Gen}(\lambda)$*: Let* $R(z, s) = 1$ *if and only if* $\mathsf{G}(s) = z$. *In other words,* R *defines the language* L *of strings* $z \in \mathcal{Z}$ *that are images of* G, *and witnesses are the corresponding pre-images. Run* $(\mathsf{fk}, \mathsf{ek}) \overset{R}{\leftarrow} \mathsf{WPRF.Gen}(\lambda, R)$. *Set* fk *to be the secret key and* ek *to be the public key.*
- $\mathsf{Enc}(\mathsf{ek})$*: sample* $s \overset{R}{\leftarrow} \mathcal{S}$ *and set* $z \leftarrow \mathsf{G}(s)$. *Output* z *as the header and* $k \leftarrow \mathsf{Eval}(\mathsf{ek}, z, s) \in \mathcal{Y}$ *as the message encryption key.*
- $\mathsf{Dec}(\mathsf{fk}, z)$*: run* $k \leftarrow \mathsf{F}(\mathsf{fk}, z)$.

Correctness is immediate. For security, we have the following:

Theorem 5. *If* WPRF *is interactively secure, then Construction 4 is a CCA secure key encapsulation mechanism. If* WPRF *is static instance non-interactively secure, then Construction 4 is CPA secure.*

Proof. We prove the CCA case, the CPA case being almost identical. Let $\mathcal{B}$ be a CCA adversary with non-negligible advantage ϵ. Define **Game 0** as the standard CCA game, and define **Game 1** as the modification where the challenge header z^* is chosen uniformly at random in $\mathcal{Z}$. The security of G implies that $\mathcal{B}$ still has advantage negligibly-close to ϵ. Let **Game 2** be the game where z^* is chosen at random, but the game outputs a random bit and aborts if z^* is in the image space of G. Since $\mathcal{Z}$ is much larger than $\mathcal{S}$, the abort condition occurs with negligible probability. Thus $\mathcal{B}$ still has advantage negligibly close to ϵ in **Game 2**. Now we construct an adversary $\mathcal{A}$ for WPRF. $\mathcal{A}$ chooses a random z^*, and makes a challenge query on z^*, obtaining k. Then it simulates $\mathcal{B}$, answering decryption queries using its F oracle. When $\mathcal{B}$ makes a challenge query, and $\mathcal{A}$ responds with z^* as the header and k as the encapsulated key. When $\mathcal{B}$ outputs a bit b', $\mathcal{A}$ outputs the same bit. $\mathcal{A}$ has advantage equal to that of $\mathcal{B}$ in **Game 2**, which is non-negligible, thus contradicting the security of WPRF.

5.2 Non-interactive Multiparty Key Exchange

A multiparty key exchange protocol allows a group of g users to simultaneously post a message to a public bulletin board, retaining some user-dependent secret. After reading off the contents of the bulletin board, all the users establish the same shared secret key. Meanwhile, and adversary who sees the entire contents of the bulletin board should not be able to learn the group key. More precisely, a multiparty key exchange protocol consists of:

- $\mathsf{Publish}(\lambda, g)$ takes as input the security parameter and the group order, and outputs a user secret s and public value pv. pv is posted to the bulletin board.
- $\mathsf{KeyGen}(\{\mathsf{pv}_j\}_{j\in[g]}, s_i, i)$ takes as input g public values, plus the corresponding user secret s_i for the ith value. It outputs a group key $k \in \mathcal{Y}$.

For correctness, we require that all users generate the same key:

$$\mathsf{KeyGen}(\{\mathsf{pv}_j\}_{j\in[g]}, s_i, i) = \mathsf{KeyGen}(\{\mathsf{pv}_j\}_{j\in[g]}, s_{i'}, i')$$

for all $(s_j, \mathsf{pv}_j) \xleftarrow{R} \mathsf{Publish}(\lambda, g)$ and $i, i' \in [g]$. For security, we have the following:

Definition 6. *A non-interactive multiparty key exchange protocol is statically secure if the following distributions are indistinguishable:*

$$\{\mathsf{pv}_j\}_{j\in[g]}, k \textit{ where } (s_j, \mathsf{pv}_j) \xleftarrow{R} \mathsf{Publish}(\lambda, g) \forall j \in [g], k \xleftarrow{R} \mathcal{Y} \textit{ and}$$

$$\{\mathsf{pv}_j\}_{j\in[g]}, k \textit{ where } (s_j, \mathsf{pv}_j) \xleftarrow{R} \mathsf{Publish}(\lambda, g) \forall j \in [g], k \leftarrow \mathsf{KeyGen}(\{\mathsf{pv}_j\}_{j\in[g]}, s_1, 1)$$

Notice that our syntax does not allow a trusted setup, as constructions based on multilinear maps [BS02, GGH13a, CLT13] require. Boneh and Zhandry [BZ14] give the first multiparty key exchange protocol without trusted setup, based on obfuscation. We now give a very similar protocol using witness PRFs.

Construction 6. *Let* $\mathsf{G} : \mathcal{S} \to \mathcal{Z}$ *be a pseudorandom generator with* $|\mathcal{S}|/|\mathcal{Z}| <$ negl. *Let* $\mathsf{WPRF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ *be a witness PRF. Let* $R_g : \mathcal{Z}^g \times (\mathcal{S} \times [g]) \to \{0,1\}$ *be a relation that outputs 1 on input* $((z_1, \ldots, z_g), (s, i))$ *if and only if* $z_i = \mathsf{G}(s)$. *We build the following key exchange protocol:*

- $\mathsf{Publish}(\lambda, g)$*: compute* $(\mathsf{fk}, \mathsf{ek}) \overset{R}{\leftarrow} \mathsf{Gen}(\lambda, R_g)$. *Also pick a random seed* $s \overset{R}{\leftarrow} \mathcal{S}$ *and compute* $z \leftarrow \mathsf{G}(s)$. *Keep* s *as the secret and publish* (z, ek).
- $\mathsf{KeyGen}(\{(z_i, \mathsf{ek}_i)\}_{i \in [g]}, s)$. *Each user sorts the pairs* (z_i, ek_i) *by* z_i, *and determines their index* i *in the ordering. Let* $\mathsf{ek} = \mathsf{ek}_1$, *and compute* $k = \mathsf{Eval}(\mathsf{ek}, (z_1, \ldots, z_g), (s, i))$

Correctness is immediate. For security, we have the following:

Theorem 7. *If* WPRF *is static witness non-interactively secure, the Construction 6 is statically secure.*

Proof. Let $\mathcal{B}$ be an adversary for the key exchange protocol with non-negligible advantage. Then $\mathcal{B}$ sees $\{(z_i, \mathsf{ek}_i)\}_{i \in [g]}$ where $z_i \leftarrow \mathsf{G}(s_i)$ for a random $s_i \overset{R}{\leftarrow} \mathcal{S}$, as well as a key $k \in \mathcal{Y}$, and outputs a guess b' for whether $k = \mathsf{F}(\mathsf{ek}_1, \{(z_i)\}_{i \in [g]}$ or $k \overset{R}{\leftarrow} \mathcal{Y}$. Call this **Game 0**. Define **Game 1** as the modification where $z_i \overset{R}{\leftarrow} \mathcal{Z}$. The security of G implies that **Game 0** and **Game 1** are indistinguishable. Next define **Game 2** as identical to **Game 1**, except that the challenger outputs a random bit and aborts if any of the z_i are in the range of G. Since $|\mathcal{S}|/|\mathcal{Z}| < \mathsf{negl}$, this abort condition occurs with negligible probability, meaning $\mathcal{B}$ still has non-negligible advantage in **Game 2**. We construct an adversary $\mathcal{A}$ for WPRF as follows: $\mathcal{A}$ choses random $z_i \in \mathcal{Z}$ for $i \in [g]$, sorts the z_i, and makes a challenge query on $(z_1, \ldots, z_g)$, obtaining key k. Then after receiving ek, it sets $\mathsf{ek}_1 = \mathsf{ek}$. For $i > 1$, $\mathcal{A}$ runs $(\mathsf{fk}_i, \mathsf{ek}_i) \overset{R}{\leftarrow} \mathsf{Gen}(\lambda, R_g)$. It then gives $\mathcal{A}$ $\{(z_i, \mathsf{ek}_i)\}_{i \in [g]}, k$. Note that for key generation, $\mathsf{ek}_1 = \mathsf{ek}$ is chosen. Also, $(z_1, \ldots, z_g)$ is chosen at random in $\mathcal{Z}^g$, and $\mathcal{A}$'s challenger aborts if any of the z_g are in the range of G (that is, if $(z_1, \ldots, z_g)$ has a witness under R_g). Therefore, the view of $\mathcal{B}$ as a subroutine of $\mathcal{A}$ and the view of $\mathcal{B}$ in **Game 2** are identical. Therefore, the advantage of $\mathcal{A}$ is also non-negligible, a contradiction.

Adaptive Security. In semi-static or active security (defined by Boneh and Zhandry [BZ14]), the same published values pv_j are used in many key exchanges, some involving the adversary. Obtaining semi-static or adaptive security from even the strongest forms of witness PRFs is not immediate. The issue, as noted by Boneh and Zhandry in the case of obfuscation, is that, even in the semi-static setting, the adversary may see the output of Eval on honest secrets, but using a malicious key ek. It may be possible for a malformed key to leak the honest secrets, thereby allowing the scheme to be broken. In more detail, consider an adversary $\mathcal{A}$ playing the role of user i, and suppose the maximum number of users in any group is 2. $\mathcal{A}$ generates and publishes params_i in a potentially malicious way (and also generates and publishes some z_i). Meanwhile, an honest user j publishes an honest ek_j and $z_j = G(s_j)$. Now, if $z_i < z_j$, user j computes the shared key for the group $\{i, j\}$ as $\mathsf{Eval}(\mathsf{ek}_i, (z_i, z_j), s_j, 2)$. While an honest ek_i would cause Eval to be independent of the witness, it may be possible for a dishonest ek_i to cause Eval to leak information about the witness.

Boneh and Zhandry circumvent this issue by using a special type of signature scheme, which they call a *puncturable* signature scheme, and only inputting signatures into Eval. Even if the entire signature leaks, it will not help the adversary produce the necessary signature to break the scheme. Such signature schemes can be built from witness indistinguishable proofs. It is straightforward to adapt Boneh and Zhandry's construction to use witness PRFs instead of obfuscation. We omit the details.

References

[AB15] Applebaum, B., Brakerski, Z.: Obfuscating circuits via composite-order graded encoding. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 528–556. Springer, Heidelberg (2015)

[ABG+13] Ananth, P., Boneh, D., Garg, S., Sahai, A., Zhandry, M.: Differing-inputs obfuscation and applications. Cryptology ePrint Archive, Report 2013/689 (2013). http://eprint.iacr.org/2013/689

[ACLL14] Albrecht, M.R., Cocis, C., Laguillaumie, F., Langlois, A.: Implementing candidate graded encoding schemes from ideal lattices. Cryptology ePrint Archive, Report 2014/928 (2014). http://eprint.iacr.org/2014/928

[AGIS14] Ananth, P.V., Gupta, D., Ishai, Y., Sahai, A.: Optimizing obfuscation: avoiding Barrington's theorem. In Ahn, G.-J., Yung, M., Li, M. (eds.) ACM CCS 14: 21st Conference on Computer and Communications Security, Scottsdale, AZ, USA, 3–7 November 2014, pp. 646–658. ACM Press (2014)

[App13] Applebaum, B.: Bootstrapping obfuscators via fast pseudorandom functions. Cryptology ePrint Archive, Report 2013/699 (2013). http://eprint.iacr.org/2013/699

[BCP14] Boyle, E., Chung, K.-M., Pass, R.: On extractability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 52–73. Springer, Heidelberg (2014)

[BGI+01] Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (Im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001)

[BGI14] Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (2014)

[BGK+14] Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 221–238. Springer, Heidelberg (2014)

[BGW05] Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)

[Bla79] Blakley, G.R.: Safeguarding cryptographic keys. In: Proceedings of AFIPS 1979 National Computer Conference, vol. 48, pp. 313–317 (1979)

[BLR+14] Boneh, D., Lewi, K., Raykova, M., Sahai, A., Zhandry, M., Zimmerman, J.: Semantically secure order-revealing encryption: multi-input functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/834 (2014). http://eprint.iacr.org/2014/834

[BR14] Brakerski, Z., Rothblum, G.N.: Virtual black-box obfuscation for all circuits via generic graded encoding. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 1–25. Springer, Heidelberg (2014)

[BS02] Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. Cryptology ePrint Archive, Report 2002/080 (2002). http://eprint.iacr.org/2002/080

[BST14] Bellare, M., Stepanovs, I., Tessaro, S.: Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 102–121. Springer, Heidelberg (2014)

[BW13] Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 280–300. Springer, Heidelberg (2013)

[BWZ14a] Boneh, D., Waters, B., Zhandry, M.: Low overhead broadcast encryption from multilinear maps. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 206–223. Springer, Heidelberg (2014)

[BWZ14b] Boneh, D., Wu, D.J., Zimmerman, J.: Immunizing multilinear maps against zeroizing attacks. Cryptology ePrint Archive, Report 2014/930 (2014). http://eprint.iacr.org/2014/930

[BZ14] Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 480–499. Springer, Heidelberg (2014)

[CGH01] Catalano, D., Gennaro, R., Howgrave-Graham, N.: The bit security of Paillier's encryption scheme and its applications. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 229–243. Springer, Heidelberg (2001)

[CHL+14] Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the multilinear map over the integers. Cryptology ePrint Archive, Report 2014/906 (2014). http://eprint.iacr.org/2014/906

[CLT13] Coron, J.-S., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 476–493. Springer, Heidelberg (2013)

[CLT14] Coron, J.-S., Lepoint, T., Tibouchi, M.: Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975 (2014). http://eprint.iacr.org/2014/975

[CS02] Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)

[CZ14] Chen, Y., Zhang, Z.: Publicly evaluable pseudorandom functions and their applications. In: Abdalla, M., De Prisco, R. (eds.) SCN 2014. LNCS, vol. 8642, pp. 115–134. Springer, Heidelberg (2014)

[GGH13a] Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013)

[GGH+13b] Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th Annual Symposium on Foundations of Computer Science, Berkeley, CA, USA, 26–29 October 2013, pp. 40–49. IEEE Computer Society Press (2013)

[GGHR14] Garg, S., Gentry, C., Halevi, S., Raykova, M.: Two-round secure MPC from indistinguishability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 74–94. Springer, Heidelberg (2014)

[GGHW14] Garg, S., Gentry, C., Halevi, S., Wichs, D.: On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 518–535. Springer, Heidelberg (2014)

[GGHZ14] Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Fully secure functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/666 (2014). http://eprint.iacr.org/2014/666

[GGSW13] Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th Annual ACM Symposium on Theory of Computing, Palo Alto, CA, USA, 1–4 June 2013, pp. 467–476. ACM Press (2013)

[GHMS14] Gentry, C., Halevi, S., Maji, H.K., Sahai, A.: Zeroizing without zeroes: cryptanalyzing multilinear maps without encodings of zero. Cryptology ePrint Archive, Report 2014/929 (2014). http://eprint.iacr.org/2014/929

[GKP+13] Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: How to run turing machines on encrypted data. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 536–553. Springer, Heidelberg (2013)

[GL89] Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: 21st Annual ACM Symposium on Theory of Computing, Seattle, Washington, USA, 15–17 May 1989, pp. 25–32. ACM Press (1989)

[GLSW14] Gentry, C., Lewko, A., Sahai, A., Waters, B.: Indistinguishability obfuscation from the multilinear subgroup elimination assumption. Cryptology ePrint Archive, Report 2014/309 (2014). http://eprint.iacr.org/2014/309

[GLW14] Gentry, C., Lewko, A., Waters, B.: Witness encryption from instance independent assumptions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 426–443. Springer, Heidelberg (2014)

[GW09] Gentry, C., Waters, B.: Adaptive security in broadcast encryption systems (with short ciphertexts). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (2009)

[HJK+14] Hofheinz, D., Jager, T., Khurana, D., Sahai, A., Waters, B., Zhandry, M.: How to generate and use universal samplers. Cryptology ePrint Archive, Report 2014/507 (2014). http://eprint.iacr.org/2014/507

[Jou04] Joux, A.: A one round protocol for tripartite Diffie-Hellman. J. Cryptol. **17**(4), 263–276 (2004)

[KNY14] Komargodski, I., Naor, M., Yogev, E.: Secret-sharing for NP. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 254–273. Springer, Heidelberg (2014)

[KPTZ13] Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Sadeghi, A.-Z., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013: 20th Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013 pp. 669–684. ACM Press (2013)

[KZ15] Komargodski, I., Zhandry, M.: Modern cryptography through the lens of secret sharing. Cryptology ePrint Archive, Report 2015/735 (2015). http://eprint.iacr.org/2015/735

[PST14] Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation from semantically-secure multilinear encodings. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 500–517. Springer, Heidelberg (2014)

[Rao14] Rao, V.: Adaptive multiparty non-interactive key exchange without setup in the standard model. Cryptology ePrint Archive, Report 2014/910 (2014). http://eprint.iacr.org/2014/910

[Sha79] Shamir, A.: How to share a secret. Commun. Assoc. Comput. Mach. **22**(11), 612–613 (1979)

[SW14] Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th Annual ACM Symposium on Theory of Computing, 31 May– 3 June 2014, pp. 475–484. ACM Press, New York (2014)

[SZ14] Sahai, A., Zhandry, M.: Obfuscating low-rank matrix branching programs. Cryptology ePrint Archive, Report 2014/773 (2014). http://eprint.iacr.org/2014/773

[Zha14a] Zhandry, M.: Adaptively secure broadcast encryption with small system parameters. Cryptology ePrint Archive, Report 2014/757 (2014). http://eprint.iacr.org/2014/757

[Zha14b] Zhandry, M.: How to avoid obfuscation using witness PRFs. Cryptology ePrint Archive, Report 2014/301 (2014). http://eprint.iacr.org/2014/301

[Zim15] Zimmerman, J.: How to obfuscate programs directly. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 439–467. Springer, Heidelberg (2015)