Obfuscation of Evasive Algebraic Set Membership

Steven D. Galbraith and Trey Li
s.galbraith@auckland.ac.nz and treyquantum@gmail.com

Abstract. Canetti, Rothblum, and Varia showed how to obfuscate membership testing in a hyperplane over a finite field of exponentially large prime order, assuming the membership predicate is evasive and the under a modified DDH assumption. Barak, Bitansky, Canetti, Kalai, Paneth, and Sahai extended their work from hyperplanes to hypersurfaces (of bounded degree), assuming multi-linear maps.

In this paper we give much more general obfuscation tools that allow to obfuscate evasive membership testing in arbitrary algebraic sets (including projective sets) over finite fields of arbitrary (prime power) order. We give two schemes and prove input-hiding security based on relatively standard assumptions. The first scheme is based on the preimage resistance property of cryptographic hash functions; and the second scheme is based on the hardness assumptions required for small superset obfuscation. We also introduce a new security notion called span-hiding, and prove that the second scheme achieves span-hiding assuming small superset obfuscation.

One special case of algebraic sets over finite fields is boolean polynomials, which means our methods can be applied to obfuscate any evasive function defined by a polynomial-size collection of boolean polynomials. As a corollary, we obtain an input-hiding obfuscator for evasive functions defined by circuits in NC^0.

1 Introduction

Let q be a prime and let $f_1, \ldots, f_m \in \mathbb{F}_q[x_1, \ldots, x_n]$ be polynomials that can be efficiently evaluated. More precisely, we assume that there are at most ℓ monomials in total among the polynomials, and that the monomials can be evaluated by polynomially many (in $\log(q)$, n, m, ℓ) squarings and multiplications. Consider the set S of all points in $\mathbb{F}_q^n$ that are solutions to all m polynomials. This is an (affine) algebraic set. For large q, the dimension of the (irreducible components of the) algebraic set is typically $n - m$ and the number of solutions is of order q^{n-m}. In the special case where $m = 1$ and the polynomials are linear, the set of solutions is an affine space of codimension 1 and has q^{n-1} points.

There are many different possible sets of polynomials that generate the same solution set. Precisely, there is an ideal corresponding to the algebraic set, and any set of polynomials that generates the ideal also gives rise to the same solution set.

We are interested in obfuscating the predicate that takes such a system of polynomial equations, and a point $P \in \mathbb{F}_q^n$, and returns 1 if and only if $f_i(P) = 0$ for all $1 \leq i \leq m$. In other words, we are obfuscating the membership predicate about whether $P \in S$. The goal of obfuscation is to "hide" the set S, so that it is hard for an attacker to learn properties of S or to find elements of S.

If an attacker knows which monomials appear in the system of polynomials, and if they can efficiently sample points P that lie in the set of solutions, then they can use linear algebra to determine the ideal (see Proposition 1). Hence, the obfuscation problem is only interesting when the predicate is evasive, meaning that a random point satisfies the predicate with negligible probability. In other words, we are forced to focus on the evasive setting.

The probability that a random point $P \in \mathbb{F}_q^n$ satisfies the system is $\#S/q^n$ which is typically approximately $1/q^m$. Hence, to ensure the system is evasive, a necessary condition is

$$q^m \geq 2^\lambda, \tag{1}$$

where λ is the security parameter[1]. A more careful analysis of evasiveness is given in Section 3.

We also remark that when $\#S$ is small (e.g., polynomial in the parameters of the system) and if one can efficiently compute the elements of S, then one can obfuscate membership of S using polynomially many obfuscated point functions. In other words, for each $Q \in S$ we obfuscate the predicate "is $P = Q$?". For this reason, we focus in this paper on the cases where $\#S$ is large. Precisely, we assume

$$q^{n-m} \geq s(\lambda), \tag{2}$$

where $s(\lambda)$ is some function that grows faster than polynomial (i.e. super-polynomial).

Some particular cases of interest are $m = 1, q > 2^{128}, n \geq 2, \ell \geq 3$; $m = 2, q > 2^{64}, \ell > n \geq 3$; $m = 128, q = 2, n > 256, \ell > 300$.

Special cases of this obfuscation problem have been considered in the literature. Canetti, Rothblum, and Varia [10] gave a solution to the special case $m = 1$, linear homogeneous polynomials, and q being prime and large. They need $q > 2^{2\lambda}$ since they rely on the difficulty of the (modified) decision Diffie-Hellman problem in a group of order q. Barak, Bitansky, Canetti, Kalai, Paneth, and Sahai (see Section 3 of [1]) extended the work of [10] from hyperplanes to hypersurfaces. So they still have $m = 1$ and q being prime, but allow more general polynomials than linear. Their solution uses a multi-linear map (graded encoding scheme), which is a very strong assumption.

In this paper we consider the general case of the problem, and provide a solution that is applicable for a much wider range of parameters than previous

[1] Here we use $1/2^\lambda$ to represent "negligible" for the convenience of parameter analysis in Section 5.2. In theory, a negligible function is any function of the form $1/s(\lambda)$ with s a super-polynomial.

solutions. We handle algebraic sets over finite fields $\mathbb{F}_q$ of arbitrary prime power order $q \geq 2$. In particular, we explain how to approach the problem of projective algebraic sets (homogeneous polynomials).

We stress that the problem of obfuscating membership of algebraic systems is fundamental, as a wide range of statements can be expressed as solutions to polynomial systems. Some applications for the case of large prime q are mentioned in [10], however we believe the cases with small q, especially $q = 2$, will be more useful in practice since they include boolean formulas. For example, any circuit in the class NC^0 of circuits with constant depth and fan-in 2 can be represented as a polynomial in $\mathbb{F}_2[x_1, \ldots, x_n]$ of degree n^C for some constant C. Hence our methods can be applied to obfuscate any evasive function defined by a polynomial number of circuits in NC^0. But also there are other useful circuits that can be represented by boolean polynomials with relatively few monomials.

There are several security notions for obfuscation of evasive functions. The most popular one is virtual black-box (VBB) security [2], as well as its various variants [14,1,19]. It is informally defined as: an attacker given the obfuscated program cannot compute any predicate of the un-obfuscated program (circuit), apart from those predicates that can be learned from oracle access to the program. Canetti, Rothblum, and Varia [10] and Barak, Bitansky, Canetti, Kalai, Paneth, and Sahai [1] prove VBB security for their obfuscators. Another notion is input-hiding security [1], and it is often the most relevant security property in many applications, such as password checks (point function obfuscation) and biometric authentication (fuzzy Hamming distance matching obfuscation). Input-hiding security is informally defined as: an attacker given the obfuscated program cannot efficiently compute an input that is accepted by the program. Relations between VBB and input-hiding for evasive functions are discussed in Section 2.2 of [1]. In general, input-hiding and VBB are incomparable, but they are equivalent in some special cases. For evasive functions it is arguably more relevant to focus on input-hiding security, and this is what we do in our paper.

1.1 Our contribution

We give two solutions to the problem of obfuscating membership of algebraic sets. The first solution is based on hash functions, and is appropriate for systems of polynomials with nonzero constant terms (hence this approach cannot be used for projective algebraic sets). In fact, the affine case of our problem is a special case of "compute-and-compare" obfuscation [19,16]. Our solution is much more efficient than applying general techniques such as in [19].

Compared with the general methods, our hashed-based approach for affine algebraic set membership obfuscation is simple and practical. In the case of hyperplane membership, our hashed-based obfuscation works for a much wider range of parameters than [10]. Specifically, [10] requires $q > 2^{2\lambda}$ for λ-bit security, while we only require $q > 2^{\lambda}$.

Our second solution handles both affine and projective cases. It is based on providing dummy polynomial equations and hiding the "real" polynomials among the dummy ones. Beyond the hashed-based solution, this solution further

hides the span of the coefficient matrix, which stores the information of the shape of the geometry body.

We briefly sketch the two solutions here.

Let $M_1, \ldots, M_\ell \in \mathbb{F}_q[x_1, \ldots, x_n]$ be the non-constant monomials. For each $1 \leq i \leq m$ write

$$f_i(x_1, \ldots, x_n) = \sum_{j=1}^{\ell} A_{i,j} M_j \ - \ b_i$$

where $A_{i,j}, b_i \in \mathbb{F}_q$. Then we can represent the system of polynomials as the linear system $AM = b$ over $\mathbb{F}_q$, where A is the $m \times \ell$ matrix of the $A_{i,j}$, M is the length ℓ column vector of the M_j, and b is the length m column vector of the b_i. To check if a point $P = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ is a solution to the system, one can evaluate the vector of monomials $M(x)$ at the point P (this is a polynomial-time computation by definition) and check if $AM(P) = b$.

To obfuscate this we simply publish A and $h = H(b)$, where H is a cryptographic hash function. To compute the membership predicate we compute $H(AM(x))$ and check if this is equal to h. Since b is a vector of length m of elements in $\mathbb{F}_q$, the number of possible b is q^m. By the evasive requirement we have $q^m > 2^\lambda$ and so it is not possible to try all values for b to reverse-engineer the system. It follows that this is a plausible construction of a secure obfuscator.

In our full construction we go further and randomize the system as $(RA)M = (Rb)$ where R is a random invertible $m \times m$ matrix over $\mathbb{F}_q$. This is necessary when b is sampled from a non-uniform distribution. Hence the obfuscated program is the matrix RA and the hash value $h = H(Rb)$.

As we explain in Section 4 this construction gives an efficient obfuscator that provides input-hiding security whenever the distribution of instances is sufficiently varied. However it cannot handle projective sets and it looks at first sight like it reveals too much information about the original function because the span of the coefficient matrix A is published. This motivates our second construction of an obfuscator.

Our second construction is based on the functionality of small superset obfuscation. A small superset function takes as input a set Y and compares it with a reference set X: If $X \subset Y$ and if Y is "small" (namely $|Y| \leq t$ for some threshold t), then the function accepts Y. Equivalently let $x \in \{0,1\}^N$ and let $0 \leq t \leq N$ be a threshold. The small superset function takes as input $y \in \{0,1\}^N$ and outputs 1 if and only if $y - x \in \{0,1\}^N$ and $|y| \leq t$, where $|y|$ denotes the Hamming weight of y.

Obfuscators for small superset functions are known [3,11].

The obfuscator for the system $AM(x) = b$ is the following. Generate k random equations (A', b') and then mix the k dummy equations with the m real equations. Let s be a length $N := m + k$ binary string of Hamming weight m that indicates which rows are the "real" equations. The obfuscated program is the $(m+k) \times (\ell+1)$ augmented matrix (A^*, b^*) and the obfuscation of the small superset function f_s.

To evaluate the program one takes the point P, computes the corresponding monomial vector $M(x)$, computes $A^*M(x)$ and determines which rows equal b_i^*

for $1 \leq i \leq m + k$. Indicate those rows using a binary string s'. If $P \in S$ then the point will satisfy the real rows and may also satisfy a small number of the dummy rows. It follows that s' is a small superset of s and so is accepted by the obfuscated small superset function, for a suitable set of parameters (such that $t > m + k/q$). On the other hand, if the point P does not satisfy one of the polynomials f_i then the corresponding equations is not satisfied and s' is not accepted. One cannot break the obfuscator by simply solving $A^*M = b^*$; see Section 5.1 for explanation.

The intuition for this obfuscator is that, since the obfuscation of f_s hides s, one cannot tell which of the $m + k$ rows are the real polynomials and which are the dummy ones. There are $\binom{m+k}{m}$ possible subsets to choose from. So if m is not too small and k is chosen large enough, then it is plausible that this is input-hiding, since finding an accepting input would tremendously reduce the search space and lead to an attack to the input-hiding of the small superset obfuscator.

1.2 Organization

Section 2 gives mathematical preliminaries. Section 3 defines algebraic set membership functions and evasive algebraic set membership function families. Section 4 gives a hash-function-based obfuscator for affine algebraic set membership functions. Section 5 gives an obfuscator for both affine and projective algebraic set membership functions based on dummy equations and small superset obfuscation.

2 Preliminaries

2.1 Algebraic Geometry

Affine Algebraic Geometry Let k be an algebraic closed field. Let $k[x_1, \ldots, x_n]$ be the polynomial ring in n variables over k. The *n-dimensional affine space* over k is the set of n-tuples $\mathbb{A}_k^n = k^n = \{(x_1, \ldots, x_n) : x_i \in k\}$.

Let J be an ideal of $k[x_1, \ldots, x_n]$. We denote $V(J)$ to be the set of common roots $x \in \mathbb{A}_k^n$ of the polynomials in J. Let X be a subset of $\mathbb{A}_k^n$. We denote $I(X)$ to be the set of polynomials $f \in k[x_1, \ldots, x_n]$ vanish everywhere in X.

A subset $X \subseteq \mathbb{A}_k^n$ is an *algebraic set* (also called *algebraic variety*) if $X = V(I)$ for some ideal $I \subseteq k[x_1, \ldots, x_n]$. Every ideal $I \subseteq k[x_1, \ldots, x_n]$ is finitely generated, denoted $I = (f_1, \ldots, f_m)$, where $f_i \in I$. Every algebraic set is finitely generated, denoted $V(I) = V(f_1, \ldots, f_m) = V(f_1) \cap \cdots \cap V(f_m)$.

An algebraic set X is irreducible if $X = X_1 \cup X_2$ implies either $X = X_1$ or $X = X_2$, where X_1 and X_2 are algebraic sets. Every algebraic set is a finite union of irreducible algebraic sets. There is a notion of dimension that can be associated to an irreducible algebraic set. An irreducible algebraic set defined by $m < n$ equations in $k[x_1, \ldots, x_n]$ will have dimension at least $n - m$. In general, the dimension is equal to $n - m$, but it can be larger when the system does not form a complete intersection. When q is large, an irreducible algebraic set

of dimension d will have approximately q^d points; for small q estimations of the size of the solution set are more subtle and we do not go into this here.

One can also study algebraic sets over finite fields $\mathbb{F}_q$, and most of the above definitions also apply when working over a field that is not algebraically closed.

To give an intuition for why random systems of polynomial equations can be evasive (namely the size of the zero set is negligible compared to the whole variety), we present the following simple result.

Lemma 1. *Let $2 < m < n < \ell$ and let q be such that $q^{m-1} > 2^\lambda$. Fix a set of ℓ monomials in $\mathbb{F}_q[x_1, \ldots, x_n]$ of degree bounded by a polynomial. Consider the distribution on sets of m polynomials defined by uniformly sampling m polynomials $f_1, \ldots, f_m \in \mathbb{F}_q[x_1, \ldots, x_n]$ with monomials in the fixed set. Then with overwhelming probability there is not an irreduducible component of $V(f_1, \ldots, f_m)$ of dimension $n - 1$.*

Proof. Let $X = V(f_1, \ldots, f_m)$ and suppose X has an irreduducible component Z of dimension $n-1$. Since Z has codimension 1 it is of the form $V(g)$ for some irreducible polynomial $g \in \mathbb{F}_q[x_1, \ldots, x_n]$. Since $Z \subseteq X$ we have $I(X) \subseteq I(Z)$, and so $f_i \in I(Z)$ for all i. In other words, $g \mid f_i$ for all i.

Consider choosing f_1 uniformly. Then f_1 has polynomially many irreducible factors g. For each such g, consider choosing the remaining polynomials f_i. If f_i is chosen uniformly then the probability that $g \mid f_i$ is at most $1/q$. It follows that $g \mid f_i$ for all i occurs with probability at most $1/q^{m-1} < 1/2^\lambda$. □

Projective Algebraic Geometry The *n-dimensional projective space* over k is the set of nonzero $(n+1)$-tuples $\mathbb{P}_k^n = k^{n+1}\backslash\{0\}/\sim$, where $\sim$ is the equivalence relation given by $(x_0, \ldots, x_n) \sim (\alpha x_0, \ldots, \alpha x_n)$ for $\alpha \in k\backslash\{0\}$.

An ideal $I \subseteq k[x_0, \ldots, x_n]$ is homogeneous if it is generated by homogeneous polynomials. We define $V(J)$ and $I(X)$ similar to the affine case, with the only difference that the ideals here are homogeneous ideals. Projective algebraic sets and ideals satisfy similar properties as described in the affine case.

2.2 Input-Hiding Obfuscation

To introduce obfuscation, we use circuits to represent functions so that it is convenient to represent the time complexity of the function by the circuit size. By a circuit we always mean a circuit of minimal size that computes a specified function. The size complexity of a circuit of minimal size is polynomial in the time complexity of the function it computes.

We call auxiliary information about a specific circuit *local auxiliary information* and auxiliary information about the entire circuit family *global auxiliary information*. Local and global informations are also known as dependent and independent auxiliary informations, respectively.

The intuition of input-hiding is that given the obfuscated Boolean function, it should be inefficient for any PPT algorithm to find an element of the fiber of 1. We call elements of the fiber of 1 of a Boolean function *accepting inputs*.

Definition 1 (Input-Hiding Obfuscator [1]). *Let $\mathcal{C} = \{C_\lambda\}_{\lambda\in\mathbb{N}}$ be a Boolean circuit collection and $\mathcal{D}$ be a class of distribution ensembles $\mathcal{D} = \{D_\lambda\}_{\lambda\in\mathbb{N}}$, where D_λ is a distribution on C_λ. A probabilistic polynomial time (PPT) algorithm O is an input-hiding obfuscator for the family $\mathcal{C}$ and the distribution $\mathcal{D}$ if the following three conditions are met.*

1. Approximate Functionality Preserving: There is some negligible function $\mu(\lambda)$ such that for all $n \in \mathbb{N}$ and for all circuits $C \in \mathcal{C}$ with input size n we have that for all x such that $C(x) = 1$:

$$\Pr[O(C)(x) = 1] \geq 1 - \mu(\lambda),$$

and for all x such that $C(x) = 0$:

$$\Pr[O(C)(x) = 0] \geq 1 - \mu(\lambda),$$

where the probability is over the coin tosses of O.

2. Polynomial Slowdown: For every n, every circuit $C \in \mathcal{C}$, and every possible sequence of coin tosses for O, there exists a polynomial p such that the circuit $O(C)$ runs in time at most $p(|C|)$, i.e., $|O(C)| \leq p(|C|)$, where $|C|$ denotes the size of the circuit C.

3. Input-hiding: There exists a negligible function $\mu(\lambda)$ such that for every PPT adversary $\mathcal{A}$, for every $\lambda \in \mathbb{N}$ and for every global auxiliary information $\alpha \in \{0,1\}^{poly(\lambda)}$ about $\mathcal{C}$,

$$\Pr_{C\leftarrow D_\lambda}[C(\mathcal{A}(O(C), \alpha)) = 1] \leq \mu(\lambda),$$

where the probability is taken over the random sampling of D_λ and the coin tosses of $\mathcal{A}$ and O.

Note that we relaxed the functionality preserving property to the *approximate-functionality-preserving* property, which requires that a good input will be correctly accepted by the obfuscated function with overwhelming probability, also a bad input will be correctly rejected with overwhelming probability. Relaxed functionality preservations have been considered in numerous literatures [15,10,7,4].

Also note that input-hiding is only defined for evasive functions families because a function sampled from a non-evasive distribution always leaks accepting inputs.

Definition 2 (Evasive Circuit Collection [1]). *A collection of circuits $\mathcal{C} = \{C_\lambda\}_{\lambda\in\mathbb{N}}$, where C_λ takes $n(\lambda)$-bit input, is evasive if there exists a negligible function $\mu(\lambda)$ such that for all $\lambda \in \mathbb{N}$ and all $x \in \{0,1\}^{n(\lambda)}$,*

$$\Pr_{C\leftarrow C_\lambda}[C(x) = 1] \leq \mu(\lambda),$$

where the probability is taken over the random sampling of C_λ.

Previous obfuscated evasive functions include point functions [9,18], conjunctions [8,6,4], fuzzy Hamming distance matching functions [12], small superset functions [3,11], big subset functions [5,11], hyperplane membership functions [10], finite automatas [13], compute-and-compare functions [19,16], etc.

2.3 Small Superset Function

Small superset function obfuscation will be used in our construction. Instances of the small superset problem are parameterized by a pair of integers (N, t), which themselves are functions of a security parameter λ.

Definition 3 (Small Superset Function, SSF [3,11]). *Let $x \in \{0,1\}^N$ be a characteristic vector of a subset of $\{1,\ldots,N\}$ and $t \in \mathbb{N}$ with $0 \leq t \leq N$ be a threshold indicating "small". A small superset function with respect to x is a function $f_{x,t} : \{0,1\}^N \to \{0,1\}$ such that $f_{x,t}(y) = 1$ if and only if $y - x \in \{0,1\}^N$ and $|y| \leq t$.*

We will only consider evasive small superset functions. To avoid simply guessing an accepting input one needs $\binom{N}{t} = \binom{N}{N-t}$ to be super-polynomial in λ. Approximating $\binom{N}{t} > (N/t)^t > 2^\lambda$ it follows that $t \log_2(N/t) > \lambda$. For simplicity we will assume the very conservative choice $\lambda < t < N - \lambda$. Example parameters are $N = 4\lambda$ and $t = 2.1\lambda$.

We assume the existence of an input-hiding secure obfuscator for small superset functions that works for all parameters (N, t) where N is polynomial in the security parameter λ and $\lambda < t < N - \lambda$. The work of Bartusek, Carmer, Jain, Jin, Lepoint, Ma, Malkin, Malozemoff and Raykova [3] provides a general solution to obfuscation of SSF for all parameters in this range based on computational assumptions in abelian groups (their security proofs are in the generic group model).

3 Function Definition

In this paper we consider algebraic sets over finite fields $\mathbb{F}_q$ with q a prime power. The algebraic closure of $\mathbb{F}_q$ is the union of the finite fields $\mathbb{F}_{q^n}$ for $n \in \mathbb{N}$. We define affine and projective algebraic set membership functions in the following.

Definition 4 (Affine Algebraic Set Membership Function, a-ASMF). *Let $\mathbb{F}_q[x_1,\ldots,x_n]$ be a polynomial ring over a finite field $\mathbb{F}_q$ of order q. Let $M_1,\ldots,M_\ell \in \mathbb{F}_q[x_1,\ldots,x_n]$ be monomials that can be evaluated in polynomial time (i.e., with bounded degree) and assume $\ell > n$. Let $(A, b) \in \mathbb{F}_q^{m\times(\ell+1)}$ be the augmented matrix of the following system of equations over $\mathbb{F}_q$:*

$$\begin{aligned} a_{1,1}M_1 + \cdots + a_{1,\ell}M_\ell &= b_1, \\ &\vdots \\ a_{m,1}M_1 + \cdots + a_{m,\ell}M_\ell &= b_m. \end{aligned}$$

An affine algebraic set membership function is a Boolean function $f_{A,b} : \mathbb{F}_q^n \to \{0,1\}$ *such that* $f_{A,b}(x) = 1$ *if and only if* x *is a solution of the equations.*

Definition 5 (Projective Algebraic Set Membership Function, p-ASMF). *Let* $\mathbb{F}_q[x_0, \ldots, x_n]$ *be a polynomial ring over a finite field* $\mathbb{F}_q$ *of order* q*. Let* $M_1, \ldots, M_\ell \in \mathbb{F}_q[x_0, \ldots, x_n]$ *be fixed monomials of the same total degree that can be evaluated in polynomial time, and assume* $\ell > n$*. Let* $A \in \mathbb{F}_q^{m \times \ell}$ *be the coefficient matrix of the following system of homogeneous equations over* $\mathbb{F}_q$*:*

$$\begin{aligned} a_{1,1}M_1 + \cdots + a_{1,\ell}M_\ell &= 0, \\ &\vdots \\ a_{m,1}M_1 + \cdots + a_{m,\ell}M_\ell &= 0. \end{aligned}$$

A projective algebraic set membership function is a Boolean function $f_A : \mathbb{F}_q^n \to \{0,1\}$ *such that* $f_A(x) = 1$ *if and only if* x *is a nonzero root of the equations.*

Note that p-ASMF only accepts nonzero roots because 0 is not a point in projective space.

Evasive ASMF Family We show in the following that an algebraic set membership function can be learned if the proportion of accepting inputs is high.

Proposition 1. *Let* $f_{A,b}$ *be an algebraic set membership function and* V *be its algebraic set. If the proportion of accepting inputs of* $f_{A,b}$ *is* $1/p(\lambda)$ *for some polynomial* p*, then there exists a PPT adversary* $\mathcal{A}$ *which finds a basis of the ideal* $I(V)$ *given oracle access to* $f_{A,b}$*.*

Proof. Let m' be the row rank of (A, b), which can be guessed. We define $\mathcal{A}$ as follows. $\mathcal{A}$ samples random points x from $\mathbb{F}_q^n$ and queries the oracle for $f_{A,b}$ until it has collected $\ell - m'$ linearly independent vectors of the form $(M_1(x), \ldots, M_\ell(x))$ corresponding to accepting inputs x. This requires polynomially-many queries. $\mathcal{A}$ then solves for the orthogonal complement (A', b') of the $\ell - m'$ vectors, which can be done in polynomial-time. We have that $\mathrm{span}(A, b) = \mathrm{span}(A', b')$. Hence (A', b') is the coefficient matrix of a set of generators of $I(V)$. □

By Proposition 1, non-evasive ASMF are learnable. Hence it is only interesting to obfuscate evasive ASMF. We define evasive a-ASMF and p-ASMF families in the following. They follow from Definition 2 immediately.

Definition 6 (Evasive a-ASMF Family). *Let* λ *be the security parameter. Let* $m, \ell \in \mathbb{N}$ *with* $m < \ell$ *be polynomial in* λ*. Let* $\mathcal{C} = \{C_{m,\ell}\}_{m,\ell \in \mathbb{N}}$ *with* $C_{m,\ell} = \{f_{A,b}\}_{A,b \in \mathbb{F}_q^{m \times (\ell+1)}}$ *be a collection of a-ASMF functions. We say* $\mathcal{C}$ *is evasive if there exists a negligible function* $\mu(\lambda)$ *such that for every* $x \in \mathbb{F}_q^n$ *and for every* $\lambda \in \mathbb{N}$*,*

$$\Pr_{A,b \leftarrow \mathbb{F}_q^{m \times (\ell+1)}}[f_{A,b}(x) = 1] \leq \mu(\lambda). \tag{3}$$

In general, evasiveness is related to the dimensions of the irreducible factors of the algebraic set.

Example 1. Fix a prime q and ring $\mathbb{F}_q[x_1, \ldots, x_n]$. Consider the class $\mathcal{C}$ of algebraic sets parameterized by $(c_1, \ldots, c_n) \in \mathbb{F}_q^n$ defining the polynomials

$$f_1 = x_1(x_1 - c_1), f_2 = x_1(x_2 - c_2), \ldots, f_n = x_1(x_n - c_n).$$

This is a system of n polynomials in n variables. The algebraic set is the union of the two irreducible algebraic sets $V(x_1)$ and $V(x_1 - c_1, x_2 - c_2, \ldots, x_n - c_n) = \{(c_1, \ldots, c_n)\}$. The set $V(x_1)$ has dimension $n-1$ and membership is not evasive unless q is exponentially large, while the other component has dimension 0 and corresponds to a point function (which is evasive as long as $q^n > 2^\lambda$).

Hence, to be evasive we will need all irreducible components to have relatively small dimension. This also applies in the projective case.

Definition 7 (Evasive p-ASMF Family). *Let λ be the security parameter. Let $m, \ell \in \mathbb{N}$ with $m < \ell$ be polynomial in λ. Let $\mathcal{C} = \{C_{m,\ell}\}_{m,\ell \in \mathbb{N}}$ with $C_{m,\ell} = \{f_A\}_{A \in \mathbb{F}_q^{m \times \ell}}$ be a collection of p-ASMF functions. We say $\mathcal{C}$ is evasive if there exists a negligible function $\mu(\lambda)$ such that for every $x \in \mathbb{F}_q^n$ and for every $\lambda \in \mathbb{N}$,*

$$\Pr_{A \leftarrow \mathbb{F}_q^{m \times \ell}}[f_A(x) = 1] \leq \mu(\lambda). \tag{4}$$

Parameters For Evasive Function Family In the affine case, for an $x \in \mathbb{F}_q^n$, let $M = (M_1(x), \ldots, M_\ell(x)) \in \mathbb{F}_q^\ell$ be the vector with x plugged in. We assume M is nonzero (this condition is violated in Example 1 above). The left kernel of the vector $(M, -1)$ has dimension ℓ hence it is of order q^ℓ. Any m vectors in the kernel form a matrix (A, b) such that $AM = b$. So the number of $m \times (\ell + 1)$ matrices (A, b) such that $AM = b$ is $q^{m\ell}$. Also the number of all $m \times (\ell + 1)$ matrices is $q^{m(\ell+1)}$.

Suppose we are generating matrices A uniformly for our family. For evasiveness, we want the probability that a uniformly sampled matrix $(A, b) \leftarrow \mathbb{F}_q^{m \times (\ell+1)}$ satisfies $AM = b$ to be

$$\Pr_{(A,b) \leftarrow \mathbb{F}_q^{m \times (\ell+1)}}[f_{A,b}(x) = 1] = \frac{q^{m\ell}}{q^{m(\ell+1)}} = \frac{1}{q^m} \leq \frac{1}{2^\lambda}. \tag{5}$$

Similarly, the requirement for the projective case is

$$\Pr_{A \leftarrow \mathbb{F}_q^{m \times \ell}}[f_A(x) = 1] = \frac{q^{m(\ell-1)}}{q^{m\ell}} = \frac{1}{q^m} \leq \frac{1}{2^\lambda}. \tag{6}$$

Both Inequality (5) and (6) give

$$q^m \geq 2^\lambda. \tag{7}$$

Suppose $\lambda = 128$, three typical choices of m are as follows: if $q = 2$ then we require $m \geq 128$; if $q = 2^{80}$ then we require $m \geq 2$; if $q = 2^{128}$ then m can be as small as 1. In the last case the problem reduces to hypersurface membership [1], of which a special case is hyperplane membership [10]. For hyperplane membership, an advantage of our hashed-based obfuscation over the DLP-based obfuscation in [10] is that our scheme works for a much wider range of parameters than [10] which requires a prime modulus $q > 2^{256}$ for 128-bit security due to the $O(\sqrt{q})$ complexity of generic DLP solving algorithms such as the Baby-Step Giant-Step algorithm and Pollard's rho algorithm [17], while our scheme works for arbitrary prime power modulus $q > 2^{128}$ for 128-bit security.

Inequality (7) (i.e. $m \geq \lambda / \log_2 q$) gives the full generality of the ASMF obfuscation problem one can solve. In this paper we solve it in approximately full generality, i.e., $m > \lambda / \log_2 q$.

4 Hashed-Based Obfuscation for Affine Algebraic Sets

Let $\{f_{A,b}\}$ be a family of a-ASMF such that no column of (A, b) is a zero vector with overwhelming probability (in particular, $b \neq 0$). We obfuscate $f_{A,b}$ by hashing b. Let H be a cryptographic hash function (collision and preimage resistent) with λ-bit output. Given a basis matrix $(A, b) \in \mathbb{F}_q^{m \times (\ell+1)}$, the obfuscator publishes $(\bar{A}, H(\bar{b})) = (RA, H(Rb))$ as the obfuscated function, where R is a uniformly random $m \times m$ matrix which transforms the matrix (A, b) to (RA, Rb). To evaluate with an input $x \in \mathbb{F}_q^n$, it computes $\bar{A}M$ and returns the truth value of $H(\bar{A}M) = H(\bar{b})$, where M is the monomial vector with x plugged in.

Let $GL_m(\mathbb{F}_q)$ be the set of invertible $m \times m$ matrices with entries in $\mathbb{F}_q$. The obfuscator and the evaluation are given in Algorithms 1 and 2.

Algorithm 1 a-ASMF Obfuscator

Input: $\lambda, q, \ell, m \in \mathbb{N}$, $A, b \in \mathbb{F}_q^{m \times (\ell+1)}$
Output: $(\bar{A} \in \mathbb{F}_q^{m \times \ell}, h \in \mathbb{Z})$

1: sample $R \leftarrow GL_m(\mathbb{F}_q)$ and compute $(\bar{A}, \bar{b}) = R(A, b)$
2: hash $\bar{b}$ as $h = H(\bar{b})$
3: **return** $(\bar{A}, h)$

The evaluation is as follows, where M denotes a monomial vector and $M(x)$ denotes M with a concrete point $x \in \mathbb{F}_q^n$ plugged in.

Algorithm 2 a-ASMF Evaluation (with embedded data $M, \bar{A}, h$)

Input: $x \in \mathbb{F}_q^n$
Output: 0 or 1

1: compute $M(x)$
2: compute $\bar{A}M(x)$ and hash it as $h' = H(\bar{A}M(x))$
3: **return** truth value of $h' = h$

4.1 Security Proofs

We start by remarking that $\bar{b} = Rb$ is uniform as long as b is nonzero.

Lemma 2. *Let $\mathcal{C}$ be any distribution on vectors $b \in \mathbb{F}_q^m - \{0\}$. Let $\mathcal{D}$ be a distribution that works as the following: it samples $R \leftarrow GL_m(\mathbb{F}_q)$ and $b \leftarrow \mathcal{C}$ and outputs $\bar{b} = Rb$. Then $\mathcal{D}$ is the uniform distribution on $\mathbb{F}_q^m - \{(0, 0, ..., 0)\}$.*

Proof. We want to prove that (1) for every pair of nonzero vectors $b, c \in \mathbb{F}_q^m - \{0\}$ there exists an invertible matrix $R \in GL_m(\mathbb{F}_q)$ such that $Rb = c$; and (2) for every $b \in \mathbb{F}_q^m - \{0\}$, the numbers of invertible matrices $R \in GL_m(\mathbb{F}_q)$ such that $Rb = c$ are the same for all $c \in GL_m(\mathbb{F}_q)$.

To prove (1), extend b to a basis $\{b, b_2, ..., b_m\}$ for $\mathbb{F}_q^m$; extend c to a basis $\{c, c_2, ..., c_m\}$ for $\mathbb{F}_q^m$. Define the linear map L by $L(b) = c$ and $L(b_i) = c_i$. Then L is a linear map which maps a basis to a basis, so it is invertible.

To prove (2) define the stabiliser $Stab(b) = \{S \in GL_m(\mathbb{F}_q) : Sb = b\}$. Let $k = \#Stab(b)$ be the size of the stabilizer. If $Rb = c$ and $S \in Stab(b)$ then $(RS)b = c$. So we always have at least k matrices mapping b to c. Conversely, let R_1, R_2 be invertible matrices mapping b to c. Then $R_2^{-1}R_1$ maps b to b, so $R_2^{-1}R_1 \in Stab(b)$. But this means $R_1 = R_2S$ for some $S \in Stab(b)$. □

Theorem 1. *The obfuscator O_A given by Algorithm 1 is input-hiding on uniform evasive affine algebraic set membership functions assuming preimage resistance of the hash function H.*

Proof. (1)**Approximate-Functionality-Preserving.** First notice that $RAM = Rb$ and $AM = b$ have the same set of solutions since R is invertible.

Again, if $RAM(x) = Rb$ then $H(RAM(x)) = H(Rb)$. Conversely, if $H(RAM(x)) = H(Rb)$ then up to negligible probability of hash collisions we have that $RAM(x) = Rb$.

Therefore with overwhelming probability accepting inputs and rejecting inputs are correctly accepted and correctly rejected respectively by Algorithm 2.

(2) **Polynomial Slowdown.** Algorithm 2 evaluates m polynomials of bounded degree and computes a hash of the resulting vector $\bar{A}M$, which is efficient by the definition of ASMF and the efficiency of the hash function. Hence the entire algorithm takes polynomial time thus has polynomial slowdown compared to the original function.

(3) **Input-Hiding.** Let $\mathcal{A}$ be any PPT algorithm that breaks input-hiding of the a-ASMF obfuscator O_A given by Algorithm 1 with probability $\mu(\lambda)$. We construct an algorithm $\mathcal{B}$ against preimage resistance of H with success probability $\nu(\lambda)$ and show that $\nu(\lambda) = \mu(\lambda)$.

Algorithm $\mathcal{B}$ takes as input a uniformly sampled instance $h \in \{0,1\}^\lambda$. Associated with λ are parameters (n, m, q, ℓ) for the ASMF. The hash function has λ-bit output, and by equation (7) $q^m > 2^\lambda$. Hence, by Lemma 2, with overwhelming probability there exists a $\bar{b} = Rb \in \mathbb{F}_q^m$ such that $h = H(\bar{b})$, for some uniform invertible matrix R.

Algorithm $\mathcal{B}$ samples a sequence $M(x)$ of ℓ monomials in $\mathbb{F}_q[x_1, \ldots, x_n]$ and a random $m \times \ell$ matrix $\bar{A}$ without zero columns and calls $\mathcal{A}$ with $(\bar{A}, h)$. Note that $\bar{A}$ is of the form RA, where R is the same matrix in $\bar{b} = Rb$ and A is sampled from the ASMF distribution. Such a pair $(\bar{A}, \bar{b})$ defines an algebraic set which is non-empty with high (at least noticeable) probability $\xi(\lambda)$ since $m < n$, and so solutions $x \in \mathbb{F}_q^n$ exist. Then $\mathcal{A}$ will return, with probability $\mu(\lambda)$, a solution $x \in \mathbb{F}_q^n$ such that $H(\bar{A}M(x)) = h$. It follows that $\bar{A}M(x)$ is a preimage of h under H.

We can see that the probability that $\mathcal{B}$ breaks preimage resistance of H is ξ times the probability that $\mathcal{A}$ breaks input-hiding of O_A. I.e., $\nu(\lambda) = \xi(\lambda) \cdot \mu(\lambda)$. By the preimage resistance property of H, $\nu(\lambda)$ is negligible. Hence $\mu(\lambda)$ is negligible. Also note that $\mathcal{A}$ is an arbitrary PPT algorithm against the input-hiding of O_A. Hence O_A is input-hiding. □

The advantage of this obfuscator is its simplicity and efficiency. This obfuscator does not handle p-ASMF, because that case has $b = 0$. Also, the obfuscated program reveals the (row) span of the matrix A and so is not VBB secure.

5 Obfuscation Based on Dummy Equations

In this section we give an obfuscator for both a-ASMF and p-ASMF for certain ranges of parameters (essentially, when m is large enough; which is the case of interest for evasive boolean formulas). This obfuscator hides the span of A. We explain our techniques via the affine case. The projective case is similar.

We obfuscate an a-ASMF $f_{A,b}$ as follows. As before, we first perform a basis randomization on (A, b) to get $(\bar{A}, \bar{b}) = (RA, Rb)$. We then sample k random rows $(A', b') \leftarrow \mathbb{F}_q^{k \times (\ell+1)}$ for a suitably chosen k (see Section 5.2). We shuffle the $m + k$ rows of $(\bar{A}, \bar{b})$ and (A', b') and denote the resulting matrix as $(A^*, b^*) \in \mathbb{F}_q^{(m+k) \times (\ell+1)}$.

Let $s = (s_1, \ldots, x_{m+k}) \in \{0,1\}^{m+k}$ be the characteristic vector indicating the positions of the real rows, i.e., $s_i = 1$ if and only if the i-th row of (A^*, b^*) is a row of (A, b), for all $i \in \{1, \ldots, m+k\}$. Let f_s be an SSF on $\{0,1\}^{m+k}$ with the "small" threshold $m + k/q < t < m + k$, where $m + k/q$ is the expected number of rows in (A^*, b^*) satisfied by an arbitrary point in the algebraic set.

Let $\mathcal{O}_{\text{SSF}}$ be an input-hiding SSF obfuscator (e.g., the obfuscator in [3] or in [11]). We publish $(A^*, b^*, \mathcal{O}_{\text{SSF}}(f_s))$ as the obfuscated function. The details are given in Algorithm 3.

Algorithm 3 ASMF Obfuscator

Input: $\lambda, q, \ell, m \in \mathbb{N}$ with $q > 2^{\lambda/m}$, $A, b \in \mathbb{F}_q^{m \times (\ell+1)}$
Output: $(A^* \in \mathbb{F}_q^{(m+k)\times \ell}, b^* \in \mathbb{F}_q^{(m+k)}, \mathcal{O}_{\mathrm{SSF}}(f_s))$

1: compute a such that $2^{\lambda/m} < (1+a)/(1+a/q+o(1)) < q$
2: choose $k = am$, $t = \lceil (1+a/q+o(1))m \rceil$
3: sample $R \leftarrow GL_m(\mathbb{F}_q)$ and compute $(\bar{A}, \bar{b}) = R(A, b)$
4: sample $A' \leftarrow \mathbb{F}_q^{k \times \ell}$
5: **if** $b \neq 0$ **then** sample $b' \leftarrow \mathbb{F}_q^k$ **else** set $b' = 0$
6: randomly permute the rows of $((\bar{A}, \bar{b}), (A', b'))^\top$ to get (A^*, b^*)
7: create $s = (s_1, \ldots, s_{m+k}) \leftarrow \{0,1\}^{m+k}$ such that $s_i = 1$ if and only if $(A^*, b^*)_i = (\bar{A}, \bar{b})_j$ for some $j \in \{1 \ldots, m\}$, for all $i \in \{1, \ldots, m+k\}$
8: obfuscate the SSF f_s (with the "small" threshold t) as $\mathcal{O}_{\mathrm{SSF}}(f_s)$
9: **return** $(A^*, b^*, \mathcal{O}_{\mathrm{SSF}}(f_s))$

Note that for p-ASMF where $b = 0$, the dummy constant vector b' is set 0. This is simply because if otherwise we sample b' uniformly then only the dummy rows (A_i^*, b_i^*) can have $b_i^* = 1$ and the attacker can immediately spot them.

To evaluate the obfuscated paper on an input x: compute the monomial vector $M(x)$, then evaluate all $m+k$ equations on M and define a characteristic vector $s' = (s'_1, \ldots, s'_{m+k}) \in \{0,1\}^{m+k}$ such that $s'_i = 1$ if and only if M is a solution of the i-th equation, for all $i \in \{1, \ldots, m+k\}$. The obfuscated program eventually outputs what $\mathcal{O}_{\mathrm{SSF}}(f_s)$ outputs on s'. This is described in Algorithm 4.

One might think a trivial attack is to solve for M, but we explain in Section 5.1 why this does not work in general.

This approach does not apply for all choices of (q, n, m, ℓ). In particular m cannot be too small. For small q we need $m > \lambda/\log_2(q)$ and for larger q we essentially need $m > \lambda/\text{constant}$. This is why line 1 of the algorithm can only be formed if m satisfies certain conditions, such as $m > \lambda/\log_2(q)$, which will be discussed in Section 5.2.

A possible approach to improve this approach to cover all choices of (q, n, m, ℓ) is to perform what we called "basis extension" before running the algorithm. The idea of basis extension is as follows. Suppose $m = 2$ and the ideal $I(X) = (f_1, f_2)$ is generated by two polynomials f_1 and f_2. We extend the basis by sampling a lot of random polynomial pairs $(h_{i,1}, h_{i,2})$ and compute $g_i = h_{i,1} \cdot f_1 + h_{i,2} \cdot f_2$. For enough pairs $(h_{i,1}, h_{i,2})$ the g_i's is expected to give the same ideal as $I(X)$. Then we can obfuscate the algebraic set defined by the set of g_i's instead. However the security of basis extension is unclear and needs further study.

To help the reader have some intuition about the algorithm, let us look at two examples of parameters. Note that $2^{\lambda/m} < (1+a)/(1+a/q+o(1)) < q$. Suppose $q = 2$, $\lambda = 128$ and $m = 129$, we can take $a = 259$. Suppose $q = 3$, $\lambda = 128$ and $m = 81$, we can take $a = 794$.

Algorithm 4 ASMF Evaluation (with embedded data $(M, A^*, b^*, \mathcal{O}_{\text{SSF}}(f_s))$

Input: $x \in \mathbb{F}_q^n$
Output: 0 or 1

1: compute $y = A^* \cdot M(x) - b^*$
2: set $s' = (s'_1, \ldots, s'_{m+k}) \in \{0,1\}^{m+k}$ such that $s'_i = 1$ if and only if $y_i = 0$, for all $i \in \{1, \ldots, m+k\}$
3: **return** $\mathcal{O}_{\text{SSF}}(f_s)(s')$

Correctness First notice that $RAM = Rb$ and $AM = b$ have the same set of solutions since R is invertible.

Now if x is a solution then s' is a superset of s because x satisfies all rows of $(\bar{A}, \bar{b})$ indicated by s. Also s' is "small" with high probability, namely its Hamming weight $|s'| \leq t$ because an x is expected to satisfy $m + k/q$ rows but $m + k/q < t := \lceil (1 + a/q + o(1))m \rceil$. By choosing suitable number for the "$o(1)$" term we can ensure that the probability that x satisfies more than t rows is negligible in λ.[2] Hence s' is a small superset of s with overwhelming probability hence $\mathcal{O}_{\text{SSF}}(f_s)(s') = 1$ and the obfuscated a-ASMF will correctly output 1 with overwhelming probability.

On the other hand, if x is not a solution, then at least one of the rows of $(\bar{A}, \bar{b})$ will not be satisfied and s' will not be a superset of s. Then $\mathcal{O}_{\text{SSF}}(f_s)(s') = 0$ and the obfuscated a-ASMF will correctly output 0.

5.1 Failure of attack using linear algebra

One might think it is trivial to break the obfuscator: Just solve the system $A^*M = b^*$ using linear algebra. One might think that such a solution will in particular satisfy the "real" equations and hence give a point on the algebraic set. But there are two issues with this.

First, there may be no such solution. There is no guarantee that the dummy equations are consistent with the "real" equations, and so (especially when $m+k$ is substantially larger than ℓ) we would not expect there to be any such solution. Second, and more important, a solution to $A^*M = b^*$ does not necessarily correspond to a point $x = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ that satisfies the algebraic set. The issue is that most vectors $M \in \mathbb{F}_q^\ell$ are not of the form $M(x)$ where $M(x)$ is a vector of fixed monomials in $x_1, \ldots, x_n$ and $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$. Precisely, there are q^n possible choices for $x = (x_1, \ldots, x_n)$, giving at most q^n possible values for $M(x)$. But $M(x) \in \mathbb{F}_q^\ell$ and we generally have ℓ much bigger than n. This crucial remark is why we do not always insist that $m + k > \ell$. Though in practice it will often be the case that $m + k > \ell$.

[2] For example, suppose $q = 2$, $\lambda = 128$, $m = 129$, $a = 259$, $k = am = 33411$, $t = \lceil (1 + a/q + 9)m \rceil = 17996$, then the probability that a point in the algebraic set satisfies $t - m = 17867$ dummy rows is $\approx 2.008 \times 10^{-46} < 1/2^{128} \approx 2.939 \times 10^{-39}$.

5.2 Parameters For Obfuscation

Now we explain the parameters in Algorithm 3 and determine the generality of our solution to the ASMF obfuscation problem. In particular, we would like to investigate how big k and t should be and how they affect the parameters q, ℓ and m of the ASMF that can be securely obfuscated by our obfuscator.

Restrictions on the parameters come from three conditions: (1) Hardness of finding m generators from the $m+k$ polynomials; (2) Hardness of finding an accepting point x by finding $d < m$ generators from the $m+k$ polynomials; and (3) Evasiveness of SSF.

For condition (1), we require that the probability that finding the m generators by randomly choosing m polynomials from the $m+k$ polynomials is $1/\binom{m+k}{m} \leq 1/2^{\lambda}$. For condition (2), we require that the probability that finding an accepting input by randomly choosing $d < m$ polynomials and solving for a random root of the d polynomials is $\left(\binom{m}{d}/\binom{m+k}{d}\right) \cdot (1/q^{m-d}) \leq 1/2^{\lambda}$. For condition (3), we require $\binom{t}{m}/\binom{m+k}{m} \leq 1/2^{\lambda}$ by Inequality (5) in [11].

Let $k = am$ and $t = cm$ for some constants a and c. For condition (1), since $1/\binom{m+k}{m} \leq (m/(m+k))^m = 1/(1+a)^m$, it is sufficient to require $(1+a)^m \geq 2^{\lambda}$. For condition (2), since $\left(\binom{m}{d}/\binom{m+k}{d}\right) \cdot (1/q^{m-d}) \leq (m/(m+k))^d \cdot (1/q^{m-d}) = (1/(1+a)^d) \cdot (1/q^{m-d})$, it is sufficient to require $(1+a)^d \cdot q^{m-d} \geq 2^{\lambda}$. For condition (3), since $\binom{t}{m}/\binom{m+k}{m} \leq (t/(m+k))^m = (c/(1+a))^m$, it is sufficient to require $((1+a)/c)^m \geq 2^{\lambda}$.

We first consider the case when $2 \leq q \leq poly(\lambda)$. If we take $a > q-1$, then $(1+a)^d \cdot q^{m-d} < (1+a)^m$, hence condition (1) is implied by condition (2). Also $q^m < (1+a)^d \cdot q^{m-d}$, hence condition (2) is already implied by Inequality (7). We therefore only need to consider condition (3). It gives a restriction on m as

$$m \geq \lambda/(\log_2(1+a)/c). \tag{8}$$

Note that a basic constraint is $m + k/q < t < m + k$. I.e., $1 + a/q < c < 1 + a$. We therefore take $c = 1 + a/q + \varepsilon$ for some small constant $\varepsilon = o(1)$. Then we can see that

$$\begin{aligned}
&\lim_{a\to\infty} \frac{1+a}{c} \\
=&\lim_{a\to\infty} \frac{1+\frac{a}{q}+\varepsilon+\frac{(q-1)a}{q}-\varepsilon}{1+\frac{a}{q}+\varepsilon} \\
=&\lim_{a\to\infty} \left(1+\frac{\frac{(q-1)a}{q}-\varepsilon}{1+\frac{a}{q}+\varepsilon}\right) \\
=&q.
\end{aligned}$$

I.e., when $a \to \infty$, condition (3) approaches to the evasive requirement of ASMF: $q^m \geq 2^{\lambda}$ (i.e., Inequality (7)). This means that if we take a to be larger and larger, our solution will get closer and closer to a full solution to the ASMF obfuscation

problem. However we can only take a to be a polynomial otherwise the size of the obfuscated function will have super-polynomial blow up. But it is almost full generality of ASMF. To see this, take our previous example after Algorithm 3, when $q = 2$ (this is actually the worst case to get a full range of m), ideally a full solution to the problem should work on ASMF with $m \geq 128$. However we have already covered $m \geq 129$, which is just one equation more. In other words, the full generality is $m \geq \lambda/\log_2 q$ and we have achieved $m > \lambda/\log_2 q$.

We therefore choose large enough a such that $2^{\lambda/m} < (1+a)/(1+a/q+o(1)) < q$ to reach to the greatest generality of our solution. At the same time t is implicitly about $(1 + a/q + o(1))m$.

Now we consider the case when $q \geq \mathit{spoly}(\lambda)$ for any super-polynomial $\mathit{spoly}(\lambda)$. In this case, we have $a < q-1$ since a must grow at most polynomially in λ (for polynomial blow up of the function size). Then $(1+a)^d \cdot q^{m-d} > (1+a)^m$, hence condition (2) is implied by condition (1). Also, since $c \geq 1$, we have $(1+a)^m > ((1+a)/c)^m$, hence condition (1) is implied by condition (3). Therefore the restriction on m is still from condition (3). It is the same as given by Inequality (8).

In sum, we choose $k = am$ for a sufficient large a such that $(1+a)/(1+a/q+o(1)) > 2^{\lambda/m}$ and choose $t = \lceil (1 + a/q + o(1))m \rceil$. Then the ASMF family that our obfuscator can obfuscate approaches to its full generality given by Inequality (7).

We stress that we do not require any relation between $m + k$ and n or ℓ. If one wanted VBB security one might need to consider $m + k > \ell$, but we do not consider this question in our paper.

5.3 Security Proofs

We reduce security of the obfuscator O given by Algorithm 3 to the input-hiding security of the small superset obfuscator $\mathcal{O}_{\mathrm{SSF}}$.

To give a general result we would need to explain how, given parameters (N, t) for a small superset instance, to associate parameters (q, ℓ, m) to an algebraic set membership instance. As a working example, consider the case $N = 4\lambda$ and $t = 2.1\lambda$ from Section 2.3. To such parameters we can choose $q = 2$, $m = \lambda$, $a = 2$ so that $t > (1 + a/q + o(1))m = (2 + o(1))\lambda$, and $N > (1 + a)m = 3\lambda$. When $t > N/2$ we can use the same parameters $(q, m, a) = (2, \lambda, 2)$, since $t > (1+a/q+o(1))m$ and $\mathcal{O}_{\mathrm{SSF}}$ will still accept the string. For $N/3 < t < N/2$ we can take $(q, m, a) = (2, \lambda, 2)$ and require $t > (1+a/q+o(1))m$ and $N \geq 6\lambda$. Note that $m = \lambda$ satisfies the lower bound $m > \lambda/\log(\lambda)$ required in Section 5.2. When $t \leq N/3$ the general strategy is to choose $q = \lfloor N/t \rfloor$ and m and a accordingly. Note that $q = \lfloor N/t \rfloor < N$ is always polynomial in the security parameter.

In other words, to be convinced of the security of the obfuscator for algebraic set membership instances over $\mathbb{F}_q$ when $q > 3$, one needs a secure small superset obfuscator for $(N, t) \approx ((q + 1)m, 2m)$ by taking $a = q$.

Theorem 2. *The obfuscator O given by Algorithm 3 is input-hiding on uniform evasive algebraic set membership functions assuming input-hiding of the small superset obfuscator $\mathcal{O}_{SSF}$ on the associated evasive parameters (N, t).*

Proof. (1) **Approximate-Functionality-Preserving.** It is shown after Algorithm 4.

(2) **Polynomial Slowdown.** Algorithm 4 evaluates $N = m + k = (1 + a)m$ polynomials of bounded degree, generates a vector s' of polynomial length N, and evaluates $\mathcal{O}_{\text{SSF}}(f_s)$ on s', which is efficient by the polynomial slowdown of $\mathcal{O}_{\text{SSF}}$. Since a is polynomial, Algorithm 4 takes polynomial time thus has polynomial slowdown compared to the original function.

(3) **Input-Hiding.** Let $\mathcal{A}$ be any PPT algorithm that breaks input-hiding of the ASMF obfuscator O given by Algorithm 3 with probability $\mu(\lambda)$. We construct an algorithm $\mathcal{B}$ against input-hiding of the SSF obfuscator $\mathcal{O}_{\text{SSF}}$ with success probability $\nu(\lambda)$ and show that $\nu(\lambda) = \mu(\lambda)$.

Given an obfuscated SSF $\mathcal{O}_{\text{SSF}}(f_s)$ with parameters satisfying the requirements that the function is evasive, together with some global auxiliary information $\alpha \in \{0,1\}^{poly(\lambda)}$ of the SSF family, $\mathcal{B}$ finds a small superset s' of s as follows.

To create an obfuscated ASMF instance, $\mathcal{B}$ first selects the parameters m, k and ℓ. Then $\mathcal{B}$ samples a random $(m + k) \times (\ell + 1)$ matrix (R^*, r^*) and calls $\mathcal{A}$ with $(R^*, r^*, \mathcal{O}_{\text{SSF}}(f_s), \alpha, \beta)$, where $\beta \in \{0,1\}^{poly(\lambda)}$ is the global auxiliary information of the ASMF family.

If $\mathcal{A}$ wins then it outputs a vector $x \in \mathbb{F}_q^n$ whose corresponding monomial vector M satisfies $RM = r$, where (R, r) are the rows indicated by s. Since the distribution of (R^*, r^*) and (A^*, b^*) (as in Algorithm 3) are the same, $\mathcal{A}$ will return a point $x \in \mathbb{F}_q^n$ which tells a small superset $s' \in \{0,1\}^{m+k}$ of s with probability $\mu(\lambda)$. Then $\mathcal{B}$ outputs s'. We see that the probability that $\mathcal{B}$ breaks input-hiding of $\mathcal{O}_{\text{SSF}}$ is equal to the probability that $\mathcal{A}$ breaks input-hiding of O. Hence $\nu(\lambda) = \mu(\lambda)$. By the input-hiding property of $\mathcal{O}_{\text{SSF}}$, $\nu(\lambda)$ is negligible. Hence $\mu(\lambda)$ is negligible. Also note that $\mathcal{A}$ is an arbitrary PPT algorithm against the input-hiding of O. Hence O is input-hiding. □

Now we introduce a new security notion. Note that the asset that we are trying to protect is the ideal associated with the algebraic set. Generators for this ideal are not uniquely defined, since an ideal can have many different sets of generators. The set of all monomials required to express generators of the ideal (this set may contain some unnecessary monomials) is not protected by our obfuscation, but what we can hope to protect is the (row) span as a vector space of the set of generators.

Definition 8 (Span-Hiding Obfuscator). *Let $\mathcal{C} = \{C_\lambda\}_{\lambda \in \mathbb{N}}$ with respect to ℓ fixed monomials $M_1, \ldots, M_\ell \in \mathbb{F}_q[x_1, \ldots, x_n]$ be an ASMF family, where m, n, ℓ are polynomial in λ. Let $\mathcal{D} = \{D_\lambda\}_{\lambda \in \mathbb{N}}$ be the corresponding distributions of $\mathcal{C}$, which are distributions over the augmented matrices $(A, b) \in \mathbb{F}_q^{m \times (\ell+1)}$ of the ASMFs. A PPT algorithm O is a* span-hiding obfuscator *for the family $\mathcal{C}$ if it satisfies the first two conditions in Definition 1 and the following span-hiding property: there exists a negligible function $\mu(\lambda)$ such that for every $\lambda \in \mathbb{N}$ and*

for every auxiliary input $\alpha \in \{0,1\}^{poly(\lambda)}$ *to* $\mathcal{A}$*:*

$$\Pr_{(A,b)\leftarrow D_\lambda}[\mathcal{A}(O(A,b),\alpha) = Span(A)] \leq \mu(\lambda),$$

where the probability is taken over the random sampling of D_λ *and the coin tosses of* $\mathcal{A}$ *and* O*.*

We stress that our hash-based obfuscator is trivially not span-hiding. We prove that our obfuscator based on dummy equations is span-hiding if the SSF obfuscator is input-hiding.

We also stress that we make no claim about span-hiding security if an attacker knows an accepting input. However since we are restricting to evasive functions it is natural to consider span-hiding security against attackers who do not know an accepting input.

One further remark, knowing the span of A — even knowing the span of (A,b) is not immediately equivalent to finding an accepting input. Given (A,b) one can solve the system of equations $AM = b$ to get a vector $M \in \mathbb{F}_q^\ell$, but an accepting input is an $x \in \mathbb{F}_q^n$ whose corresponding monomial vector M satisfies $AM = b$. Going from M to x may be nontrivial depending on the monomials that appear.

Theorem 3. *The obfuscator given by Algorithm 3 is span-hiding on uniform evasive algebraic set membership functions assuming input-hiding of the small superset obfuscator* $\mathcal{O}_{SSF}$ *it uses and that* $\ell \geq n$*.*

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ which, with noticeable probability, computes the span of A from the program obfuscated using Algorithm 3, then there exists a PPT algorithm $\mathcal{B}$ which breaks input-hiding of the obfuscator $\mathcal{O}_{\text{SSF}}$. The structure of the proof is similar to the proof of Theorem 2.

The algorithm $\mathcal{B}$ takes as input an obfuscated SSF $\mathcal{O}_{\text{SSF}}(f_s)$ for some parameters (N,t), and some global auxiliary information $\alpha \in \{0,1\}^{poly(\lambda)}$. Algorithm $\mathcal{B}$ wants to compute an accepting input s'.

Algorithm $\mathcal{B}$ samples a random $(m+k)\times(\ell+1)$ matrix (R^*, r^*) and calls $\mathcal{A}$ with $(R^*, r^*, \mathcal{O}_{\text{SSF}}(f_s), \alpha, \beta)$, where $\beta \in \{0,1\}^{poly(\lambda)}$ is additional global auxiliary information.

By assumption, $\mathcal{A}$ outputs a basis for a vector space V of dimension m. Algorithm $\mathcal{A}$ wins if V is the span of A, where A is by definition the matrix given by the rows of R^* such that $s_i = 1$. Suppose this is the case.

Algorithm $\mathcal{B}$ then considers each of the N rows of R^*. For each $1 \leq i \leq N$, if the i-th row lies in V then it sets $s'_i = 1$, and otherwise sets $s'_i = 0$. It outputs s'.

We now argue that s' is a "small superset" of s.

We first argue that s' is a "superset" of s. Since we are assuming $\mathcal{A}$ wins, all the rows in A have $s'_i = 1$. Hence s' is a "superset" of s.

Now we argue that s' is "small". The dummy rows of R^* are sampled uniformly from $\mathbb{F}_q^\ell$. The probability that a row lies in the m-dimensional space V is

$q^m/q^\ell = 1/q^{\ell-m}$. Now $\ell \geq n$ so Equation (2) implies $q^{\ell-m}$ is super-polynomial. Hence we expect $s'_i = 1$ with negligible probability for each dummy row. We therefore expect that s' has exactly m entries set to 1, which means that s' is expected to be a "small" set that is of the same size m as s. □

References

1. Barak, B., Bitansky, N., Canetti, R., Kalai, Y.T., Paneth, O., Sahai, A.: Obfuscation for evasive functions. In: Lindell, Y. (ed.) Theory of Cryptography. pp. 26–51. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
2. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S., Yang, K.: On the (im) possibility of obfuscating programs. In: Annual International Cryptology Conference (CRYPTO). pp. 1–18. Springer (2001)
3. Bartusek, J., Carmer, B., Jain, A., Jin, Z., Lepoint, T., Ma, F., Malkin, T., Malozemoff, A.J., Raykova, M.: Public-key function-private hidden vector encryption (and more). In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology – ASIACRYPT 2019. pp. 489–519. Springer International Publishing, Cham (2019)
4. Bartusek, J., Lepoint, T., Ma, F., Zhandry, M.: New techniques for obfuscating conjunctions. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 636–666. Springer International Publishing, Cham (2019)
5. Beullens, W., Wee, H.: Obfuscating simple functionalities from knowledge assumptions. In: Lin, D., Sako, K. (eds.) Public-Key Cryptography – PKC 2019. pp. 254–283. Springer International Publishing, Cham (2019)
6. Bishop, A., Kowalczyk, L., Malkin, T., Pastro, V., Raykova, M., Shi, K.: A simple obfuscation scheme for pattern-matching with wildcards. In: Annual International Cryptology Conference (CRYPTO). pp. 731–752. Springer (2018)
7. Bishop, A., Kowalczyk, L., Malkin, T., Pastro, V., Raykova, M., Shi, K.: A simple obfuscation scheme for pattern-matching with wildcards. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018. pp. 731–752. Springer International Publishing, Cham (2018)
8. Brakerski, Z., Vaikuntanathan, V., Wee, H., Wichs, D.: Obfuscating conjunctions under entropic ring lwe. In: Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science. pp. 147–156. ITCS '16, ACM, New York, NY, USA (2016). https://doi.org/10.1145/2840728.2840764,
9. Canetti, R.: Towards realizing random oracles: Hash functions that hide all partial information. In: Kaliski, B.S. (ed.) Advances in Cryptology — CRYPTO '97. pp. 455–469. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)
10. Canetti, R., Rothblum, G.N., Varia, M.: Obfuscation of hyperplane membership. In: Theory of Cryptography Conference (TCC). pp. 72–89. Springer (2010)
11. Galbraith, S.D., Li, T.: Small superset and big subset obfuscation. In Baek, J. and Ruj, S. (eds.) Information Security and Privacy - 26th Australasian Conference, ACISP 2021, Springer LNCS 13083, pp. 68–87 (2021)
12. Galbraith, S.D., Zobernig, L.: Obfuscated fuzzy Hamming distance and conjunctions from subset product problems. In: Hofheinz, D., Rosen, A. (eds.) Theory of Cryptography, LNCS 11891, pp. 81–110. Springer International Publishing, Cham (2019)
13. Galbraith, S.D., Zobernig, L.: Obfuscating finite automata. in Dunkelman, O., Jacobson, M. J., and O'Flynn, C. (eds.), Selected Areas in Cryptography (SAC) 2020, Springer LNCS 12804 (2020) pp. 90–114.

14. Goldwasser, S., Kalai, Y.T.: On the impossibility of obfuscation with auxiliary input. In: 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05). pp. 553–562. IEEE (2005)
15. Goldwasser, S., Rothblum, G.N.: On best-possible obfuscation. In: Theory of Cryptography Conference (TCC). pp. 194–213. Springer (2007)
16. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS). pp. 612–621 (2017)
17. Pollard, J.M.: Monte carlo methods for index computation (mod p). Mathematics of computation **32**(143), 918–924 (1978)
18. Wee, H.: On obfuscating point functions. In: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing. pp. 523–532. ACM (2005)
19. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under lwe. In: 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS). pp. 600–611. IEEE (2017)