Algebraic Membership Obfuscation

by

Trey Li

Department of Mathematics

University of Auckland

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy in Mathematics, the University of Auckland, 2021.

ABSTRACT

An algebraic set is the solution set of a system of algebraic equations. The usual way to determine if a point is in an algebraic set is to determine if the point satisfies all the equations. This dissertation studies how to give membership testing of an algebraic set without selling out the equations.

The discrete logarithm problem (DLP) is reduced to the subset product problem (SP). The twin subset product problem (TSP) with respect to twin points is defined. The multiple subset product problem (MSP) with respect to relative points is defined. A lattice attack to low density MSP with identical points is given. In order to avoid the attack, the subset product with errors problem (SPE) is proposed. SP is reduced to SPE. The twin subset product with errors problem (TSPE) is proposed. The multiple subset product with errors problem (MSPE) is proposed. With a DLP algorithm, MSPE can be converted into a similar form to the learning with errors problem (LWE). The ideal subset product with errors problem (ISPE) is proposed. SPE, TSPE, MSPE and ISPE are conjectured to be post-quantum hard.

A new obfuscation for small superset and big subset testing is given based on SP. A new obfuscation for conjunctions is given based on TSP. The new obfuscations are more elegant than the previous works.

An obfuscation for algebraic set membership testing is given based on small superset obfuscation. This is the first solution to tackle the problem in great generality.

From algebraic membership obfuscation, a general obfuscation method is given to deal with membership problems in generic topological spaces. Topology is generalized to genology, giving a more flexible way to represent mathematical objects that have finite generators. The general method gives a new view to look at several open problems, revealing a fundamental relation between these seemingly unrelated problems.

ACKNOWLEDGMENTS

I have kept in mind a big list of people to thank, which this page is too small to contain. Then I decided to go the other way and make it as simple as possible. Anyone who is not named here does not mean that I forgot you.

I am eternally grateful to my supervisor Steven D. Galbraith for his immense support and guidance during my PhD journey. I appreciate all his help at all the key moments and all the beautiful elements he brings into my life.

I would like to thank my colleges Yan Bo Ti, Lukas Zobernig, Samuel Dobson, Yi-Fu Lai as well as Shujie Cui, Shalini Banerjee and my co-supervisor Giovanni Russello for being great people to work with.

I would like to thank Ron Steinfeld and Felipe Voloch who indirectly inspired me on some works of this dissertation.

I would like to thank Mark Zhandry and Amin Sakzad for their comments and questions.

I would like to thank the anonymous conference reviewers for their unusually abundant and enlightening comments by which I was quite touched.

I would like to thank Libin Wang and Xiaoping Du for their support of my PhD.

I would like to thank my family for their support of my study.

I give my gratitude to everyone I love and everyone who loves me.

CONTENTS

Chapter 1

INTRODUCTION

The goal of this dissertation is to introduce algebraic geometry to function obfuscation by obfuscating algebraic set membership.

The motivation of function obfuscation is to hide a function without affecting its usage. As we all know, a function f is a relation that associates each element x of a set X to a single element y of another set Y. It is uniquely identified by the set of all pairs $(x, f(x))$. A function that can be executed by an algorithm is called a computable function. A computable function may be computed by different programs or circuits. Some programs or circuits are easy to understand and some are not. Given a function computed by f, we would like to find an algorithm O to convert f into $O(f)$ such that $O(f)$ is functionally equivalent to f but hard to understand. We call O an obfuscator and $O(f)$ an obfuscated function.

The strongest notion of obfuscation is virtual black-box (VBB) [BGIRSVY01], as well as its various variants [GK05; BBCKPS14; WZ17]. It requires to hide all predicates of f beyond what is leaked by black-box access to the function. This is the most we can hide about f without affecting its functionality. However VBB is impossible to achieve for general functions. Hence in the same paper the authors proposed a weaker notion called indistinguishability obfusction (iO). It avoids the impossibility of VBB and has been proved to be the best-possible obfuscation [GR07] under certain restrictions.

Since the births of VBB and iO, the research has been divided into two paths: special purpose obfuscation and general purpose obfuscation. Special purpose obfuscation obfuscates restricted kinds of functions. For example, point functions [Can97; Wee05], conjunctions [BVWW16; BKMPRS18; BLMZ19], fuzzy Hamming distance matching functions [GZ19], big subset functions [BW19], small superset functions [BCJJLMMMR19], hyperplane membership functions [CRV10], finite automatas [GZ20], compute-and-compare functions [WZ17; GKW17], etc. [LPS04]. General purpose obfuscation aims at obfuscating general functions in a uniform way. Milestone works on this line include [GGH13; GGHRSW13; SW14; AJ15; BV15; Lin16].

Compared to VBB and iO, input-hiding [BBCKPS14] is really the main security property of interest in many applications such as password checks and biometric authentication. It is also a notion that is mathematically more intuitive than VBB and

iO. It simply aims at hiding the fiber of a particular element in the codomain of the function. It is traditionally defined as hiding the fiber of 1 of a Boolean function. It therefore is a notion only for evasive functions [BBCKPS14], which are the kind of Boolean functions that have a relatively small fiber of 1 compared to the entire domain. The kind of algebraic set membership functions of interest are also evasive.

Now we give motivation to algebraic geometry problems. In short it is due to its significantly important relations with many important areas.

During history, from the early introduction of coordinate geometry by René Descartes and Pierre de Fermat, to the algebraization through commutative algebra by David Hilbert, and then to the foundation recasting using sheaf theory by Jean-Pierre Serre and Alexander Grothendieck, algebraic geometry makes itself a bridge between many important disciplines such as analytic geometry and topology.

As the fundamental object of study in algebraic geometry, algebraic sets have deep connections to the central objects in other disciplines such as analytic subspaces and topological subspaces. By Chow's theorem, all closed analytic subspaces are actually algebraic sets. In particular, all projective complex manifolds are algebraic sets, meaning that they are all given by the zero locus of a finite number of homogeneous polynomials. Jean-Pierre Serre's GAGA theorems further give general results to associate categories of algebraic sets with categories of analytic spaces, unifying the study of analytic geometry with algebraic geometry. Also, algebraic subsets are closed sets of the Zariski topology on the variety. By generalizing the base ring of the variety to the spectrum of an arbitrary commutative ring the variety is converted into a scheme, giving it a much more fruitful structure to study.

Also notice that these connections do not only have interest in pure mathematics, they have had important applications in broad areas such as physics, cosmology, biology, computer science, etc.

Due to the significance of algebraic geometry, this dissertation attempts to introduce algebraic geometry problems to function obfuscation. To achieve this, we aim at two targets: (1) input-hiding obfuscation for algebraic set membership, and (2) a general obfuscation method for objects that are comparable to algebraic sets.

Following is the roadmap to achieve the two targets.

Chapter 2 gives mathematical preliminaries.

Chapter 3 gives a family of computational problems that serve as a tool set for matching obfuscation, which is a component of the general method introduced in Chapter 6.

In particular, the discrete logarithm problem (DLP) is reduced to the subset product problem (SP). The twin subset product problem (TSP) and the multiple subset product problem (MSP) are proposed. A quantum attack to low density MSP is presented. In order to avoid this attack, the subset product with errors problem (SPE) is proposed, as well as its twin and multi-instance versions called the twin subset product with errors problem (TSPE) and the multiple subset product with errors problem (MSPE). SPE is at least as hard as SP. With a DLP algorithm, MSPE can be converted into the multiple subset sum with errors problem (MSSE), which has a similar form to the learning with errors problem (LWE). The ideal version of SPE is also given, which is called the ideal subset product with errors problem (ISPE). Post-quantum hardness of SPE, TSPE, MSPE and ISPE are conjectured.

From Chapter 4 on, we start our journey to the general method which consists of three components: object representation, object randomization and matching obfuscation. We study them backwards.

Chapter 4 studies matching obfuscation. In particular, small superset as well as big subset functions for $r \leq n/2 - \sqrt{\lambda n \ln 2}$ are obfuscated using SP, where r is the maximal difference of sizes allowed between an accepting input set and the set being obfuscated. Small superset obfuscation serves as a building block of the algebraic set membership obfuscation.

Chapter 5 studies object randomization. The algebraic set membership obfuscation is proposed in this chapter. The idea to allow membership testing of an algebraic set without leaking its ideal is to mix the ideal generators with some dummy polynomials such that a point is in the algebraic set if and only if all real generators vanish at the point. Let λ be the security parameter, q be the order of the ground field, ℓ be the number of monomials of bounded degree, and m be the number of polynomial equations. The obfuscation method solves the algebraic set membership obfuscation problem for prime powers $q \geq 2$ and $\lambda/\log_2 q < m < \ell$.

Chapter 6 studies object representation. The general obfuscation method is proposed in this chapter. The fundamental idea to generalize the algebraic set membership obfuscation is to view algebraic sets as closed sets of the Zariski topology, and that an algebraic set of an ideal is the intersection of the algebraic sets of the ideal generators. Beyond this view algebraic set membership is immediately extended to membership problems of open subsets of generic topological spaces. To better describe the generality of the method, topology is generalized to "genology" to describe the daily concept of "generation" in mathematics, giving a more flexible way to represent mathematical

objects that have finite generators.

Chapter 7 discusses related problems, showing how the general obfuscation method provides new views to these problems and how these seemingly unrelated problems are actually related via the algebraic set membership problem. Till then the goal of this dissertation is fulfilled.

Chapter 8 discusses future work.

Chapter 2

MATHEMATICAL BACKGROUND

This chapter gives necessary mathematical background to this dissertation.

2.1 Topology

Let X be a set. A topology τ on X is a collection of subsets of X containing $\varnothing$ and X and that is closed under finite and infinite unions and finite intersections. The elements of τ are called *open sets*. The complement $X \backslash S$ of an open set $S \in \tau$ is called a *closed set*. We also call them *open subsets* and *closed subsets* of X, respectively. The pair (X, τ) is called a *topological space*. We sometimes simply call X a topological space, with τ implied.

Let X, Y be two topological spaces. A function $f : X \to Y$ is *continuous* if the preimage $f^{-1}(S)$ of every open set S of Y is an open set of X. A *homeomorphism* (also called *bicontinuous function*) is a continuous function $f : X \to Y$ between topological spaces that is bijective and has continuous inverse f^{-1}.

A *manifold* M is a topological space that is locally Euclidean, i.e., every point on M has an open neighborhood that is homeomorphic to an open ball of the Euclidean space $\mathbb{R}^n$. A map is said to be *smooth* if it is arbitrary differentiable. A *smooth manifold* is a manifold that has smooth structure, that is, a topological space that has an open cover $\{U_\alpha\}$ with homeomorphisms $\varphi_\alpha : U_\alpha \to \mathbb{R}^n$ such that on each intersection $U_\alpha \cap U_\beta$, the maps $\varphi_\beta \circ \varphi_\alpha^{-1} : \mathbb{R}^n \to \mathbb{R}^n$ are all smooth maps.

2.2 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 $V(I) = V(f_1, \ldots, f_m) = V(f_1) \cap \cdots \cap V(f_m)$.

Projective Algebraic Geometry The *n-dimensional projective space* over k is the set of $(n+1)$-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 discussed in the affine case.

Sheaves and Schemes Let X be a topological space. A *presheaf* $\mathcal{F}$ on X assigns to each open set U of X a set $\mathcal{F}(U)$, also denoted $\Gamma(U, \mathcal{F})$, and to every pair of open sets $U \subseteq V \subseteq X$ a restriction map $\mathrm{res}_{V,U} : \mathcal{F}(V) \to \mathcal{F}(U)$ satisfying: (1) $\mathrm{res}_{U,U} : \mathcal{F}(U) \to \mathcal{F}(U)$ is the identity map on $\mathcal{F}(U)$; and (2) $\mathrm{res}_{W,V} \circ \mathrm{res}_{V,U} = \mathrm{res}_{W,U}$ for open sets $U \subseteq V \subseteq W \subseteq X$. The restriction $\mathrm{res}_{V,U}(s)$ of an element $s \in \mathcal{F}(V)$ is also denoted $s|_U$.

A *sheaf* is a presheaf such that for any open covering $\mathcal{U} = \{U_i\}_{i \in I}$ of an open set U: (1) (locality) the equality $s|_{U_i} = t|_{U_i}$ of the restrictions of $s, t \in \mathcal{F}(U)$ implies the equality $s = t$ of the original elements, for all $U_i \in \mathcal{U}$; and (2) (gluing) if a section $s_i \in \mathcal{F}(U_i)$ is given for every U_i such that for each pair U_i, U_j the restrictions $s_i|_{U_i \cap U_j} = s_j|_{U_i \cap U_j}$, then there is a section $s \in \mathcal{F}(U)$ such that $s|_{U_i} = s_i$ for each $i \in I$.

Let $x \in X$ and i be the inclusion of the subspace $\{x\}$ into X. Let $\mathcal{F}$ be a sheaf on on X. Then the *stalk* of $\mathcal{F}$ at x, denoted $\mathcal{F}_x$, is the inverse image sheaf $i^{-1}\mathcal{F}$, i.e., $\mathcal{F}_x = i^{-1}\mathcal{F}(\{x\})$.

Let X be a topological space and O_X be a sheaf of rings on X. A *ringed space* is a pair (X, O_X). We call O_X the *structure sheaf* of X. A *locally ringed space* is a ringed space (X, O_X) such that all stalks of the structure sheaf O_X are local rings.

Let R be a commutative ring. The *spectrum* $\mathrm{Spec}(R)$ of R is the set of all prime ideals of R. An *affine scheme* is any locally ringed space isomorphic to $\mathrm{Spec}(R)$. A *scheme* is a locally ringed space X admitting a covering by open sets which are affine schemes.

Let I be an ideal of R and V_I be the set of prime ideals of R containing I. The *Zariski topology* on $\mathrm{Spec}(R)$ is the collection $\{V_I\}$ of V_I for all $I \subseteq R$. The Zariski topology on a variety is a topology with its closed sets the algebraic subsets of the variety.

2.3 Algebraic Number Theory

A complex number is an *algebraic number* if it is a root of some nonzero polynomial over $\mathbb{Q}$ (or equivalently, over $\mathbb{Z}$ if we clear out the denominators). A *number field* is a field $k = \mathbb{Q}(\alpha_1, \ldots, \alpha_n)$ for finitely many algebraic numbers α_i.

A complex number is an *algebraic integer* if it is a root of some monic polynomial over $\mathbb{Z}$. Let $\mathbb{B}$ be the set of algebraic integers. The *order* of a number field k is the ring $\mathfrak{O}_k := k \cap \mathbb{B}$, also called the *ring of integers* of k.

Let R be a commutative ring with unity. An ideal $\mathfrak{p} \subset R$ is a *prime ideal* if and only if $ab \in \mathfrak{p}$ implies $a \in \mathfrak{p}$ or $b \in \mathfrak{p}$. A *Dedekind domain* is an integral domain R of which every nonzero proper ideal $\mathfrak{a} \subset R$ factors into prime ideals $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdots \mathfrak{p}_n^{e_n}$ and the factorization is unique up to the order of factors. $\mathfrak{O}_k$ is a Dedekind domain. Hence every nonzero ideal $\mathfrak{a} \subseteq \mathfrak{O}_k$ has a unique factorization $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdots \mathfrak{p}_n^{e_n}$.

Let R be a commutative ring with unity. Two ideals $\mathfrak{a}, \mathfrak{b} \subseteq R$ are *coprime* if and only if $\mathfrak{a} + \mathfrak{b} = R$. The product of coprime ideals equals their intersection: $\mathfrak{ab} = \mathfrak{a} \cap \mathfrak{b}$.

Let $k = \mathbb{Q}$. Then $\mathfrak{O}_k = \mathbb{Z}$. Let $(p_1), \ldots, (p_n)$ be ideals of $\mathbb{Z}$ with p_i primes. By Bezout's lemma, there exists $s, t \in \mathbb{Z}$ such that $sp_i + tp_j = 1$, for $p_i \neq p_j$. I.e., $(p_i) + (p_j) = R$. (p_i) and (p_j) are coprime ideals. Hence $(p_i)(p_j) = (p_i) \cap (p_j)$. Apply this argument on all (p_i) we have: $(p_1) \cdots (p_n) = (p_1) \cap \cdots \cap (p_n) = (\mathrm{lcm}(p_1 \cdots p_n)) = (p_1 \cdots p_n)$.

2.4 Metric Theory

Let X be a set. A *metric* on X is a distance function $d : X \times X \to \mathbb{R}_{\geq 0}$ such that $d(x, y) = 0 \Leftrightarrow x = y$ (identity of indiscernibles), $d(x, y) = d(y, x)$ (symmetry), and $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality), where $\mathbb{R}_{\geq 0}$ denotes the non-negative real numbers. The pair (X, d) is called a *metric space*.

Every metric space is a topological space. A topological space is said to be *metrizable* if its topology can be describe by a metric. It is *non-metrizable* otherwise.

The Zariski topology is non-metrizable.

2.5 Measure Theory

Let X be a set. A *σ-algebra* Σ is a collection of subsets of X containing $\varnothing$ and that is closed under complement and countable unions. Note that $X \in \Sigma$ is implied since X is the complement of $\varnothing$. The pair (X, Σ) is called a *measurable space*.

Let (X, Σ) be a measurable space. A *measure* is a function $\mu : \Sigma \to [0, \infty]$ such that $\mu(\varnothing) = 0$ (null empty set), $\mu(S) \geq 0$ (non-negativity) for all $S \in \Sigma$, and $\mu \bigcup_{i=1}^{\infty} S_i = \sum_{i=1}^{\infty} \mu(S_i)$ for all countable disjoint collections $\{S_i \in \Sigma \mid i \in \mathbb{N}\}$ of sets in Σ (countable additivity). The triple (X, Σ, μ) is called a *measure space*.

The definitions of σ-algebra and topology are similar. They are connected by the *Borel σ-algebra*. It is the σ-algebra generated by the open sets of a topological space.

2.6 Representation Theory

Let V be a finite dimensional vector space. The *general linear group* $\mathrm{GL}(V) = Aut(V)$ of V is the group of automorphisms (invertible linear maps) of V. The *real general linear group* $\mathrm{GL}_n \mathbb{R}$ (i.e., the group of invertible $n \times n$ real matrices, which is an open subset of the vector space of all $n \times n$ real matrices) is the group of automorphisms of an n-dimensional real vector space V.

A *representation* of a group G on a vector space V is a group homomorphism $\rho : G \to \mathrm{GL}(V)$. I.e., ρ maps every group element to a matrix. We also call the vector space V itself a representation of G.

A *Lie group* $(G, \cdot)$ is a smooth manifold G with a smooth mapping $\cdot : G \times G \to G, a \cdot b \mapsto c$ that gives G a group structure. A *Lie subgroup* is a subgroup of a Lie group and itself is a Lie group. Let G be a Lie group. By the closed subgroup theorem, H is a Lie subgroup of G if H is a closed subgroup of G.

$\mathrm{GL}_n \mathbb{R}$ is a Lie group, with the smooth mapping $\cdot$ given by matrix multiplication: $A \cdot B = AB$. The Lie subgroups of $\mathrm{GL}_n \mathbb{R}$ are called *matrix Lie groups*.

2.7 Lattice Theory

Let $\mathbb{R}^m$ be the m-dimensional Euclidean space. A lattice in $\mathbb{R}^m$ is the set $L(B) = \{Bx : x \in \mathbb{Z}^n\}$, where $B = (b_1, \ldots, b_n) \in \mathbb{R}^{m \times n}$ is a matrix with linear independent columns. We call $\{b_1, \ldots, b_n\}$ the *lattice basis*. The integers n and m are called the

rank and *dimension* of the lattice, denoted $\text{rank}(L(B))$ and $\dim(L(B))$, respectively. If $n = m$ the lattice is called a *full rank* lattice. The determinant of a lattice is the n-dimensional volume of the fundamental parallelepiped spanned by the basis vectors, denoted $\det(L(B))$. It equals the square root of the determinant of the Gram matrix $B^\top B$, i.e., $\det(L(B)) = \sqrt{\det(B^\top B)}$. If $L(B)$ is full rank, then $\det(L(B)) = |\det(B)|$. The Gaussian heuristic for the shortest vector in a random full rank lattice $L(B) \in \mathbb{R}^m$ is $\lambda_1 = \sqrt{\frac{m}{2\pi e}}\,\text{vol}(L)^{\frac{1}{m}}$.

The *dual* of a lattice L is the set $\hat{L}$ of vectors $x \in \text{span}(L)$ such that the inner product $x \cdot y \in \mathbb{Z}$ for all $y \in L$.

The closest vector problem (CVP) is given a lattice basis $B \in \mathbb{Z}^{m \times n}$ and a target vector $t \in \mathbb{Z}^m$, find a lattice vector $v \in L(B)$ closest to t.

The γ-gap closest vector problem (GapCVP_γ) is given an instance (B, t, r), where $B \in \mathbb{Z}^{m \times n}$ is a lattice basis, $t \in \mathbb{Z}^m$ is a vector and $r \in \mathbb{Q}$ is a rational number, decide whether it is a YES instance or a NO instance, where YES instances are triples (B, t, r) such that $||Bz - t|| \leq r$ for some $z \in \mathbb{Z}^n$, and NO instances are triples (B, t, r) such that $||Bz - t|| > \gamma r$ for all $z \in \mathbb{Z}^n$.

The γ-bounded distance decoding (BDD_γ) problem is given a lattice basis B and a vector t such that $\text{dist}(B, t) < \gamma\lambda_1(B)$, find the lattice vector $v \in L(B)$ closest to t.

The γ-unique shortest vector problem (uSVP_γ) is given a lattice B such that $\lambda_2(B) > \gamma\lambda_1(B)$, find a nonzero vector $v \in L(B)$ of length $\lambda_1(B)$.

2.8 Function Obfuscation

Computational functions can be represented by circuits. 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*. They are also known as dependent and independent auxiliary information, respectively.

Evasive Functions Evasive functions are the kind of Boolean functions that have small fiber of 1 compared to the domain of the function.

Definition 1 (Evasive Circuit Collection [BBCKPS14]). *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 negligi-*

ble 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.

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

Definition 2 (Input-Hiding [BBCKPS14]). *Let $\lambda \in \mathbb{N}$ be the security parameter. Let $\mathcal{C} = \{C_\lambda\}_{\lambda \in \mathbb{N}}$ be a circuit collection. Let $\mathcal{D}$ be a class of distribution ensembles $D = \{D_\lambda\}_{\lambda \in \mathbb{N}}$, where D_λ is a distribution over C_λ for all $\lambda \in \mathbb{N}$. 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 properties are satisfied.*

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

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

and

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

where the probability is over the coin tosses of O.

2. Polynomial Slowdown: There exists a polynomial p such that for all n, all circuits $C \in \mathcal{C}$, and all possible sequences of coin tosses of O, 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 all PPT adversaries $\mathcal{A}$, for all $\lambda \in \mathbb{N}$ and for all global auxiliary information $\alpha \in \{0,1\}^{poly(\lambda)}$ about C_λ,

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

where the probability is taken over the random sampling of D_λ and the coin tosses

of $\mathcal{A}$ and O.

Note that input-hiding is particularly defined for evasive functions. This is because non-evasive functions always leak accepting inputs. Also note that input-hiding is incomparable with VBB [BBCKPS14]. An obfuscator which always leaks an accepting input of the function can still be VBB [BBCKPS14].

Virtual Black-Box Obfuscation The intuition of VBB obfuscation is that anything one can efficiently compute from the obfuscated function, one should be able to efficiently compute given just black-box access to the function [BGIRSVY01]. It attempts to hide everything about a circuit without affecting the usage of its functionalities.

Definition 3 (Average-Case Virtual Black-Box Obfuscator, VBB [BGIRSVY01]). *Let $\lambda \in \mathbb{N}$ be the security parameter. Let $\mathcal{C} = \{C_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of polynomial size circuits. Let $\mathcal{D}$ be a class of distribution ensembles $D = \{D_\lambda\}_{\lambda\in\mathbb{N}}$, where D_λ is a distribution over C_λ for all $\lambda \in \mathbb{N}$. A PPT algorithm O is a VBB obfuscator for the family $\mathcal{C}$ and the distribution $\mathcal{D}$ if it satisfies the functionality preserving and polynomial slowdown properties in Definition 2 and the following virtual black-box property: There exists a negligible function $\mu(\lambda)$ such that for every PPT adversary $\mathcal{A}$, there exists a PPT simulator $\mathcal{S}$ such that for all polynomial size predicates $\varphi : C \to \{0,1\}$ and all global auxiliary information $\alpha \in \{0,1\}^{poly(\lambda)}$ about C_λ,*

$$\left|\Pr_{C\leftarrow D_\lambda}[\mathcal{A}(O(C),\alpha) = \varphi(C)] - \Pr_{C\leftarrow D_\lambda}[\mathcal{S}^C(1^\lambda,\pi,\alpha) = \varphi(C)]\right| \leq \mu(\lambda) \tag{2.1}$$

where the first probability is taken over the random sampling of D_λ and the coin tosses of $\mathcal{A}$ and O, and the second probability is taken over the random sampling of D_λ and the coin tosses of $\mathcal{S}$; π is a set of parameters associated to C (e.g., input size, output size, circuit size, etc.) which we are not required to hide, and $\mathcal{S}^C$ means that $\mathcal{S}$ has black-box access to the circuit C.

The following variant of VBB is used in Chapter 4. The difference is that in the definition of VBB, the auxiliary information α is about the entire circuit family $\mathcal{C}$, while in the following definition, α is about the specific circuit C being obfuscated.

Definition 4 (Distributional Virtual Black-Box Obfuscator, DVBB [WZ17]). *Let $\lambda \in \mathbb{N}$ be the security parameter. Let $\mathcal{C} = \{C_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of polynomial size circuits. Let $\mathcal{D}$ be a class of distribution ensembles $D = \{D_\lambda\}_{\lambda\in\mathbb{N}}$, where D_λ*

is a distribution over C_λ and polynomial size auxiliary information α, for all $\lambda \in \mathbb{N}$. A PPT algorithm O is a DVBB obfuscator for the distribution class $\mathcal{D}$ over the circuit family $\mathcal{C}$ if it satisfies the functionality preserving and polynomial slowdown properties in Definition 2 and the following distributional virtual black-box property: There exists a negligible function $\mu(\lambda)$ such that for every polynomial size adversary $\mathcal{A}$, there exists a PPT simulator $\mathcal{S}$ such that for all distribution ensemble $D = \{D_\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{D}$, and all polynomial size predicates $\varphi : \mathcal{C} \to \{0,1\}$,

$$\left| \Pr_{(C,\alpha)\leftarrow D_\lambda}[\mathcal{A}(O(C),\alpha) = \varphi(C)] - \Pr_{(C,\alpha)\leftarrow D_\lambda}[\mathcal{S}^C(1^\lambda,\pi,\alpha) = \varphi(C)] \right| \leq \mu(\lambda), \tag{2.2}$$

where the first probability is taken over the coin tosses of $\mathcal{A}$ and O, the second probability is taken over the coin tosses of $\mathcal{S}$; π is a set of parameters associated to C (e.g., input size, output size, circuit size, etc.) which we are not required to hide, and $\mathcal{S}$ has black-box access to the circuit C.

Note that black-box access to evasive functions is useless. Hence it makes sense to consider a definition that does not give the simulator black-box access to the circuit C.

Definition 5 (Distributional-Indistinguishability [WZ17]). *A PPT algorithm O for the distribution class $\mathcal{D}$ over a family of circuits $\mathcal{C}$, satisfies distributional-indistinguishability, if there exists a PPT simulator $\mathcal{S}$, such that for every distribution ensemble $D = \{D_\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{D}$, where D_λ is a distribution over $C_\lambda \times \{0,1\}^{poly(\lambda)}$ for all $\lambda \in \mathbb{N}$, we have that*

$$(O(1^\lambda, C), \alpha) \overset{c}{\approx} (\mathcal{S}(1^\lambda,\pi),\alpha),$$

where $(C,\alpha) \leftarrow D_\lambda$, and α is some auxiliary information. I.e., there exists a negligible function $\mu(\lambda)$ such that for all PPT algorithm $\mathcal{A}$,

$$\left| \Pr_{(C,\alpha)\leftarrow D_\lambda}[\mathcal{A}(O(1^\lambda,C),\alpha) = 1] - \Pr_{(C,\alpha)\leftarrow D_\lambda}[\mathcal{A}(\mathcal{S}(1^\lambda,\pi),\alpha) = 1] \right| \leq \mu(\lambda), \tag{2.3}$$

where the first probability is taken over the coin tosses of $\mathcal{A}$ and O, the second probability is taken over the coin tosses of $\mathcal{A}$ and $\mathcal{S}$.

Distributional-indistinguishability with auxiliary information $\alpha' = (\alpha, \varphi(C))$ implies DVBB with auxiliary information α [WZ17], where $\varphi(C)$ is an arbitrary 1-bit predicate of the circuit. To state the theorem, we need the following definition of *predicate augmentation*, which allows to add an arbitrary 1-bit predicate of the circuit to the

auxiliary information.

Definition 6 (Predicate Augmentation [BGIRSVY01; WZ17]). *For a distribution class $\mathcal{D}$, we define its augmentation under predicates, denoted $aug(\mathcal{D})$, as follows. For any polynomial-time predicate $\varphi : \{0,1\}^* \to \{0,1\}$ and any $D = \{D_\lambda\} \in \mathcal{D}$ the class $aug(\mathcal{D})$ indicates the distribution ensemble $D' = \{D'_\lambda\}$ where D'_λ samples $(C, \alpha) \leftarrow D_\lambda$, computes $\alpha' = (\alpha, \varphi(C))$ and outputs (C, α').*

Theorem 1 (Distributional-Indistinguishability implies DVBB [WZ17]). *For any family of circuits $\mathcal{C}$ and a distribution class $\mathcal{D}$ over $\mathcal{C}$, if an obfuscator O satisfies distributional-indistinguishability for the class of distributions $\mathcal{D}' = aug(\mathcal{D})$, i.e., if there exists a negligible function $\mu(\lambda)$, a PPT simulator $\mathcal{S}$, such that for every PPT distinguisher $\mathcal{B}$, for every distribution ensemble $D' = \{D'_\lambda\}$ where D'_λ samples $(C, \alpha) \leftarrow D_\lambda$ with $C \in \mathcal{C}$, computes $\alpha' = (\alpha, \varphi(C))$ and outputs (C, α'),*

$$\left| \Pr_{(C,\alpha') \leftarrow D'_\lambda} [\mathcal{A}(O(1^\lambda, C), \alpha') = 1] - \Pr_{(C,\alpha') \leftarrow D'_\lambda} [\mathcal{A}(\mathcal{S}(1^\lambda, \pi), \alpha') = 1] \right| \leq \mu(\lambda), \tag{2.4}$$

then it also satisfies DVBB security for the distribution class $\mathcal{D}$.

Note that the auxiliary informations α in input-hiding and VBB are global information for the entire function family, while the α in DVBB and distributional-indistinguishability are local information about the specific function being obfuscated.

Chapter 3

SUBSET PRODUCT WITH ERRORS

We first reduce the discrete logarithm problem (DLP) to the subset product problem (SP) [GZ19] with density $d \geq n/(n + \log_2 p(n))$ for some polynomial $p(n)$. The twin subset product problem (TSP) with respect to twin points is then defined. The multiple subset product problem (MSP) with respect to relative points is defined. A lattice algorithm is presented to solve low density MSP with identical points. To avoid the MSP algorithm, the subset product with errors problem (SPE) is defined. SP is reduced to SPE. The multiple subset product with errors problem (MSPE) is defined. The ideal subset product with errors problem (ISPE) is defined. Both SPE and ISPE, as well as their twin variants and multi-instance variants are conjectured to be post-quantum hard.

3.1 Subset Product

In this chapter, all parameters are functions in λ with $\lambda \leq n \in \mathbb{N}$. The subset product problem is the following.

Definition 7 (Subset Product Problem, SP [GZ19]). *Given $n+1$ distinct primes $p_1, \ldots, p_n, q$ and an integer $X \in \mathbb{Z}_q^*$, find a vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ (if it exists) such that $X = \prod_{i=1}^n p_i^{x_i} \pmod q$.*

The decisional version is the following.

Definition 8 (Decisional Subset Product Problem, d-SP [GZ19]). *Given $n+1$ distinct primes $p_1, \ldots, p_n, q$ and an integer $X \in \mathbb{Z}_q^*$, decide if there exists a vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ such that $X = \prod_{i=1}^n p_i^{x_i} \pmod q$.*

In order to define hard SP, we avoid parameters that will make the problem trivial.

If $q \geq \prod_{i=1}^n p_i^{x_i}$, then x_i is immediately leaked by checking whether $p_i | X$. Hence we require $q < \prod_{i=1}^n p_i^{x_i}$. In particular, we can set q to lie between a length r prime product and a length $r+1$ prime product, for some suitably chosen $r < n$.

Now if $r \geq n/2$, the problem is still trivial. One can just sample a uniform $y \in \{0,1\}^n$ and decode x from $XY^{-1} \pmod q$ using the naive or improved attack based on Theorem

3 or 4, where $Y = \prod_{i=1}^{n} p_i^{y_i} \pmod q$. The naive attack works when the Hamming distance between x and y is $\leq r$. Note that a uniform y is expected to be $n/2$ away from x. Hence if $r \geq n/2$, the naive attack is expected to work. To avoid this, we require negligible probability of y being r-close to x. I.e., $\Pr_{y \leftarrow \{0,1\}^n}[|x \oplus y| \leq r] \leq \mu(\lambda)$ for some negligible function $\mu(\lambda)$, where $\oplus$ denotes the XOR operation. If we take $\mu(\lambda) = 1/2^{\lambda}$, then for uniformly sampled x and y, the above inequality gives

$$r \leq \frac{n}{2} - \sqrt{\lambda n \ln 2}. \tag{3.1}$$

For a proof of this, see Lemma 2 in [GZ19].

Again, if x is from a low entropy distribution, finding a point y close to x is easy. For example, suppose all points cluster together. Then one can just find y by searching in the cluster. To avoid this, we require the distribution of x to have conditional Hamming ball min-entropy λ, which is defined as follows.

Definition 9 (Conditional Hamming Ball Min-Entropy [GZ19; FRS16]). *The Hamming ball min-entropy of random variables X conditioned on a correlated variable Y is*

$$H_{Ham,\infty}(X \mid Y) = -\ln\left(\mathbb{E}_{y \leftarrow Y}[\max_{y \in \{0,1\}^n} \Pr[|X \oplus y| \leq r \mid Y = y]]\right),$$

where $r < n \in \mathbb{N}$.

Also note that a composite modulus will leak Legendre symbol relations about x with respect to different factors of the modulus. We therefore use prime modulus.

Now we are ready to define the hard SP distribution.

Definition 10 ((n, r, B, X_n)-SP Distribution). *Let λ, n, r, B be positive integers with $n \geq \lambda$ polynomial in λ, r satisfies Inequality (3.1), and B larger than the n-th prime. Let X_n be a distribution over $\{0,1\}^n$ with Hamming ball min-entropy λ. Let $(x_1 \ldots, x_n) \leftarrow X_n$. Let $p_1, \ldots, p_n$ be distinct primes uniformly sampled from the primes in $\{2, \ldots, B\}$. Let q be a uniformly sampled safe prime in $\{B^r, \ldots, (1 + o(1))B^r\}$. Then we call the distribution $(p_1, \ldots, p_n, q, X)$ with $X = \prod_{i=1}^{n} p_i^{x_i} \pmod q$ the (n, r, B, X_n)-SP distribution.*

The hard SP and hard d-SP are the following.

Assumption 1 (Hard SP). *Let λ, n, r, B, X_n satisfy the conditions in Definition 10. Then for every PPT algorithm $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, there exists a negligible function $\mu(\lambda)$ such that the probability that $\mathcal{A}$ solves SP of instances sampled from the (n, r, B, X_n)-SP distribution is not greater than $\mu(\lambda)$.*

Assumption 2 (Hard d-SP). *Let λ, n, r, B, X_n satisfy the conditions in Definition 10. Let $D_0 = (p_1, \ldots, p_n, q, X)$ be the (n, r, B, X_n)-SP distribution and let D_1 be D_0 with $X = \prod_{i=1}^{n} p_i^{x_i} \pmod q$ replaced by a random element in $\mathbb{Z}_q^*$. Then for every PPT algorithm $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, there exists a negligible function $\mu(\lambda)$ such that*

$$\left| \Pr_{d_0 \leftarrow D_0}[\mathcal{A}(d_0) = 1] - \Pr_{d_1 \leftarrow D_1}[\mathcal{A}(d_1) = 1] \right| \leq \mu(\lambda). \tag{3.2}$$

By the search-to-decision reductions in [IN96] and [MM11], d-SP is plausibly at least as hard as SP. In the following we show that SP is at least as hard as DLP for certain parameter ranges. Let $d := n/\log_2 q$ be the density of SP. An informal statement of this result for SP with density $d \geq 1$ was given in [GZ19]. In the following we give a rigorous proof for a better lower bound of the density: $d \geq n/(n + \log_2 p(n))$.

Definition 11 (Discrete Logarithm Problem, DLP). *Let G be a finite group of order N written in multiplicative notation. The discrete logarithm problem is given $g, h \in G$ to find a (if it exists) such that $h = g^a$.*

Assumption 3 (Hard DLP). *Let $\mathbb{Z}_q^*$ be the multiplicative group of integers modulo q, where $q = 2p + 1 \geq 2^\lambda$ is a safe prime for some prime p. If g is sampled uniformly from $\mathbb{Z}_q^*$ and a is sampled uniformly from $\{0, \ldots, q-2\}$, then for every PPT algorithm $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, there exists a negligible function $\mu(\lambda)$ such that the probability that $\mathcal{A}$ solves the DLP (g, g^a) is not greater than $\mu(\lambda)$.*

Two heuristics are needed for the reduction.

Heuristic 1. *The number of elements $X \in \mathbb{Z}_q^*$ being a subset product $\prod_{i=1}^{n} p_i^{x_i} \pmod q$ over the (n, r, B, U_n)-SP distribution $(p_1, \ldots, p_n, q, X)$ with $q \leq 2^n p(n)$ is $\geq q/p(n)$, for polynomial $p(n)$, where U_n is the uniform distribution.*

This means that if q is not larger than polynomial times 2^n, then a uniformly chosen X from $\mathbb{Z}_q^n$ is a subset product with noticeable probability. Also notice that the requirement $q \leq 2^n p(n)$ implies the density $d \geq n/(n + \log_2 p(n))$.

Heuristic 2. *The number of random DLP group elements g^a needed for getting polynomially many SP solutions x that span $\mathbb{Z}_\ell^n$ for every prime factor ℓ of $q - 1$ is polynomial, where $x \in \{0,1\}^n$ is such that $g^a = \prod_{i=1}^{n} p_i^{x_i} \pmod q$ and SP and DLP are as defined in Assumption 1 and 3.*

This means that when writing different DLP group elements g^a in terms of subset products $\prod_{i=1}^n p_i^{x_i} \pmod q$ with respect to some random primes $p_1, \ldots, p_n$, the exponent vectors $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ have high probability to give a full rank matrix over $\mathbb{Z}_\ell$, for every prime factor ℓ of $q-1$. This makes sense if we think about the randomness of the primes $p_1, \ldots, p_n$. Also note that q is a safe prime such that $q - 1 = 2p$ has only two prime factors 2 and p. Hence this is not a serious requirement because there are only two subspaces $\mathbb{Z}_2^n$ and $\mathbb{Z}_p^n$ needed to be satisfied.

THEOREM 2. *Assuming Heuristic 1 and 2, if there exists a PPT algorithm to solve SP (as defined in Assumption 1, with $q \leq 2^n p(n)$) with overwhelming probability in time T, then there exists an algorithm to solve DLP (as defined in Assumption 3) in expected time $O(t(\lambda)T)$, for some polynomial $t(\lambda)$.*

Proof. Let (g, h) be a DLP instance as defined in Assumption 3. Let $\mathcal{A}$ be a PPT algorithm that solves SP as defined in Assumption 1, with the same q as the DLP. We solve the DLP as follows. Sample a uniform a from $\{0, \ldots, q-2\}$, then call $\mathcal{A}$ to solve $(p_1, \ldots, p_n, q, g^a)$. If g^a is a subset product, then with overwhelming probability $\mathcal{A}$ can solve for an $x \in \{0,1\}^n$ such that $g^a = \prod_{i=1}^n p_i^{x_i} \pmod q$. Since $q \leq 2^n p(n)$, by Heuristic 1 we have that $n_{\text{SP}} \geq q/p(n)$, where n_{SP} is the number of subset products in $\mathbb{Z}_q^*$. Hence the probability that g^a being a subset product is $\geq 1/p(n)$. We therefore expect that after $np(n)$ samples of a, we can solve for n vectors $x \in \{0,1\}^n$ such that $a \equiv \sum_{i=1}^n x_i \log_g(p_i) \pmod{q-1}$.

Also by Heuristic 2, with at most $np(n)p'(n)$ samples of a, we expect to be able to choose n vectors $x \in \{0,1\}^n$ to span $\mathbb{Z}_\ell^n$ for each prime factor ℓ of $q-1$, for some polynomial $p'(n)$. We therefore have n relations $a \equiv \sum_{i=1}^n x_i \log_g(p_i) \pmod \ell$ whose coefficient matrix is full rank, for each prime factor ℓ of $q-1$. Then we can solve the systems of equations for different ℓ respectively and use the Chinese remainder theorem to lift the solutions to $\mathbb{Z}_{q-1}$, obtaining $\log_g(p_i) \pmod{q-1}$ for all $i \in \{1, \ldots, n\}$.

Lastly we sample $b \leftarrow \{0, \ldots, q-2\}$, compute $hg^b \pmod q$, and call $\mathcal{A}$ to solve it. With at most $p(n)$ extra samples of b, we expect one more relation $\log_g(h) + b \equiv \sum_{i=1}^n x_i \log_g(p_i) \pmod{q-1}$ with $x \in \{0,1\}^n$. Then $\log_g(h) = \sum_{i=1}^n x_i \log_g(p_i) - b \pmod{q-1}$. □

Attacks

We discuss two potential attacks for SP. They are inefficient. The first attack relies on finding a point y close to x. The second attack improves the first attack by allowing

the use of points arbitrary far away from x. This attack adds exponential overhead hence is still inefficient.

THEOREM 3 (Diophantine Approximation [Hur91]). *Let $\alpha \in \mathbb{R}$, $a, b \in \mathbb{Z}$, $b > 0$ and $\gcd(a, b) = 1$. If $|\alpha - \frac{a}{b}| < \frac{1}{2b^2}$ then a/b is a convergent of the continued fraction of α.*

The attack based on Theorem 3 is as follows. Having an input y such that the Hamming distance between x and y is bounded by r, we compute $E = XY^{-1} \pmod q = \prod_i p_i^{x_i - y_i} \pmod q = UV^{-1} \pmod q$, where UV^{-1} is the simplest form of XY^{-1} modulo q with $U = \prod_{i=1}^{n} p_i^{u_i}$ and $V = \prod_{i=1}^{n} p_i^{v_i}$, for $u_i, v_i \in \{0, 1\}$. We have that $EV - kq = U$ hence $|\frac{E}{q} - \frac{k}{V}| = \frac{U}{qV}$, where $\gcd(k, V) = 1$ since $\gcd(U, V) = 1$. By Theorem 3, if $UV < \frac{q}{2}$, then $\frac{k}{V}$ is a convergent of $\frac{E}{q}$. Finding this convergent from the continued fraction of $\frac{E}{q}$ is efficient because the set of convergents is of polynomial size. So we have k and V, and thus $U = EV - kq$. We then factor U and V to find all different bits between x and y, and recover x by flipping y accordingly.

The following theorem shows a way to push the continued fraction algorithm beyond the naive limits given by Theorem 3.

THEOREM 4 (Extended Legendre Theorem [Duj04]). *Let α be an irrational number, let the fractions $\frac{p_i}{q_i} \in \mathbb{Q}$ be its continued fraction, and let a, b be coprime nonzero integers satisfying the inequality $|\alpha - \frac{a}{b}| < \frac{c}{b^2}$, where c is a positive real number. Then $(a, b) = (rp_{m+1} \pm sp_m, rq_{m+1} \pm sq_m)$, for some nonnegative integers m, r and s such that $rs < 2c$.*

By Theorem 4 one can always find a and b by tuning c, which gets rid of the limitation of $|\alpha - \frac{a}{b}| < \frac{1}{2b^2}$. But this adds exponential overhead. Specifically, we are now allowed to recover x using any string $y \in \{0, 1\}^n$ that is arbitrary far away from x. However the search space for $\frac{k}{V}$ is now not the set of convergents of $\frac{E}{q}$, but a set of combinations of nominators and denominators of the convergents, where the coefficients are all possible pairs (r, s) such that $rs < 2c$. This c is determined by the distance between x and y. So if x and y are far away from each other, e.g. $n/2$ away (for a random $y \in \{0, 1\}^n$), then the search space will be exponentially large and the current searching methods (e.g., the one in [Duj09]) takes exponential time. For a more concrete example, suppose $n \approx 6\lambda$, $q \approx (n \ln n)^r$ and $r \approx n/3$. Then for a point y which is $n/2$ away from x, we have $UV \approx (n \ln n)^{n/2}$. To use Theorem 4, we have $c > \frac{UV}{q} \approx (n \ln n)^{n/6} = (n \ln n)^{\lambda}$. Then the number of combinations of $r, s < c$ is exponential.

3.2 Twin Subset Product

Imagine we are given more than one SP instance with respect to the same x. Then the task of finding x should be easier since we have more clues about x. Nonetheless, we conjecture that the problem is still hard when the number of instances is just 2. One step further, we do not require the x's to be exactly the same but close.

We call a pair of points $x, y \in \{0,1\}^n$ *twin points*, denoted by $x \approx y$, if there exists a polynomial $p(\lambda)$ such that the Hamming distance d between x and y satisfies $\sum_{i=0}^{d} \binom{n}{i} \leq p(\lambda)$. This simply means x and y are in polynomial size Hamming balls centered at each other. Namely it is easy to find one from the other.

Definition 12 (Twin Subset Product Problem, TSP). *Given two independent SP instances with respect to a pair of twin points, find the twin points.*

Assumption 4 (Hard TSP). *Let X_n be a distribution over twin points $(x, y) \in \{0,1\}^n \times \{0,1\}^n$. Let $((p_1, \ldots, p_n, q, X), (s_1, \ldots, s_n, t, Y))$ be a TSP instance with each SP defined as in Assumption 1, where $X = \prod p_i^{x_i} \pmod q$ and $Y = \prod s_i^{y_i} \pmod t$ with $(x, y) \leftarrow X_n$. Then for every PPT algorithm $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, there exists a negligible function $\mu(\lambda)$ such that the probability that $\mathcal{A}$ solves this TSP is not greater than $\mu(\lambda)$.*

Note that if $q = t$ and $p_i = s_i$ for all $i \in \{1, \ldots, n\}$, the distinct bits between x and y can be learned. To see this, we simply divide X by Y. We expect that many primes will be canceled since x and y are close. What is left is $XY^{-1} \pmod q = \prod_{i=1}^{n} p_i^{x_i - y_i} \pmod q = UV^{-1} \pmod q$ for UV small, where UV^{-1} is the simplest form of XY^{-1} modulo q. Then we can use the attacks based on Theorem 3 or 4 to find the factors of U and V. From the factors of U and V we can tell the distinct bits between x and y. Specifically, if p_i is a factor of U then $x_i = 1$ and $y_i = 0$; if p_i is a factor of V then $x_i = 0$ and $y_i = 1$. Otherwise if p_i is neither a factor of U or V, then $x_i = y_i$.

However this attack does not reveal the common bits at all. Leaking the distinct bits between x and y does not make the problem much easier.

On the other hand, this setting should be avoided when defining the decisional version of the problem. Using the above attack one can identify the TSP distribution by checking whether the two SP instances are with respect to two points that are close to each other.

With a DLP algorithm, TSP reduces to the twin subset sum problem.

Definition 13 (Subset Sum Problem, SS). *Let $\mathbb{Z}_N$ be the additive group modulo $N \in \mathbb{N}$. The subset sum problem is given n numbers $a_1, \ldots, a_n \in \mathbb{Z}_N$ and an integer $X \in \mathbb{Z}_N$, find $x \in \{0,1\}^n$ (if it exists) such that $X = \sum_{i=1}^n a_i x_i \pmod N$.*

Definition 14 (Twin Subset Sum Problem, TSS). *Given two independent SS instances with respect to a pair of twin points, find the twin points.*

To see the reduction from TSP to TSS, simply notice the reduction from SP to SS. Specifically, for an SP we have $X = \prod_{i=1}^n p_i^{(x_i)} \pmod q = \prod_{i=1}^n (g^{a_i})^{x_i} \pmod q = g^{\sum_{i=1}^n x_i a_i \pmod{q-1}} = g^{a \pmod{q-1}}$ for some generator g of $\mathbb{Z}_q^*$. Hence by choosing a random generator g and call the DLP algorithm to solve for the exponents a_i of p_i for all $i \in \{1, \ldots, n\}$ as well as the exponent a of X, we have a subset sum instance $(a_1, \ldots, a_n, q-1, a)$ with $a = \sum_{i=1}^n x_i a_i \pmod{q-1}$.

3.3 Multiple Subset Product

Is the problem still hard if the number of SP instances is more than 2? Similar to TSP, we define the multiple subset product problem to be with respect to different close points. The problem with an identical point is a special case.

Let $V = \{x_1, \ldots, x_k \mid x_i \in \{0,1\}^n, i \in \{1, \ldots, k\}\}$ be a set (polynomial size) of points. Let $E = \{\{x_i, x_j\} \mid x_i \approx x_j, i, j \in \{1, \ldots, k\}, i \neq j\}$ be a set of unordered pairs given by the twin points relations. We call the points $x_1, \ldots, x_k$ *relative points* if the graph $G = (V, E)$ is connected. This simply means finding one point is enough to brute force all others.

A special case of relative points is pair-wise twin points, namely all of the points $x_1, \ldots, x_k$ are inside a polynomial size Hamming ball.

Definition 15 (Multiple Subset Product Problem, MSP). *Given $k > 2$ SP instances with respect to k relative points, find the k points.*

Note that if we encode distinct relative points using the same prime vector $p^{(1)} = \cdots = p^{(k)}$ and the same modulus $q_1 = \cdots = q_k$, then MSP is solvable if the relative points have not many common entries. The attack is to learn distinct bits between more and more pairs of relative points using the discussed attack for TSP, and gradually recover more and more entries of a point.

With a DLP algorithm, MSP reduces to the multiple subset sum problem.

Definition 16 (Multiple Subset Sum Problem, MSS). *Given $k > 2$ independent SS instances with respect to k relative points, find the k points.*

The decision versions of MSP and MSS are the following.

Definition 17 (Decisional Multiple Subset Product Problem, d-MSP). *Let P and Q be two distributions over two sets of primes respectively. The decisional multiple subset product problem is to distinguish the distribution*

$$D_1 = (p^{(1)}, \ldots, p^{(k)}, q_1, \ldots, q_k, X_1, \ldots, X_k)$$

with $p^{(j)} \leftarrow P^n$, $q_j \leftarrow Q$, $x^{(j)} \in \{0,1\}^n$ relative points, $X_j = \prod_{i=1}^{n} (p_i^{(j)})^{x_i^{(j)}} \pmod{q_j}$ and $j \in \{1, \ldots, k\}$, from the distribution

$$D_2 = (p^{(1)}, \ldots, p^{(k)}, q_1, \ldots, q_k, X_1, \ldots, X_k)$$

with $p^{(j)} \leftarrow P^n$, $q_j \leftarrow Q$, $X_j \leftarrow \mathbb{Z}_{q_j}$ and $j \in \{1, \ldots, k\}$.

Definition 18 (Decisional Multiple Subset Sum Problem, d-MSS). *Let X_n be a distribution over relative points $x^{(1)}, \ldots, x^{(k)} \in \{0,1\}^n$. Let A_j be a distribution over $\mathbb{Z}_j^n$, for $j \in \mathbb{N}$. Let N be a distribution over a set of integers. The decisional multiple subset sum problem is to distinguish the distribution*

$$D_1 = (a^{(1)}, \ldots, a^{(k)}, N_1, \ldots, N_k, X_1, \ldots, X_k)$$

with $(x^{(1)}, \ldots, x^{(k)}) \leftarrow X_n$, $N_j \leftarrow N$, $a^{(j)} \leftarrow A_{N_j}$, $X_j = \sum_{i=1}^{n} x_i^{(j)} \cdot a_i^{(j)} \pmod{N_j}$ and $j \in \{1, \ldots, k\}$, from the distribution

$$D_2 = (a^{(1)}, \ldots, a^{(k)}, N_1, \cdots, N_k, X_1, \ldots, X_k)$$

with $N_j \leftarrow N$, $a^{(j)} \leftarrow A_{N_j}$, $X_j \leftarrow A_{N_j}$ and $j \in \{1, \ldots, k\}$.

MSP is not conjectured to be hard. Assuming a DLP algorithm, we give a lattice attack for low density MSP with identical points $x^{(1)} = \cdots = x^{(k)}$. We first reduce low density MSP to a CVP, then reduce the CVP from even lower density MSP to an LLL solvable γ-uSVP.

Reducing MSP to CVP

HEURISTIC 3. *In the low density case ($q \gg 2^n$), every subset product $X = \prod_{i=1}^{n} p_i^{x_i} \pmod q$ has a unique solution $(x_1, \ldots, x_n)$ with $x_i \in \{0, 1\}$.*

In the following we prove it for $q > (\sqrt{2\pi e})^n + 1$. The proof assumes that the lattices L in the proof satisfy the Gaussian heuristic. Experimental results for the precision of the Gaussian heuristic for these lattices are given in Section 4.2.

LEMMA 1. *Assuming the Gaussian heuristic, if $q > (\sqrt{2\pi e})^n + 1$ and $\{p_1, \ldots, p_n\}$ generates $\mathbb{Z}_q^*$, then every subset product $X = \prod_{i=1}^{n} p_i^{x_i} \pmod q$ has a unique solution $(x_1, \ldots, x_n)$ with $x_i \in \{0, 1\}$.*

Proof. Consider the lattice

$$L = \left\{ z \in \mathbb{Z}^n \;\middle|\; \prod_{i=1}^{n} p_i^{z_i} = 1 \pmod q \right\}$$

given by the kernel of the group morphism

$$\phi : \mathbb{Z}^n \to \mathbb{Z}_q^*,$$

$$(x_1, \ldots, x_n) \mapsto \prod_{i=1}^{n} p_i^{x_i} \pmod q.$$

Let λ_1 be the length of the shortest vector of L. We show by contradiction that if there exist two solutions $x \neq y \in \{0, 1\}^n$ to the SP, then there exists a vector in L having length $< \lambda_1$.

We first derive λ_1. By the Gaussian heuristic, it is

$$\lambda_1 \sim \sqrt{\frac{n}{2\pi e}} \operatorname{vol}(L)^{\frac{1}{n}},$$

where $\operatorname{vol}(L)$ is the volume of L.

By the first isomorphism theorem, the volume of the lattice $\operatorname{vol}(L)$ is given by the size of the image $|\operatorname{im} \phi|$ of ϕ. Hence

$$\operatorname{vol}(L) \leq \varphi(q) = q - 1,$$

where φ is the Euler totient function. The equality holds if and only if $\{p_1, \ldots, p_n\}$

generates $\mathbb{Z}_q^*$. So

$$\lambda_1 \sim \sqrt{\frac{n}{2\pi e}}\,\mathrm{vol}(L)^{\frac{1}{n}} = \sqrt{\frac{n}{2\pi e}}(q-1)^{\frac{1}{n}}.$$

Hence if we take $\lambda_1 = \sqrt{\frac{n}{2\pi e}}(q-1)^{\frac{1}{n}}$ and plug in $q > (\sqrt{2\pi e})^n + 1$, we have $\lambda_1 > \sqrt{n}$.

Now, suppose there exist two SP solutions $x \neq y \in \{0,1\}^n$ such that $\prod_{i=1}^n p_i^{x_i} = \prod_{i=1}^n p_i^{y_i} \pmod q$. Then $\prod_{i=1}^n p_i^{x_i - y_i} = 1 \pmod q$ with $z = x - y \in \{-1, 0, 1\}^n$ and $z \neq 0$. This means a nonzero vector $z \in L$ of length $\leq \sqrt{n}$, contradicting $\lambda_1 > \sqrt{n}$. Hence the SP solution is unique. □

LEMMA 2. *Assuming the Gaussian heuristic and a DLP algorithm, if a d-MSP (with identical relative points) has at least one $j \in \{1, \ldots, k\}$ such that $q_j > (\sqrt{2\pi e})^n + 1$ and that $\{p_1^{(j)}, \ldots, p_n^{(j)}\}$ generates $\mathbb{Z}_{q_j}^*$, then it reduces to GapCVP$_1$.*

Proof. We show that if there exists a PPT distinguisher $\mathcal{A}$ for GapCVP$_1$, then there exists a PPT distinguisher $\mathcal{B}$ for low density d-MSP. Given a low density d-MSP instance p, $\mathcal{B}$ works as follows. It first reduces p to a d-MSS instance $s := (a^{(1)}, \ldots, a^{(k)}, N_1, \ldots, N_k, X_1, \ldots, X_k)$ by solving DLP, where $N_j = q_j - 1$ for $j \in \{1, \ldots, k\}$. It then creates a GapCVP$_1$ instance $c := (B, t, r)$, where

$$B = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ a_1^{(1)} & \ldots & a_n^{(1)} & N_1 & & \\ \vdots & \vdots & \vdots & & \ddots & \\ a_1^{(k)} & \ldots & a_n^{(k)} & & & N_k \end{bmatrix} = (b_1, \ldots, b_{n+k}), \tag{3.3}$$

is a lattice basis,

$$t = (\frac{1}{2}, ..., \frac{1}{2}, X_1, ..., X_k)^\top \tag{3.4}$$

with $X_j \in \mathbb{Z}_{N_j}$ for all $j \in \{1, \ldots, k\}$ is an $n + k$ dimensional vector, and $r = rat(\frac{\sqrt{n}}{2})$ is a rational number, where $rat(a)$ is a function that takes as input a real number a and outputs a rational number in $[a, \sqrt{a^2+1})$. $\mathcal{B}$ then calls $\mathcal{A}$ on c and outputs what $\mathcal{A}$ outputs.

By Lemma 1, p is equivalent to s, namely they have the same solution x. What is left to show is that s and c fix each other.

We show that if s is a YES instance of d-MSS then c is a YES instance of GapCVP$_1$; and if s is a YES instance of GapCVP$_1$ then s is a YES instance of d-MSS.

Note that

$$||Bz - t||^2 = \sum_{j=1}^{k} |\sum_{i=1}^{n} x_i a_i^{(j)} + y_j N_j - X_j|^2 + \sum_{i=1}^{n} |x_i - \frac{1}{2}|^2,$$

where $z = (x_1, \ldots, x_n, y_1, \ldots, y_k) \in \mathbb{Z}^{n+k}$.

For the first direction, if s is a YES instance of d-MSS, then by Lemma 3, there exists a unique solution $(x_1, \ldots, x_n) \in \{0,1\}^n$ such that $\sum_{i=1}^{n} x_i a_i^{(j)} + y_j N_j - X_j = 0$ for all $j \in \{1, \ldots, k\}$. Hence $||Bz - t||_2 = \frac{\sqrt{n}}{2}$ and c is a YES instance of GapCVP$_1$.

For the second direction, if c is a YES instance of GapCVP$_1$, i.e., there exists $z \in \mathbb{Z}^{n+k}$ such that $||Bz - t||_2 \leq rat(\frac{\sqrt{n}}{2})$, then, since $\sum_{i=1}^{n} |x_i - \frac{1}{2}|^2 \geq \frac{n}{4}$ (with equality achieves only when $x_i \in \{0,1\}$ for all $i \in \{1, \ldots, n\}$), we must have $x_i \in \{0,1\}$ for all $i \in \{1, \ldots, n\}$ and at the same time $\sum_{i=1}^{n} x_i a_i^{(j)} + y_j N_j - X_j = 0$ for all $j \in \{1, \ldots, k\}$. Hence s is a YES instance of d-MSS. □

The proof of Lemma 2 in fact gives a reduction from MSP to CVP.

LEMMA 3. *Assuming the Gaussian heuristic and a DLP algorithm, if an MSP (with identical relative points) has at least one $j \in \{1, \ldots, k\}$ such that $q_j > (\sqrt{2\pi e})^n + 1$ and that $\{p_1^{(j)}, \ldots, p_n^{(j)}\}$ generates $\mathbb{Z}_{q_j}^*$, then it reduces to CVP.*

Proof. Observe the target vector t given by Equation (3.4), we see that the distance between any vector $By \in L(B)$ and t is $||By - t|| \geq r$. Hence a YES instance (B, t, r) of GapCVP$_1$ (where $||Bz - t|| \leq r$) fixes the solution Bz to the CVP (B, t). Again, by the two directions in the proof of Lemma 2, this Bz fixes the solution x to the MSS s, which is the same solution to the original MSP p. This x is easy to recover from $Bz - t$ (it is the first n entries of z). Hence we can solve for x of the MSP p by solving for Bz of the CVP (B, t). □

Reducing CVP to uSVP$_\gamma$

In the following we show that for very low density MSP, the corresponding CVP is solvable. We may assume $k < n$ since the case $k \geq n$ was not hard to deal with as it leaks too many Legendre symbol relations about x.

The following lemma will be used in the proof of Theorem 5.

LEMMA 4. *[LLL82]. There exists a polynomial time algorithm such that given $t \in \mathbb{R}^m$ and a basis B for a lattice $L(B)$, outputs a lattice vector Bz such that $||Bz - t|| \in [\text{dist}(t, L(B)), 2^m \text{dist}(t, L(B))]$.*

THEOREM 5. *Assuming the Gaussian heuristic, there exists a polynomial time (quantum) algorithm that solves MSP (with identical relative points) which has at least one $j \in \{1, \ldots, k\}$ such that $q_j > (\sqrt{2\pi e})^n + 1$ and that $\{p_1^{(j)}, \ldots, p_n^{(j)}\}$ generates $\mathbb{Z}_{q_j}^*$, and the moduli $q_1, \ldots, q_k$ satisfy*

$$\frac{\sqrt{n}}{2} \leq \frac{1 - \frac{1}{n+k}}{2 \cdot 2^{\frac{n+k+1}{4}}} \cdot \sqrt{\frac{n+k}{2\pi e}} \cdot \left(\prod_{j=1}^{k} (q_j - 1) \right)^{\frac{1}{n+k}}. \tag{3.5}$$

Note that Inequality 3.5 requires large product $\prod_{j=1}^{k} (q_j - 1)$, which implies large moduli q_j on average hence low density SP instances on average. Also note that when k is too small, e.g. $k = 2$, then Inequality 3.5 requires very low density SP instances. In that case, even SP was not conjectured to be post-quantum hard because the corresponding SS is not hard. So the algorithm only makes sense to work with $k \gg 2$ and TSP is not attacked by this algorithm.

Proof. The algorithm is the following.

Algorithm 1 MSP Algorithm

Input: $n, k \in \mathbb{N}$, $\{p_1^{(j)}, \cdots, p_n^{(j)}, q_j, X_j\}_{j \in \{1,\ldots,k\}}$ with p's and q's prime numbers and $X_j \in \mathbb{Z}_{q_j}$
Output: $x \in \{0,1\}^n$ or 0

1: choose random generator of $\mathbb{Z}_{q_j}^*$ and solve DLP of $p_i^{(j)}$ in $\mathbb{Z}_{q_j}^*$ to get $a_i^{(j)} \in \mathbb{Z}_{q_j - 1}$, for all $i \in \{1, \ldots, n\}, j \in \{1, \ldots, k\}$
2: create B and t as defined by Equation (3.3) and (3.4)
3: sample $\nu \in [\text{dist}(t, L(B)), 2^{n+k} \text{dist}(t, L(B))]$ and sample $\mu \in \{\nu/(1 + 1/(n+k))^i : 0 \leq i \leq \log_{1+1/(n+k)} 2^{n+k}\}$ and create $B' = \begin{bmatrix} B & t \\ 0 & \mu \end{bmatrix}$
4: call LLL to solve for the $2^{\frac{n+k+1}{4}}$-unique shortest vector in $L(B')$ to get $c' = (a, \alpha) \in \mathbb{Z}^n \times \mathbb{Z}$
5: compute $z = B^{-1}(a - t)$ and denote it as $z := (x, y) \in \mathbb{Z}^n \times \mathbb{Z}^k$
6: **return** x **if** $x \in \{0,1\}^n$ **else** 0

Given an MSP instance p, the MSP algorithm first reduces p to an MSS s and creates a CVP instance (B, t) with B and t defined by Equation (3.3) and (3.4) respectively. It then samples $\nu \in [\text{dist}(t, L(B)), 2^m \text{dist}(t, L(B))]$ (this can be done in polynomial

time according to Lemma 4) and samples $\mu \in \{\nu/(1+1/m)^i : 0 \le i \le \log_{1+1/m} 2^m\}$ (the goal is to get some $\mu \in [\mathrm{dist}(t, L(B)), (1+1/m)\,\mathrm{dist}(t, L(B))]$, which has noticeable probability since the set that μ samples from is polynomial size) and creates an extended basis

$$B' = \begin{bmatrix} B & t \\ 0 & \mu \end{bmatrix},$$

where $m := n + k$. It then uses LLL to solve for the γ-unique shortest vector $c' = (a, \alpha)$ in $L(B')$, where

$$\gamma = 2^{\frac{m+1}{4}}, \tag{3.6}$$

$a \in \mathbb{Z}^n$, $\alpha \in \mathbb{Z}$. This can be done in polynomial time since γ-uSVP is LLL solvable when $\gamma \approx O(2^{\mathrm{rank}(B')/4})$ [LSL13]. It eventually outputs the vector x, which is the first n entries of $z = B^{-1}(a + t)$, as the solution to p.

We show in the following that x is indeed the solution to p. It suffices to reduce p to the γ-uSVP in $L(B')$ with the γ-unique shortest vector to be

$$c' = B'x' = \begin{bmatrix} B & t \\ 0 & \mu \end{bmatrix} \begin{bmatrix} z \\ -1 \end{bmatrix} = \begin{bmatrix} Bz - t \\ -\mu \end{bmatrix},$$

where $z = (x, y) \in \mathbb{Z}^n \times \mathbb{Z}$ such that x is the solution of p.

We achieve the reduction by three steps:

(1) Reducing the MSP to some CVP,

(2) Verifying the CVP is a $\mathrm{BDD}_{(1-1/m)/(2\gamma)}$,

(3) Reducing the $\mathrm{BDD}_{(1-1/m)/(2\gamma)}$ to some uSVP_γ.

Step (1) is done by Lemma 3. We complete Step (2) and Step (3) in the following.

For Step (2), notice that $L(B)$ is a full rank lattice of rank m. By the Gaussian heuristic, the predicted length of the shortest vector in $L(B)$ is

$$\lambda_1 = \sqrt{\frac{m}{2\pi e}}\,\mathrm{vol}(L)^{\frac{1}{m}}, \tag{3.7}$$

where

$$\mathrm{vol}(L) = |\det(B)| = \prod_{j=1}^{k} (q_j - 1) \tag{3.8}$$

is the volume of $L(B)$.

Plug in Inequality (3.5) with the right hand sides of Equation (3.6), (3.7) and (3.8), we have $\sqrt{n}/2 < \lambda_1(1-1/m)/(2\gamma)$. Note also that $\mathrm{dist}(t, L(B)) \leq \sqrt{n}/2$. Hence $\mathrm{dist}(t, L(B)) < \lambda_1(1-1/m)/(2\gamma)$. By definition, the CVP instance (B, t) is a $\mathrm{BDD}_{(1-1/m)/(2\gamma)}$ instance. Step (2) is done.

For Step (3), we employ the method of [LM09] to show that $c' = (a, \alpha) = (Bz - t, -\mu)$ is the γ-unique shortest vector in $L(B')$ if and only if $c := Bz$ is the closest vector to t in $L(B)$ (and thus x is the correct solutions to the MSP by Lemma 3).

The "$\Rightarrow$" direction is obvious: If c is not the closest vector to t, then there exists another vector $v \neq t$ such that $||v - t|| < ||c - t||$, then $c'' = (v - t, -\mu)$ is a vector shorter than $c' = (c - t, -\mu)$, contradicting the assumption that c' is the γ-unique shortest vector in $L(B')$.

In the following we show the "$\Leftarrow$" direction. Suppose c is the closest vector to t. To show c' is the γ-unique shortest vector, it suffices to show that for all $v' \in L(B')$ which are not multiples of c' it holds that $||v'|| > \gamma \cdot ||c'||$. We show by contradiction that if there exists $v' \in L(B')$ such that $||v'|| \leq \gamma \cdot ||c'||$, then there exists a non-zero vector $u \in L(B)$ such that $||u|| < \lambda_1(B)$.

Let

$$v' = \begin{bmatrix} B & t \\ 0 & \mu \end{bmatrix} \begin{bmatrix} y \\ \beta \end{bmatrix} = \begin{bmatrix} By + \beta t \\ \beta\mu \end{bmatrix} = \begin{bmatrix} v + \beta t \\ \beta\mu \end{bmatrix},$$

where $\beta \in \mathbb{Z}$ and $v = By$.

Denote $\lambda_1 := \lambda_1(B)$, $d := \mathrm{dist}(t, L(B))$, $S := 1 - 1/m$ and $T := 1 + 1/m$. Since

$d \leq \mu \leq Td$ and $||c-t|| = d < \frac{S\lambda_1}{2\gamma}$, we have

$$\begin{aligned}
||c'|| &= \sqrt{||c-t||^2+\mu^2} \\
&= \sqrt{d^2+\mu^2} \\
&\leq \sqrt{d^2+(Td)^2} \\
&= \sqrt{1+T^2}\cdot d \\
&< \sqrt{1+T^2}\cdot\frac{S\lambda_1}{2\gamma} \\
&= \sqrt{(1+T^2)S^2}\cdot\frac{\lambda_1}{2\gamma} \\
&= \sqrt{\left[1+\left(1+\frac{1}{m}\right)^2\right]\cdot\left(1-\frac{1}{m}\right)^2}\cdot\frac{\lambda_1}{2\gamma} \\
&= \sqrt{2-\frac{2}{m}-\frac{1}{m^2}+\frac{1}{m^4}}\cdot\frac{\lambda_1}{2\gamma} \\
&< \frac{\lambda_1}{\sqrt{2}\gamma}.
\end{aligned}$$

For contradiction, suppose $||v'|| \leq \gamma \cdot ||c'||$, i.e.,

$$\begin{aligned}
\sqrt{||v+\beta t||^2+(\beta\mu)^2} &< \gamma\cdot\frac{\lambda_1}{\sqrt{2}\gamma}, \\
||v+\beta t|| &< \sqrt{\frac{\lambda_1^2}{2}-(\beta\mu)^2}.
\end{aligned}$$

Now consider the vector $u := v + \beta c \in L(B)$. Note that u is a non-zero vector since otherwise $v = -\beta c$ and

$$v' = \begin{bmatrix} v+\beta t \\ \beta\mu \end{bmatrix} = \begin{bmatrix} -\beta c+\beta t \\ \beta\mu \end{bmatrix} = -\beta\begin{bmatrix} c-t \\ -\mu \end{bmatrix}$$

is a multiple of c', contradicting our assumption that v' is not a multiple of c'.

Now we look at its length:

$$\begin{aligned}
||u|| &= ||v + \beta c|| \\
&= ||v + \beta t + \beta(c - t)|| \\
&\leq ||v + \beta t|| + \beta||c - t|| \\
&< \sqrt{\frac{\lambda_1^2}{2} - (\beta\mu)^2} + \beta d \\
&\leq \sqrt{\frac{\lambda_1^2}{2} - (\beta\mu)^2} + \beta\mu.
\end{aligned}$$

Take square of both sides we have

$$\begin{aligned}
||u||^2 &< \frac{\lambda_1^2}{2} - (\beta\mu)^2 + (\beta\mu)^2 + 2\beta\mu\sqrt{\frac{\lambda_1^2}{2} - (\beta\mu)^2} \\
&\leq \frac{\lambda_1^2}{2} + \left[(\beta\mu)^2 + \left(\sqrt{\frac{\lambda_1^2}{2} - (\beta\mu)^2}\right)^2\right] \\
&= \lambda_1^2,
\end{aligned}$$

where the second line is by the AM-GM inequality. Take square root of both sides we have the expected contradiction: $||u|| < \lambda_1$. Hence v' does not exist and c' is the γ-unique vector in $L(B')$. This completes the proof. □

3.4 Subset Product With Errors

Even though the lattice algorithm (i.e., Algorithm 1) only works for very low density MSP, we are not confident that higher density MSP are hard. In order to be confident we have a hard problem, we want to avoid the lattice algorithm completely.

Note that the lattice algorithm relies on the construction of the lattice basis B (as given by Equation (3.3)), whose entries $a_i^{(j)}$ are given by solving DLP of the primes $p_i^{(j)}$. Hence to protect the problem from the lattice algorithm, we can hide some of the primes $p_i^{(j)}$ to hinder the construction of B. That is, we only give $p_1, \ldots, p_s, q, X$ with $s < n$ and ask to find $x \in \{0,1\}^n$ as well as primes $p_{s+1}, \ldots, p_n$ such that $X = \prod_{i=1}^n p_i^{x_i} \pmod q$.

Note that hiding some primes $p_{s+1}, \ldots, s_n$ is equivalent to joining extra primes $p_{n+1}, \ldots, p_{n+k}$. Hence we define the problem as the following.

Definition 19 (Subset Product With Errors Problem, SPE). *Given $n+1$ primes $p_1, \ldots, p_n, q$ and integers $B_1 < B_2$, k, and $X \in \mathbb{Z}_q^*$, find different primes $\ell_1, \ldots, \ell_k \in$*

$\{B_1, \ldots, B_2\}$ *with* $\ell_i \neq p_j$ *for all* $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n\}$, *a vector* $(x_1, \ldots, x_n) \in \{0,1\}^n$ *and a vector* $(e_1, \ldots, e_k) \in \{-1, 1\}^k$ *such that* $X = \prod_{i=1}^{n} p_i^{x_i} \prod_{i=1}^{k} \ell_i^{e_i} \pmod q$.

By restricting the support of the error primes ℓ_i and the size of q, SPE can be set to have a unique solution $(\ell_1, \ldots, \ell_k, x_1, \ldots, x_n, e_1, \ldots, e_k)$. In the following we reduce SP to unique solution SPE.

THEOREM 6. *Unique solution SPE is at least as hard as SP.*

Proof. Let $(p_1, \ldots, p_n, q, X)$ be an SP instance with $X = \prod_{i=1}^{n} p_i^{x_i}$ for some $x = (x_1, \ldots, x_n) \in \{0,1\}^n$. Let $\mathcal{A}$ be a PPT algorithm that solves SPE. We solve the SP as follows. Sample k small primes $\ell_1, \ldots, \ell_k$ and a vector $(e_1, \ldots, e_k) \in \{-1, 1\}^k$. Compute $Y = X \prod_{i=1}^{k} \ell_i^{e_i} \pmod q$ and call $\mathcal{A}$ to solve $(p_1, \ldots, p_n, q, Y, k)$. $\mathcal{A}$ returns k small primes $t_1, \ldots, t_k$, a vector $y = (y_1, \ldots, y_n) \in \{0,1\}^n$ and a vector $f = (f_1, \ldots, f_k) \in \{-1, 1\}^k$ such that $Y = \prod_{i=1}^{n} p_i^{y_i} \prod_{i=1}^{k} t_i^{f_i} \pmod q$. Since we are working in the unique solution case of SPE, we have that $y = x$ is the solution to the SP. □

With a DLP algorithm, SPE reduces to the subset sum with errors problem defined in the following.

DEFINITION 20 (Subset Sum With Errors Problem, SSE). *Given* $n+3$ *integers* $a_1, \ldots, a_n, X, N, k$, *find a vector* $(x_1, \ldots, x_n) \in \{0,1\}^n$ *and a vector* $e_1, \ldots, e_k \in \{-1, 1\}^k$ *such that* $X = \sum_{i=1}^{n} a_i x_i + \sum_{i=1}^{k} b_i e_i \pmod N$ *for some integers* $b_1, \ldots, b_k$ *in some prefixed range and that* $b_i \neq a_j$ *for all* $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n\}$.

The twin version of the problem is the following.

DEFINITION 21 (Twin Subset Product With Errors Problem, TSPE). *Given two independent SPE instances with respect to a pair of twin points, find the twin points.*

When we speak of avoiding the MSP algorithm, we should consider the multi-instance version of the problem.

DEFINITION 22 (Multiple Subset Product With Errors Problem, MSPE). *Given* $k > 2$ *independent SPE instances with respect to* k *relative points* $x \in \{0,1\}^n$, *find the* k *points.*

The hardness of MSPE is clearer when it is converted into the form of subset sum.

DEFINITION 23 (Multiple Subset Sum With Errors Problem, MSSE). *Given* $k > 2$ *independent SSE instances with respect to* k *relative points* $x \in \{0,1\}^n$, *find the* k *points.*

Note that MSSE (with respect to MSPE over random primes) has a similar form to the LWE problem except that it has a different distribution of coefficients and errors.

Ideal Subset Product With Errors

Now we introduce the last variant of SPE, the ideal SPE. Note that the reason we use prime numbers is to take advantage of their irreducibility and the uniqueness of factorization of integers. Hence a natural generalization of SPE is to consider generic irreducible objects such as prime ideals of a Dedekind domain.

Definition 24 (Ideal Subset Product With Errors Problem, ISPE). *Given $n+1$ prime ideals $\mathfrak{p}_1, \ldots, \mathfrak{p}_n, \mathfrak{q} \in Spec(\mathcal{R})$ of a Dedekind domain $\mathcal{R}$, an ideal $\mathfrak{a} \subset \mathcal{R}/\mathfrak{q}$ of the quotient ring $\mathcal{R}/\mathfrak{q}$, and an integer k, find k different prime ideals $\mathfrak{l}_1, \ldots, \mathfrak{l}_k \in Spec(\mathcal{R})$ with $\mathfrak{l}_i \neq \mathfrak{p}_j$ for all $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n\}$, a vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ and a vector $(e_1, \ldots, e_k) \in \{-1, 1\}^k$ such that $\mathfrak{a} = \prod_{i=1}^{n} \mathfrak{p}_i^{x_i} \prod_{i=1}^{k} \mathfrak{l}_i^{e_i}$ in $\mathcal{R}/\mathfrak{q}$.*

Specific examples are prime ideals of an order (ring of integers) and prime ideals of a polynomial ring. In the following we give the polynomial ring variant with respect to univariate polynomial rings, the order variant is similar.

Definition 25 (Polynomial Ideal Subset Product With Errors Problem, PISPE). *Given $n+1$ prime (or irreducible) ideals $\mathfrak{p}_1, \ldots, \mathfrak{p}_n, \mathfrak{q} \subset \mathbb{F}_q[x]$, an ideal $\mathfrak{a} \subset \mathbb{F}_q[x]/\mathfrak{q}$ of the quotient ring $\mathbb{F}_q[x]/\mathfrak{q}$, and an integer k, find different prime ideals $\mathfrak{l}_1, \ldots, \mathfrak{l}_k \in \mathbb{F}_q[x]$ such that $\mathfrak{l}_i \neq \mathfrak{p}_j$ for all $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n\}$, a vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ and a vector $(e_1, \ldots, e_k) \in \{-1, 1\}^k$ such that $\mathfrak{a} = \prod_{i=1}^{n} \mathfrak{p}_i^{x_i} \prod_{i=1}^{k} \mathfrak{l}_i^{e_i}$ in $\mathbb{F}_q[x]/\mathfrak{q}$.*

The ideal variant of SP is the following.

Definition 26 (Ideal Subset Product Problem, ISP). *Given $n+1$ prime ideals $\mathfrak{p}_1, \ldots, \mathfrak{p}_n, \mathfrak{q} \in Spec(\mathcal{R})$ of a Dedekind domain $\mathcal{R}$, an ideal $\mathfrak{a} \subset \mathcal{R}/\mathfrak{q}$ of the quotient ring $\mathcal{R}/\mathfrak{q}$, find a vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ such that $\mathfrak{a} = \prod_{i=1}^{n} \mathfrak{p}_i^{x_i}$ in $\mathcal{R}/\mathfrak{q}$.*

The polynomial variant with respect to univariate polynomial rings is the following.

Definition 27 (Polynomial Ideal Subset Product Problem, PISP). *Given $n+1$ prime (or irreducible) ideals $\mathfrak{p}_1, \ldots, \mathfrak{p}_n, \mathfrak{q} \subset \mathbb{F}_q[x]$, an ideal $\mathfrak{a} \subset \mathbb{F}_q[x]/\mathfrak{q}$ of the quotient ring $\mathbb{F}_q[x]/\mathfrak{q}$, and a natural number $k \in \mathbb{N}$, find a vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ such that $\mathfrak{a} = \prod_{i=1}^{n} \mathfrak{p}_i^{x_i}$ in $\mathbb{F}_q[x]/\mathfrak{q}$.*

The multi-instance as well as decision versions of the problems can be defined in the natural way, which we do not go into details.

Chapter 4

SMALL SUPERSET OBFUSCATION

Given two sets $X, Y \subset \{1, \ldots, n\}$, a way to determine whether $Y \supset X$ and that the size of Y is not larger than some threshold $t \in \mathbb{N}$ is to check whether the corresponding characteristic vectors $x, y \in \{0,1\}^n$ satisfy $y - x \in \{0,1\}^n$ and $|y| \leq t$. We aim at giving small superset testing of X without leaking X.

Small superset functions (SSF) were first introduced in [BCJJLMMMR19]. Big subset functions (BSF) were first introduced in [BW19]. Both of them were originated from the conjunction obfuscation in [BLMZ19], which was in turn a dual scheme of the conjunction obfuscation in [BKMPRS18]. Conjunction obfuscation was also studied in [BVWW16; GZ19].

This chapter gives a new solution to SSF and BSF obfuscation based on SP, which is a much simpler and trustworthy assumption in the sense that it is at least as hard as DLP for some range of parameters (as shown in Chapter 3). Let r be the maximal difference of size allowed between an accepting input set and the set being obfuscated. We obfuscate SSF and BSF with $r \leq n/2 - \sqrt{\lambda n \ln 2}$. Conjunction is then redefined using SSF and BSF. An elegant obfuscator for conjunctions is presented based on TSP.

4.1 Small Superset Testing

Denote the Hamming weight of a binary string x as $|x|$.

Definition 28 (Small Superset Function, SSF). *Let $x \in \{0,1\}^n$ be a characteristic vector of a subset of $\{1, \ldots, n\}$. A small superset function with respect to x is a function $f_x : \{0,1\}^n \to \{0,1\}, y \mapsto f_x(y)$ such that $f_x(y) = 1$ if and only if $y - x \in \{0,1\}^n$ and $|y| \leq t$, where $t \in \mathbb{N}$ with $0 \leq t \leq n$ is a threshold indicating "small".*

Definition 29 (Big Subset Function, BSF). *Let $x \in \{0,1\}^n$ be a characteristic vector of a subset of $\{1, \ldots, n\}$. A big subset function with respect to x is a function $f_x : \{0,1\}^n \to \{0,1\}, y \mapsto f_x(y)$ such that $f_x(y) = 1$ if and only if $x - y \in \{0,1\}^n$ and $|y| \geq t$, where $t \in \mathbb{N}$ with $0 \leq t \leq n$ is a threshold indicating "big".*

Note that we only need to study SSF obfuscation since BSF obfuscation can be

reduced to SSF obfuscation. To see this, let f_x with threshold t be a BSF. Then $f_{\bar{x}}$ with threshold $n-t$ is an SSF, where $\bar{x}$ is the complement of x. So if we can obfuscate SSF we can also obfuscate BSF by firstly converting BSF to SSF.

Also, to simplify the security analysis, we consider function families where the sets have the same size. I.e., all x in the function family have the same Hamming weight.

Evasive Function Family

Denote by $B_{n,w}$ the set of binary strings of length n and Hamming weight w. Denote by $U_{n,w}$ the uniform distribution over $B_{n,w}$.

Only "evasive" SSF are interesting to obfuscate since x is immediately leaked once a small superset y of x is leaked.

PROPOSITION 1. *There is a polynomial time algorithm which learns the set x of an SSF given a small superset of x and black-box access to the SSF.*

Proof. Let y be a small superset of x. We flip the 1's of y one by one and queries the black-box of f_x. If the y with a 1-position flipped is still a small superset of x, then the corresponding position of x is a 0; otherwise it is a 1. Running through all 1's in y we learn x. In particular, if all the flipped y's are rejected, then we learn that $x = y$. □

We define evasive SSF families in the following.

DEFINITION 30 (Evasive SSF Family). *Let λ be the security parameter and n, t, w with $t \leq n$, $w \leq n$ be polynomial in λ. Let $\{X_n\}_{n\in\mathbb{N}}$ be an ensemble of distributions over $B_{n,w}$. The corresponding SSF family is said to be evasive if there exists a negligible function $\mu(\lambda)$ such that for every $\lambda \in \mathbb{N}$, and for every $y \in \{0,1\}^n$:*

$$\Pr_{x\leftarrow X_n}[f_x(y) = 1] \leq \mu(\lambda). \tag{4.1}$$

Parameters For Evasive Function Family

Now we consider parameters for evasive function family. Let us start with uniform distributions $\{X_n\}_{n\in\mathbb{N}} = \{U_{n,w}\}$, remembering that n and w are polynomials in λ.

If $|y| < w$ or $|y| > t$, then y will never be a small superset of any x with Hamming weight w hence Inequality (4.1) always holds. If $w \leq |y| \leq t$, then there are at most $\binom{t}{w}$ many x with Hamming weight $|x| = w$ such that y is a superset of x, Inequality (4.1)

holds if and only if

$$\binom{t}{w}/\#B_{n,w} = \binom{t}{w}/\binom{n}{w} \leq 1/2^{\lambda}. \tag{4.2}$$

An asymptotic way to see this is the following. By Stirling's approximation, when $k \ll n$ we have $\binom{n}{k} \approx n^k/(k^k\sqrt{2\pi k})$. So if $w \ll t \ll n$ then $\binom{n}{w}/\binom{t}{w} \approx (n/t)^w$. Also note that this is the most basic requirement for t in the sense that it is obtained under the best possible (i.e., highest entropy) distributions.

Now we consider general distributions $\{X_n\}_{n\in\mathbb{N}}$. We first define the following entropy.

Definition 31 (Conditional Small Superset Min-Entropy). *Let $0 \leq t \leq n \in \mathbb{N}$. The small superset min-entropy of a random variable X conditioned on a correlated variable Y is defined as*

$$H_{Sup,\infty}(X \mid Y) = -\ln\left(\mathbb{E}_{y\leftarrow Y}[\max_{y\in\{0,1\}^n} \Pr[y - X \in \{0,1\}^n, |y| \leq t \mid Y = y]]\right).$$

Now let us see what exactly Inequality (4.1) means. In words, it means that for every $y \in \{0,1\}^n$, an x sampled from the distribution X_n has negligible probability to have y as a small superset. Intuitively, this requires that in the space $\{0,1\}^n$, the number of points x representing SSF is large enough and at the same time they are "well spread out" in the sense that small superset relations between points occur sparsely and evenly in the space. Rigorously, the following requirement implies Inequality (4.1). where we now include auxiliary information.

Definition 32 (Small Superset Evasive Distribution). *Let $X = \{(X_n, \alpha_n)\}_{n\in\mathbb{N}}$ be an ensemble of distributions on $B_{n,w} \times \{0,1\}^{poly(\lambda)}$. We denote a sample from X_n as (x, α) where $\alpha \in \{0,1\}^{poly(\lambda)}$ is considered to be auxiliary information about x. We say that X is small superset evasive if the conditional small superset min-entropy of x conditioned on α (as in Definition 31) is at least λ.*

Note that asking for a small superset is a stronger question than asking for a "close" set. Hence the above requirement is somehow looser than the evasiveness requirement for fuzzy Hamming distance matching. Intuitively, in the case of fuzzy Hamming distance matching, we require that the points in the Hamming space are spread out such that their Hamming balls do not overlap too seriously; while in the case of SSF, the Hamming balls can overlap more seriously. For example, let $x = (01||c)$ and $y = (10||c)$ be two strings with only the first two bits different, where $c \in \{0,1\}^{n-2}$. We can see

that x and y have very small Hamming distance $|x \oplus y| = 2$, but neither of them is a superset of the other.

This means that in the same space $\{0,1\}^n$, there are more evasive SSF distributions than evasive fuzzy Hamming distance matching distributions.

Nonetheless, our obfuscation for SSF has to work under the stronger requirement of evasive Hamming distance matching. This is because an attacker can always recover the secret in our scheme x by merely finding a "close" set and not necessarily a small superset. We therefore use the following definition for evasiveness of SSF.

Definition 33 (Hamming Distance Evasive Distribution [GZ19]). *Let λ be the security parameter and n, t, w with $t \leq n$, $w \leq n$ be polynomial in λ. Let $X = \{X_n\}_{n \in \mathbb{N}}$ be an ensemble of distributions on $B_{n,w} \times \{0,1\}^{poly(\lambda)}$. We say that the distribution X_n is Hamming distance evasive if for all $\lambda \in \mathbb{N}$, the conditional Hamming ball min-entropy of x conditioned on α (as in Definition 9 with $r := t - w$) is at least λ.*

4.2 Small Superset Obfuscation

We explain the obfuscation as follows. The high level idea is to encode $x \in \{0,1\}^n$ as a subset product $X = \prod_{i=1}^{n} p_i^{x_i} \pmod q$ with respect to some small primes $p_1, \ldots, p_n$ and a larger prime modulus q so that if and only if an input $y \in \{0,1\}^n$ is a small superset of x (which implies that x and y have many bits in common) the product $\prod_{i=1}^{n} p_i^{y_i - x_i}$ is smaller than q and thus

$$YX^{-1} \pmod q = \prod_{i=1}^{n} p_i^{y_i - x_i} \pmod q = \prod_{i=1}^{n} p_i^{y_i - x_i}$$

factors over $\{p_1, \ldots, p_n\}$, where $Y = \prod_{i=1}^{n} p_i^{y_i} \pmod q$. We explain the idea explicitly as follows.

Let $n, t \in \mathbb{N}$ with $t < n$. Let $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ with Hamming weight w and $r = t - w$. We require $r \leq n/2$. To obfuscate, the obfuscator samples n different small primes $p_1, \ldots, p_n$ from $\{2, \ldots, B\}$ for some sufficiently large $B \in \mathbb{N}$, and a safe prime q such that $B^r < q < (1 + o(1))B^r$. It then computes the product $X = \prod_{i=1}^{n} p_i^{x_i} \pmod q$ and publishes $(p_1, \ldots, p_n, q, X)$ as the obfuscated function.

To evaluate with an input $y = (y_1, \ldots, y_n) \in \{0,1\}^n$, the obfuscated function firstly checks if $w \leq |y| \leq t$. If not then it terminates and outputs 0. If $w \leq |y| \leq t$, then it

further computes $Y = \prod_{i=1}^{n} p_i^{y_i} \pmod{q}$ and $E = YX^{-1} \pmod{q} = \prod_{i=1}^{n} p_i^{y_i - x_i} \pmod{q}$, and tries to factor E by dividing it by the primes $p_1, \ldots, p_n$ one by one. If $y - x \in \{0,1\}^n$ then E factors over $\{p_1, \ldots, p_n\}$ and the function outputs 1. Otherwise, if $y - x \notin \{0,1\}^n$, which means y is not a superset of x, then with high probability E will not factor over $\{p_1, \ldots, p_n\}$ and the function outputs 0. The obfuscator is as follows.

Algorithm 2 SSF Obfuscator

Input: $n, t, r, w \in \mathbb{N}$, $x \in \{0,1\}^n$ with $r := t - w \leq n/2 - \sqrt{\lambda n \ln 2}$

Output: $((p_1, \ldots, p_n) \in \mathbb{N}^n, q \in \mathbb{N}, X \in \mathbb{Z}_q^*)$

1: sample distinct primes $p_1, \ldots, p_n$ from $\{2, \ldots, B\}$ where $B = 3n \ln n$
2: sample safe prime q from $\{B^r, \ldots, 3B^r\}$
3: compute $X = \prod_{i=1}^{n} p_i^{x_i} \mod q$
4: **return** $(p_1, \ldots, p_n, q, X)$

Note that in Algorithm 2 we require $r \leq \frac{n}{2} - \sqrt{\lambda n \ln 2}$ due to Inequality (3.1).

The following factoring algorithm (Algorithm 3) is a sub-procedure of the evaluation algorithm (Algorithm 4).

Algorithm 3 Factor

Input: $n \in \mathbb{N}$, $(p_1, \ldots, p_n) \in \mathbb{N}^n$, $a \in \mathbb{N}$

Output: 0 or 1

1: **for** $i = 1, \ldots, n$ **do**
2: **if** $p_i | a$ **then** $a \leftarrow a / p_i$
3: **end for**
4: **return** 1 **if** $a = 1$ **else** 0

The evaluation algorithm is the following.

Algorithm 4 SSF Evaluation (with embedded data $(p_1, \ldots, p_n) \in \mathbb{N}^n$, $q \in \mathbb{N}$, $X \in \mathbb{Z}_q^*$)

Input: $y \in \{0,1\}^n$

Output: 0 or 1

1: $F \leftarrow 0$
2: **if** $w \leq |y| \leq t$ **then**
3: compute $Y = \prod_{i=1}^{n} p_i^{y_i} \pmod{q}$
4: compute $E = YX^{-1} \pmod{q}$
5: compute $F \leftarrow \mathsf{Factor}(n, (p_1, \ldots, p_n), E)$
6: **end if**
7: **return** 1 **if** $F = 1$ **else** 0

Note that if y is not a small superset of x, then it will either (1) result in some E which contains a prime factor not in $\{p_1, \ldots, p_n\}$ or $e \notin \{0,1\}^n$; or (2) result in some E such that E is still a product of primes in $\{p_1, \ldots, p_n\}$ and the factorization of E is square-free. The former case will be correctly rejected by Algorithm 4. The latter case will be falsely accepted. We call a $y \in \{0,1\}^n$ a *false positive* if it is not a small superset of x but is accepted by Algorithm 4.

Now we discuss how to avoid false positives.

Let y be a false positive. We have that $E = \prod_{i=1}^n p_i^{y_i - x_i} \pmod q = \prod_{i=1}^n p_i^{e_i} \pmod q$ with $\prod_{i=1}^n p_i^{e_i} < q$ and $e = (e_1, \ldots, e_n) \in \{0,1\}^n$. I.e., $\prod_{i=1}^n p_i^{y_i - x_i - e_i} = 1 \pmod q$ with $y - x - e \neq 0$. This implies a nonzero short vector $z \in \{-2, -1, 0, 1\}^n$ of length $\leq 2\sqrt{n}$ in the lattice

$$L = \left\{ z \in \mathbb{Z}^n \;\middle|\; \prod_{i=1}^n p_i^{z_i} = 1 \pmod q \right\}.$$

To avoid false positives, it is sufficient that the shortest vector in the above lattice is longer than $2\sqrt{n}$. If the primes $p_1, \ldots, p_n$ are sufficiently random, which means that the lattice is sufficiently random, then we can employ the Gaussian heuristic to estimate the length of the shortest vector as

$$\lambda_1 \sim \sqrt{\frac{n}{2\pi e}} \operatorname{vol}(L)^{\frac{1}{n}}.$$

Also, by the first isomorphism theorem, the volume of the lattice $\operatorname{vol}(L)$ is given by the size of the image $|\operatorname{im} \phi|$ of the group morphism

$$\phi : \mathbb{Z}^n \to \mathbb{Z}_q^*,$$

$$(x_1, \ldots, x_n) \mapsto \prod_{i=1}^n p_i^{x_i} \pmod q$$

whose kernel defines L. Hence

$$\operatorname{vol}(L) \leq \varphi(q) = q - 1,$$

where φ is the Euler totient function. The equality holds if and only if $\{p_1, \ldots, p_n\}$

generates $\mathbb{Z}_q^*$. So

$$\lambda_1 \sim \sqrt{\frac{n}{2\pi e}}\,\mathrm{vol}(L)^{\frac{1}{n}} \leq \sqrt{\frac{n}{2\pi e}}(q-1)^{\frac{1}{n}} < \sqrt{\frac{n}{2\pi e}}q^{\frac{1}{n}}.$$

If we take $\lambda_1 = \sqrt{\frac{n}{2\pi e}}q^{\frac{1}{n}}$ and $q \sim (3n\ln n)^r > (n\ln n)^r$, for $\lambda_1 > 2\sqrt{n}$ we require that

$$r > \frac{n\ln(2\sqrt{2\pi e})}{\ln(n\ln n)}. \tag{4.3}$$

If r satisfies this condition then heuristically there are no false positives.

Evidence for the Gaussian Heuristic in These Lattices

To provide evidence for the Gaussian heuristic on the relation lattice L, we give some experimental results. Due to the limitation of computational resources, we only work with small parameters such as $n = 20$ or 30 or 40, $r = \lfloor \frac{n}{\ln n} \rfloor$ (which is an appropriate choice as we will be discussing in the later section about parameters), and $B = 3n\ln n$.

Let λ_1 denote the length of the shortest vector in a lattice and let γ denote the Gaussian heuristic. For each $n = 20$ or 30 or 40, we create 1000 lattices L from random subset products, calculate the proportion of lattices that λ_1/γ falls into the 20 intervals $[0.0, 0.1), [0.1, 0.2), \ldots, [1.9, 2.0]$, respectively. The results are as follows.

When $n = 20$, $r = \lfloor \frac{n}{\ln n} \rfloor$, $B = 3n\ln n$, the sequence of proportions is:

$$(0,0,0,0,0,0,0,0,0,0,\frac{9}{20},\frac{11}{20},0,0,0,0,0,0,0,0).$$

When $n = 30$, $r = \lfloor \frac{n}{\ln n} \rfloor$, $B = 3n\ln n$, the sequence of proportions is:

$$(0,0,0,0,0,0,0,0,\frac{2}{1000},\frac{26}{1000},\frac{399}{1000},\frac{557}{1000},\frac{16}{1000},0,0,0,0,0,0,0).$$

When $n = 40$, $r = \lfloor \frac{n}{\ln n} \rfloor$, $B = 3n\ln n$, the sequence of proportions is:

$$(0,0,0,0,0,0,0,0,0,\frac{29}{1000},\frac{702}{1000},\frac{269}{1000},0,0,0,0,0,0,0,0).$$

We can see that for most cases $\lambda_1/\gamma \in [1.0, 1.2]$, which means that the Gaussian heuristic is quite close to the true length of the shortest vectors most of the time. Also λ_1 tends to be larger than γ, which gives more confidence in Inequality (4.3) to avoid

false positives.

Dealing with False Positives by Hashing Another way to deal with false positives is to use a hash function or a point function obfuscator. Let us take hash as an example. To avoid false positives, all we need to do is to compute and output an extra value $h = H(x)$ in Algorithm 2, where H is a collision resistant hash function modeled as a random oracle; and in Factor, store the factors of E in a list F and replace "return 1" with "return F"; also in Algorithm 4, add a process to recover x from F and compare its hash value against $H(x)$. If y is a small superset of x, then the factors of E will tell the positions of the distinct bits between x and y, then one can recover x by flipping y at those positions. Otherwise if y is a false positive, then doing so will give a wrong $x' \neq x$ which can be detected by comparing the hash values.

Functionality Preservation

THEOREM 7. *Assuming the Gaussian heuristic and Inequality 4.3, the obfuscator given by Algorithm 2 and 4 is functionality-preserving.*

Proof. Note that the inputs y with $|y| < w$ or $|y| > t$ will always be correctly rejected. We therefore only discuss the case where $w \leq |y| \leq t$.

Let $E = YX^{-1} \pmod q = \prod_{i=1}^{n} p_i^{e_i} \pmod q$ with $e = (e_1, \ldots, e_n) = y - x \in \{-1, 0, 1\}^n$. If y is a small superset of x, then $e \in \{0, 1\}^n$ and $|e| \leq r$, hence $\prod_{i=1}^{n} p_i^{e_i} < B^r < q$. This means E is a product of primes in $\{p_1, \ldots, p_n\}$ hence will be reduced to 1 in Factor and y will be correctly accepted by Algorithm 4.

On the other hand, false positives are avoided by Inequality 4.3 assuming the Gaussian heuristic. Hence if y is not a small superset of x, it can only result in some E which contains a prime factor not in $\{p_1, \ldots, p_n\}$ or $e \notin \{0, 1\}^n$. This will be correctly rejected by Algorithm 4. □

Polynomial Slowdown

To show polynomial slowdown compared to the unobfuscated function, it is sufficient to show polynomial complexity of the obfuscated function.

THEOREM 8. *The time complexity of the obfuscated function given by Algorithm 4 is polynomial in λ.*

Proof. In the obfuscating algorithm (Algorithm 2), we sample $n+1$ primes, perform $n-1$ modular multiplications of integers of size $< q$. Therefore the time complexity of the obfuscation is linear in the number of modular multiplications of integers of size $< q$.

Again, in the evaluation algorithm (Algorithm 4), we perform $n-1$ modular multiplications of integers of size $< q$ to compute Y, and 1 inversion, 1 modular multiplication of integers of size $< q$ to compute E, also n inversions and n modular multiplications of integers of size $< q$ to run Factor (Algorithm 3). Therefore the time complexity of the evaluation is also linear in the number of modular multiplications of integers of size $< q$.

Now since $q < (1+o(1))B^r$, we have $\ln q < r\ln((1+o(1))B) < n\ln(cn\ln n) = poly(\lambda)$, where c is a constant. Hence the time complexity of the obfuscated function is polynomial in λ hence has polynomial slowdown compared to the original function. □

Security

The security is based on hardness assumptions that are slightly different from Assumption 1 and 2. We consider SP and d-SP over points $x \in \{0,1\}^n$ with fixed Hamming weight $w \approx n/2$ and with auxiliary information given.

The following assumption serves the proof of input-hiding, which involves some global auxiliary information $\alpha \in \{0,1\}^{poly(\lambda)}$ about the entire function family.

Assumption 5 (Hard SP with Global Auxiliary Information). *Let X_n be a distribution over $B_{n,w}$ where $n/2 - n/8 \le w \le n/2 + n/8$. Let $\alpha \in \{0,1\}^{poly(\lambda)}$ be auxiliary information. For every PPT algorithm A, for every $\lambda \in \mathbb{N}$, there exists a negligible function $\mu(\lambda)$ such that the probability that A, provided with α, solves an SP sampled from the (n, r, B, X_n)-SP distribution is not greater than $\mu(\lambda)$.*

We then state the hard d-SP assumption which serves the proof of DVBB. In contrast to Assumption 5 where the auxiliary information α is global information for the entire function family, the following Assumption 6 assumes that d-SP is hard even given local auxiliary information α about the specific x sampled from X_n. Furthermore, for convenience in proving distributional-indistinguishabiliy, we define d-SP in the "predicate-augmentation" style (as in Definition 6), namely to define it over a distribution D'_b which outputs $\alpha' = (\alpha, \varphi(x))$ instead of just α, for any (non-uniform) polynomial size predicate $\varphi : X_n \to \{0,1\}$, where $b \in \{0,1\}$.

Assumption 6 (Hard d-SP with Local Auxiliary Information). *Fix a (non-uniform) polynomial time predicate $\varphi : \{0,1\}^n \to \{0,1\}$. Let X_n be a distribution over $B_{n,w} \times \{0,1\}^{poly(\lambda)}$ which samples (x, α) with α some auxiliary information about x that satisfies Definition 33 (i.e., the conditional Hamming ball min-entropy of the distribution X_n conditioned on α is still at least λ). Let $X'_n = (x, \alpha')$ be a distribution over $B_{n,w} \times \{0,1\}^{poly(\lambda)} \times \{0,1\}$, where $\alpha' = (\alpha, \varphi(x))$. Let $D'_0 = (p_1, \ldots, p_n, q, X, \alpha')$ be the (n, r, B, X_n)-SP distribution corresponding to X'_n. Let D'_1 be D'_0 with $X = \prod_{i=1}^n p_i^{x_i} \pmod q$ replaced by uniformly sampled $X' \leftarrow \mathbb{Z}_q^*$, but all other terms the same. Then for every PPT algorithm A, for every $\lambda \in \mathbb{N}$, there exists a negligible function $\mu(\lambda)$ such that*

$$\left| \Pr_{d_0 \leftarrow D'_0}[A(d_0) = 1] - \Pr_{d_1 \leftarrow D'_1}[A(d_1) = 1] \right| \leq \mu(\lambda). \tag{4.4}$$

Now we show input-hiding from the hardness of SP.

Theorem 9. *Let n, t, r, B satisfy Definition 10, the Gaussian Heuristic and Inequality (4.3). Then assuming the hardness of SP (Assumption 5), the SSF obfuscator given by Algorithm 2 is input-hiding.*

Proof. Let $(p_1, \ldots, p_n, q, X)$ with $X = \prod_{i=1}^n p_i^{x_i} \pmod q$ for some unknown $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ be an SP instance defined in Assumption 5, and $\alpha \in \{0,1\}^{poly(\lambda)}$ be some auxiliary information. Let $\mathcal{A}$ be a PPT algorithm that breaks input-hiding of the obfuscation given by Algorithm 2-4. Then we solve the SP as follows. We directly call $\mathcal{A}$ on input $(p_1, \ldots, p_n, q, X, \alpha)$. Since r satisfies Inequality (4.3), i.e., there are no false positives, $\mathcal{A}$ will return a small superset y of x such that $E = YX^{-1} \pmod q = \prod_{i=1}^n p_i^{y_i - x_i} \pmod q = \prod_{i=1}^n p_i^{e_i}$ with $e = (e_1, \ldots, e_n) \in \{0,1\}^n$. Then we can factor E to get e and recover x by flipping y at the positions i such that $e_i = 1$. □

We show DVBB from the hardness of d-SP.

Theorem 10. *Let X_n be a distribution over $B_{n,w} \times \{0,1\}^{poly(\lambda)}$ with conditional (on α) Hamming ball min-entropy λ. Then assuming Assumption 6, the obfuscation given by Algorithm 2 - 4 is DVBB (with heuristic correctness if we use the lattice technique to avoid false positives).*

Proof. Functionality preservation and polynomial slowdown are shown in the proof of Theorem 9. Now we show distributional VBB. We show distributional-indistinguishability, which implies DVBB by Theorem 1. Fix a predicate φ. For every circuit $C \leftarrow C_\lambda$

(which contains the secret $x \leftarrow X_n$), let $O(1^\lambda, C) = (p_1, \ldots, p_n, q, X)$ be the obfuscated function of C. We define a simulator S which works as follows: S takes $\pi = (n, t, B)$, samples n primes $p'_1, \ldots, p'_n$ and a modulus q' in the same way as O, and samples $X' \leftarrow \mathbb{Z}_q$. Denote $S(1^\lambda, \pi) = (p'_1, \ldots, p'_n, q', X')$. We will show that the two probabilities in Inequality (2.4) equal the two probabilities in Inequality (4.4) respectively.

For the first equality, we have that for every PPT distinguisher $\mathcal{A}$, for every $\lambda \in \mathbb{N}$,

$$\Pr_{(x,\alpha') \leftarrow X'_n}[\mathcal{A}(p_1, \ldots, p_n, q, X, \alpha') = 1] = \Pr_{d_0 \leftarrow D'_0}[\mathcal{A}(d_0) = 1],$$

where $d_0 = (p_1, \ldots, p_n, q, X, \alpha')$ and both probabilities are over the randomness of $x, p_1, \ldots, p_n$, q and α'. This holds simply from the definition of D'_0 (as in Assumption 6).

Replace x with C, X'_n with D'_λ, and $p_1, \ldots, p_n, q, X$ with $O(1^\lambda, C)$ we have that

$$\Pr_{(C,\alpha') \leftarrow D'_\lambda}[\mathcal{A}(O(1^\lambda, C), \alpha') = 1] = \Pr_{d_0 \leftarrow D'_0}[\mathcal{A}(d_0) = 1], \tag{4.5}$$

where the first and the second probabilities are the first probabilities of Inequality (2.4) and Inequality (4.4) respectively.

For the second equality, we have that for every PPT distinguisher $\mathcal{A}$, for every $\lambda \in \mathbb{N}$,

$$\Pr_{(x,\alpha') \leftarrow X'_n}[\mathcal{A}(p'_1, \ldots, p'_n, q', X', \alpha') = 1] = \Pr_{d_1 \leftarrow D'_1}[\mathcal{A}(d_1) = 1],$$

where $d_1 = (p'_1, \ldots, p'_n, q', X', \alpha')$ and the probability is over the randomness of $x, p'_1, \ldots, p'_n, q'$, X' and α'. This holds from the definition of D'_1 (as in Assumption 6). Note that the α' in both probabilities are the same α' as in Equation (4.5), which is the auxiliary information about the unique real x sampled at the beginning of the game. In particular the α' in d_1 is not generated by the simulator but copied from the left hand side.

Replace x with C, X'_n with D'_λ, and $p'_1, \ldots, p'_n, q', X'$ with $S(1^\lambda, \pi)$ we have that

$$\Pr_{(C,\alpha') \leftarrow D'_\lambda}[\mathcal{A}(S(1^\lambda, \pi), \alpha') = 1] = \Pr_{d_1 \leftarrow D'_1}[\mathcal{A}(d_1) = 1], \tag{4.6}$$

where the first and the second probabilities are the second probabilities of Inequality (2.4) and Inequality (4.4) respectively.

By Assumption 6, there exists a negligible function $\mu(\lambda)$ such that the difference between the right hand sides of Equation (4.5) and Equation (4.6) is not greater that $\mu(\lambda)$. Therefore the difference between the left hand sides of Equation (4.5) and Equa-

tion (4.6) is not greater that $\mu(\lambda)$. I.e., Inequality (2.4) holds. This completes the proof. □

Parameters

Restrictions for the parameters λ, n, t, r, and q are as follows.

(1) For evasiveness, the basic requirement is Inequality (4.2).

(2) For the hardness of finding a y close to x such that it decodes (which will recover x), we require r to be small enough, i.e., the Hamming ball of any x should be small enough. This requires $r(n) \leq n/2 - \sqrt{\lambda n \ln 2}$. (Inequality (3.1)).

(3) To avoid false positives without using a hash function or a point function obfuscator, we require $r > \frac{n \ln(2\sqrt{2\pi e})}{\ln(n \ln n)}$ (Inequality (4.3)).

From (2) and (3) we have that

$$\frac{n \ln(2\sqrt{2\pi e})}{\ln(n \ln n)} < r(n) \leq \frac{n}{2} - \sqrt{n\lambda \ln 2}.$$

Notice that

$$\frac{n \ln(2\sqrt{2\pi e})}{\ln(n \ln n)} \prec \frac{n}{\ln n} \prec \frac{n}{\ln \ln n} \prec \frac{n}{2} - \sqrt{n\lambda \ln 2},$$

both $r(n) \sim \frac{n}{\ln n}$ and $r(n) \sim \frac{n}{\ln \ln n}$ are possible functions for r, where by $f \sim g$ we mean $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$ and by $f \prec g$ we mean $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

If we take $r(n) = \lfloor \frac{n}{\ln n} \rfloor$, then the condition $r \leq \frac{n}{2} - \sqrt{n\lambda \ln 2}$ gives

$$\begin{aligned} & \sqrt{n\lambda \ln 2} \leq n \left(\frac{1}{2} - \frac{1}{\ln n} \right) \\ \iff & \lambda \leq \frac{n}{\ln 2} \left(\frac{1}{2} - \frac{1}{\ln n} \right)^2 \\ \impliedby & \lambda \leq \frac{n}{6}, \end{aligned}$$

where for the last line we assume $n \geq 1024$.

Hence a possible choice of r is the prime counting function $r(n) = \lfloor n/\ln n \rfloor \sim \pi(n)$, i.e., the number of primes not greater than n. A possible function family for the uniform distribution $B_{n,w}$ is $(n = 6\lambda, r = \lfloor n/\ln n \rfloor)$. In terms of t, it is $(n = 6\lambda, t = w + \lfloor n/\ln n \rfloor)$. A concrete setting is: $\lambda = 128$; $n = 1024$; $t = 659$; $w = 512$; $B = 8161$ (the 1024-th prime, 13 bits); $q \approx 2B^r$ (about 1912 bits); X_n has conditional Hamming ball min-entropy λ. Note that this X_n is easy to achieve with the settings of n, w

and t, because n is much larger than λ and there is a big gap between a λ min-entropy distribution and the uniform distribution. Even when we consider auxiliary information which reduces the entropy a little bit, it is still easy to have a λ min-entropy distribution conditioned on the auxiliary information. Also note that an elementary requirement is that $w > r$ since otherwise the encoding of x, namely $\prod_{i=1}^{n} p_i^{x_i} \pmod q$ is aways factorable and x will be exposed immediately.

Nevertheless, the choice $r = \lfloor n/\ln n \rfloor$ does not satisfy the size of q required by the DLP-to-SP reduction (as given by Theorem 2), which is $q \leq 2^n p(n)$ for some polynomial $p(n)$. Indeed no r satisfies this requirement. Namely even if we take the lower bound $r = n\ln(2\sqrt{2\pi e})/\ln(n\ln n)$ we still result in a $q > 2^n p(n)$ for any polynomial $p(n)$. However we still conjecture that SP is hard for the above parameters.

Application: Conjunction Obfuscation

As an application of TSP, we now show how to obfuscate conjunctions by redefining them using SSF and BSF. This obfuscation is a simpler solution than the conjunction obfuscation in [GZ19], which relies on a complicated algorithm to deal with continued fractions.

Conjunction is also called pattern matching with wildcards function. It is typically defined as a function $f_{n,r,x} : \{0,1\}^n \to \{0,1\}$ such that $f_{n,r,x}(y) = 1$ if and only if $y_i = x_i$ for all $x_i \neq *$, where $x \in \{0,1,*\}^n$ is a pattern with $r < n \in \mathbb{N}$ wildcards. In the following we denote the number of 1's in x by $|x|$.

We give an equivalent definition based on SSF and BSF. The intuition is to do both superset and subset tests so that any unmatched bit at the non-wildcard positions will be exposed.

DEFINITION 34. *Let $x \in \{0,1,*\}^n$ be a pattern with $r < n \in \mathbb{N}$ wildcards. A conjunction is a boolean function $f_{n,r,x} : \{0,1\}^n \to \{0,1\}$ such that $f_{n,r,x}(y) = 1$ if and only if y is a subset of x with all wildcards being replaced by 1 and at the same time y is a superset of x with all wildcards being replaced by 0.*

Note that when both the "superset" and "subset" conditions are satisfied, they are naturally "small" and "big" respectively.

Evasiveness

Let $w \in \{0,1\}^n$ be x with the wildcard positions set to 0, and $z \in \{0,1\}^n$ be x with the wildcard positions set to 1. We use two functions to define the conjunction $f_{n,r,x}(y)$, which are the SSF $f_{n,t_1,w}(y)$ with $|w| = |x|$, $t_1 = |x|+r$, and the BSF $f_{n,t_1',z}(y)$ with $|z| = |x|+r$, $t_1' = |x|$. If we further convert the BSF into an SSF, it is the SSF $f_{n,t_2,\bar{z}}(\bar{y})$ with $|\bar{z}| = n - |z| = n - |x| - r$ and $t_2 := n - t_1'$, where $\bar{z}$ is the complement of z.

Suppose w and z are uniform. By the approximation form of Inequality (4.2), for evasivenesses of both $f_{n,t_1,w}(y)$ and $f_{n,t_2,\bar{z}}(\bar{y})$, we require both $t_1^{|w|}/n^{|w|} \leq 1/2^\lambda$ and $t_2^{|\bar{z}|}/n^{|\bar{z}|} \leq 1/2^\lambda$. Plug in $|w| = |x|$, $|\bar{z}| = n - |x| - r$, $t_1 = |x|+r$ and $t_2 = n - |x|$ we have $[(|x|+r)/n]^{|x|} \leq 1/2^\lambda$ and $[(n-|x|)/n]^{n-|x|-r} \leq 1/2^\lambda$. Suppose $\lambda \leq n/2$ and $|x| = n/2$, the second inequality gives $r \leq n/2 - \lambda$. This also satisfies the first inequality if we plug in $\lambda \leq n/2$ and $|x| = n/2$. Hence for evasiveness parameters, we can think of $\lambda \leq n/2$, $|x| \approx n/2$ and $r \leq n/2 - \lambda$.

Obfuscation

The high level idea is the following. Let $n, r, B \in \mathbb{N}$ with $r < n/2$ and $B \in O(n \ln n)$. Let $f_{n,r,x}$ be a conjunction. To obfuscate $f_{n,r,x}$, we derive two binary strings $w, z \in \{0,1\}^n$ from x, where w is x with the wildcard positions set to 0, and z is x with the wildcard positions set to 1. We then choose two sequences of small primes, $(p_1, \ldots, p_n)$ and $(\ell_1, \ldots, \ell_n)$, from $\{2, \ldots, B\}$, and two primes q and s from $\{B^r, \ldots, (1+o(1))B^r\}$. We encode w and z into two different subset products as:

$$X_1 = \prod_{i=1}^{n} p_i^{w_i} \pmod{q},$$
$$X_2 = \prod_{i=1}^{n} \ell_i^{z_i} \pmod{s}.$$

Then we output $(p_1, \ldots, p_n, \ell_1, \ldots, \ell_n, q, s, X_1, X_2)$ as the obfuscated function. The algorithm is the following.

Algorithm 5 Conjunction Obfuscator

Input: $n \in \mathbb{N}$, $r \in \mathbb{N}$, $x \in \{0,1,*\}^n$

Output: $(p_1, \ldots, p_n, \ell_1, \ldots, \ell_n) \in \mathbb{N}^{2n}, q, s \in \mathbb{N}, X_1, X_2 \in \mathbb{Z}_q^*$

1: set w as x with $* \leftarrow 0$, set z as x with $* \leftarrow 1$, and set $\bar{z}$ as the complement of z

2: sample distinct primes $p_1, \ldots, p_n, \ell_1, \ldots, \ell_n$ from $\{2, \ldots, B\}$ where $B \in O(n \ln n)$

3: sample safe primes q, s from $\{B^r, \ldots, (1+o(1))B^r\}$

4: compute $X_1 = \prod_{i=1}^n p_i^{w_i} \mod q$ and $X_2 = \prod_{i=1}^n \ell_i^{\bar{z}_i} \mod s$

5: **return** $(p_1, \ldots, p_n, \ell_1, \ldots, \ell_n, q, s, X_1, X_2)$

To evaluate with an input $y \in \{0,1\}^n$, we compute

$$Y_1 = \prod_{i=1}^n p_i^{y_i} \pmod q,$$
$$Y_2 = \prod_{i=1}^n \ell_i^{\bar{y}_i} \pmod s,$$

and $E_1 = Y_1 X_1^{-1} \pmod q$, $E_2 = Y_2 X_2^{-1} \pmod s$. We then use Algorithm 3 to factor E_1 using the primes $p_1, \ldots, p_n$, and factor E_2 using $\ell_1, \ldots, \ell_n$. If both E_1 and E_2 factor successfully, then output 1, otherwise output 0. The evaluation algorithm is as follows.

Algorithm 6 Conjunction Evaluation (with embedded data $(p_1, \ldots, p_n, \ell_1, \ldots, \ell_n) \in \mathbb{N}^{2n}$, $q, s \in \mathbb{N}$, $X_1, X_2 \in \mathbb{Z}_q^*$

Input: $y \in \{0,1\}^n$

Output: 0 or 1

1: compute $Y_1 = \prod_{i=1}^n p_i^{y_i} \pmod q$ and $Y_2 = \prod_{i=1}^n \ell_i^{\bar{y}_i} \pmod s$

2: compute $E_1 = Y_1 X_1^{-1} \pmod q$ and $E_2 = Y_2 X_2^{-1} \pmod s$

3: compute $F_1 \leftarrow \mathsf{Factor}(n, (p_1, \ldots, p_n), E_1)$ and $F_2 \leftarrow \mathsf{Factor}(n, (\ell_1, \ldots, \ell_n), E_2)$

4: **return** 1 **if** $F_1 = F_2 = 1$ **else** 0

Avoiding False Positives Using Lattice Arguments

Denote $U_1 = \prod_{i=1}^n p_i^{w_i}$, $U_2 = \prod_{i=1}^n \ell_i^{\bar{z}_i}$, $V_1 = \prod_{i=1}^n p_i^{y_i}$, and $V_2 = \prod_{i=1}^n \ell_i^{\bar{y}_i}$. Then $E_1 = V_1/U_1 \pmod q$, $E_2 = V_2/U_2 \pmod s$. We define a *false positive* to be a $y \in \{0,1\}^n$ such that y is not a matching pattern to x yet both E_1 and E_2 factor successfully in Algorithm 6.

Let us see how to avoid false positives. Note that a false positive $y \in \{0,1\}^n$ gives a

short vector $u = y - w - e_1 \in \{-2, -1, 0, 1\}^n$ in the lattice

$$L_1 = \left\{ u \in \mathbb{Z}^n \;\middle|\; \prod_{i=1}^{n} p_i^{u_i} = 1 \pmod{q} \right\},$$

and a short vector $v = \bar{y} - \bar{z} - e_2 \in \{-2, -1, 0, 1\}^n$ in the lattice

$$L_2 = \left\{ v \in \mathbb{Z}^n \;\middle|\; \prod_{i=1}^{n} \ell_i^{v_i} = 1 \pmod{s} \right\},$$

where $e_1, e_2 \in \{0, 1\}^n$ are "good" error vectors, meaning that the corresponding products E_1, E_2 factor successfully. In other words, a false positive means that the same y gives short vectors in the two lattices L_1, L_2 simultaneously. Hence to avoid false positives, it is sufficient (more than enough) to avoid short vectors in both lattices. Notice that this is implied by the lattice analysis for SSF in Section 4.2. Hence the restriction to r is the same as Inequality 4.3.

Functionality Preservation

THEOREM 11. *Assuming the Gaussian heuristic and Inequality 4.3, the obfuscator given by Algorithm 5 and 6 is functionality-preserving.*

Proof. If y matches x at all non-wildcard positions, then y is a small superset of w and a big subset of z (i.e., y is a small superset of w and $\bar{y}$ is a small superset of $\bar{z}$), so $E_1 = V_1/U_1$ is an integer $< q$, and $E_2 = V_2/U_2$ is an integer $< s$, then both the factorings of E_1 and E_2 will succeed and Algorithm 6 will correctly output 1.

On the other hand, false positives are avoided by Inequality 4.3 assuming the Gaussian heuristic. Hence if there is any non-wildcard position of x does not matched by y, then at least one of V_1/U_1 and V_2/U_2 will be a proper rational, and thus at least one of E_1 and E_2 will not factor successfully. Then Algorithm 6 correctly will output 0. □

Polynomial Slowdown

To show polynomial slowdown compared to the unobfuscated function, it is sufficient to show polynomial complexity of the obfuscated function.

THEOREM 12. *The time complexity of the obfuscated function given by Algorithm 6 is polynomial in λ.*

Proof. It is not hard to see that the time Algorithm 6 takes is about twice the time the SSF evaluation algorithm (i.e., Algorithm 4) takes. Hence by Theorem 8, the time complexity of Algorithm 6 is polynomial in λ. □

Security

We prove input-hiding security of our scheme. The security is based on the hardness of TSP.

THEOREM 13. *Let n, t, r, B satisfy Definition 10, the Gaussian Heuristic and Inequality (4.3). Then assuming the hardness of TSP (Assumption 4), the conjunction obfuscation given by Algorithm 5-6 is input-hiding.*

Proof. Let $((p_1, \ldots, p_n, q, X), (\ell_1, \ldots, \ell_n, s, X'))$ be a TSP with respect to the a pair of twin points (x, x') such that x' is superset of x. Let $\mathcal{A}$ be a PPT algorithm that solves the obfuscation given Algorithm 5-6. We solve the TSP as follows. We query $\mathcal{A}$ on the TSP above. Since the TSP is exactly an obfuscation instance, $\mathcal{A}$ can solve for a $y \in \{0,1\}^n$ which is a superset of x and a subset of x'. We then compute and factor $E = XY^{-1} \pmod q = \prod_{i=1}^n p_i^{x'_i - y_i} \pmod q$ to obtain the error vector $e = x' - y \in \{0,1\}^n$. Then we can recover x' by flipping y at the positions i such that $e_i = 1$. This contradicts the hardness of TSP hence $\mathcal{A}$ does not exist and the obfuscator given by Algorithm 5-6 is input-hiding. □

The DVBB security is plausible assuming the hardness of the decisional TSP, which we do not go into details.

Chapter 5

ALGEBRAIC MEMBERSHIP OBFUSCATION

An algebraic set is the solution set of some polynomial equations. Given a point x and an algebraic set X defined by some polynomials $f_1, \ldots, f_m$, the usual way to determine if $x \in X$ is to determine if $f_i(x) = 0$ for all $i \in \{1, \ldots, m\}$. We are curious about how to give membership testing of an algebraic set without leaking the polynomials.

Previous work handled the special cases of hyperplane and hypersurfaces over large finite fields [CRV10; BBCKPS14]. This dissertation is the first to address the problem for smaller fields and in greater generality.

Canetti, Rothblum, and Varia [CRV10] gave a solution to a special case of this problem. Let $\mathbb{F}_q$ be a finite field of order q. A hyperplane is defined by a vector $a = (a_1, \ldots, a_n) \in \mathbb{F}_q^n$ orthogonal to it. A hyperplane membership function with respect to a hyperplane through the origin is a Boolean function $f_a : \mathbb{F}_q^n \to \{0, 1\}$ such that $f_a(x) = 1$ if and only if $x = (x_1, \ldots, x_n)$ is a nonzero root of the homogeneous linear polynomial $f(x_1, \ldots, x_n) = a_1 x_1 + \cdots + a_n x_n$. Let G be a group of order q generated by g and let 1_G be its identity. The idea of [CRV10] is to encode each coefficient a_i by the discrete logarithm problem (DLP) as g^{a_i}, and to evaluate on an input $x = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ by determining whether $(g^{a_1})^{x_1} \cdots (g^{a_n})^{x_n} = 1_G$. However DLP is not hard when q is small or when it is not prime. Hence their method only works when q is an exponentially large prime. Moreover, due to the $O(\sqrt{q})$ complexity of some generic DLP algorithms such as the Baby-Step Giant-Step algorithm [Sha73; Nec94] and Pollard's rho algorithm [Pol78], they need $q > 2^{2\lambda}$ to achieve λ-bit security.

Later, Barak, Bitansky, Canetti, Kalai, Paneth, and Sahai [BBCKPS14] extended the work of [CRV10] from hyperplanes to hypersurfaces. But they still only consider a single equation and rely on an idealized multi-linear map whose existence is currently a dream.

This chapter studies the obfuscation problem of more general algebraic sets given by multiple equations with an arbitrary prime power q. Specifically, let λ be the security parameter, ℓ be the number of monomials of bounded degree, and m be the number of polynomial equations. We obfuscate algebraic set membership for prime powers $q \geq 2$ and $\lambda / \log_2 q < m < \ell$.

5.1 Algebraic Membership Testing

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 35 (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 $(A, b) \in \mathbb{F}_q^{m \times (\ell+1)}$ with $b \neq 0$ be the augmented matrix of the following system of equations over $\mathbb{F}_q$:*

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

where $M_1, \ldots, M_\ell \in \mathbb{F}_q[x_1, \ldots, x_n]$ are fixed monomials that can be evaluated in polynomial time (i.e., with bounded degree). 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 36 (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 $A \in \mathbb{F}_q^{m \times \ell}$ be the coefficient matrix of the following system of homogeneous equations over $\mathbb{F}_q$:*

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

where $M_1, \ldots, M_\ell \in \mathbb{F}_q[x_0, \ldots, x_n]$ are fixed monomials of the same total degree and that can be evaluated in polynomial time, also M_i and M_j do not share a common invariant for $i \neq j$. 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 non-zero root of the equations.

Note that p-ASMF only accept non-zero roots because 0 is not a point in projective space.

Evasive Function Family

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

PROPOSITION 2. *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 exist a PPT adversary $\mathcal{A}$ which finds a basis of the ideal $I(V)$ given black-box access to $f_{A,b}$.*

Proof. Let m' be the row rank of (A, b). We define $\mathcal{A}$ as follows. $\mathcal{A}$ keeps sampling random points x from $\mathbb{F}_q$ and queries the black-box of $f_{A,b}$ until having $\ell - m'$ linearly independent accepting vectors of the form $(M_1(x), \ldots, M_\ell(x))$, for $\ell - m'$ different accepting inputs x. $\mathcal{A}$ then solves for the orthogonal complement (A', b') of the $\ell - m'$ vectors. Notice that $\mathrm{span}(A, b) = \mathrm{span}(A', b')$. Hence (A', b') is the coefficient matrix of a basis of $I(V)$. □

By Proposition 2, 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 1 immediately.

DEFINITION 37 (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 $\lambda \in \mathbb{N}$,*

$$\Pr_{A,b \leftarrow \mathbb{F}_q^{m \times (\ell+1)}}[f_{A,b}(x) = 1] \leq \mu(\lambda). \tag{5.1}$$

DEFINITION 38 (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 $\lambda \in \mathbb{N}$,*

$$\Pr_{A \leftarrow \mathbb{F}_q^{m \times \ell}}[f_A(x) = 1] \leq \mu(\lambda). \tag{5.2}$$

Evasiveness gives a restriction on the monomials. Note that the function family is about a fixed public monomial sequence. One issue we might concern about is whether

the public monomials themselves leak accepting inputs. For example, suppose the monomial sequence is $(x_1x_2, x_2x_3, \dots)$, then all nonzero points of the form $(0 : x_2 : 0 : \cdots : 0)$ or $(x_1 : 0 : x_3 : 0 : \cdots : 0)$ are roots to a p-ASMF, meaning that no matter how we hide the coefficient matrix, input-hiding will never be achieved. However, this concern is not necessary because this kind of monomial sequences are already avoided by evasiveness. This is because if the monomial sequence is $(x_1x_2, x_2x_3, \dots)$, then for a nonzero point of the form $(0 : x_2 : 0 : \cdots : 0)$ or $(x_1 : 0 : x_3 : 0 : \cdots : 0)$, the probability that a random p-ASMF accepts this point is 100%, which contradicts the evasiveness requirement.

Parameters For Large Enough Algebraic Set

It is generally inefficient (super-polynomial complexity) to solve polynomial equation systems. Hence in general we do not need to worry about the trivial obfuscation which simply hashes every single point in the algebraic set. However in the cases where the systems are efficiently solvable (e.g., linear equation systems), we require that the size of the algebraic set to be super-polynomial to avoid the trivial obfuscator. For this it is sufficient to require that

$$q^{n-m} \geq spoly(\lambda) \tag{5.3}$$

for some super-polynomial $spoly(\lambda)$. This gives a requirement for the gap between m and ℓ according to the size of q. Namely if q is smaller, we require the gap larger; if q is larger, then we can have m closer to ℓ.

Parameters For Evasive Function Family

In the affine case, for an $x \in \mathbb{F}_q^n$, let $M = (M_1(x), \dots, M_\ell(x)) \in \mathbb{F}_q^n$ be the vector with x plugged in. 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 $q^{m(\ell+1)}$. 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

$$\Pr_{A,b \leftarrow \mathbb{F}_q^{m \times (\ell+1)}}[f_{A,b}(x) = 1] = \frac{q^{m\ell}}{q^{m(\ell+1)}} = \frac{1}{q^m} \leq \frac{1}{2^\lambda}. \tag{5.4}$$

Similarly, the requirement for the projective case is

$$\Pr_{A \leftarrow \mathbb{F}_q^{m \times \ell}}[f_A(x) = 1] = \frac{q^{m(\ell-1)}}{q^{m\ell}} = \frac{1}{q^m} \leq \frac{1}{2^\lambda}. \tag{5.5}$$

Both Inequality (5.4) and (5.5) give

$$q^m \geq 2^\lambda. \tag{5.6}$$

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 [BBCKPS14], of which a special case is hyperplane membership [CRV10].

Note that Inequality (5.6) gives the full generality of the ASMF obfuscation problem one can solve. In this chapter we solve it in approximately full generality.

5.2 Algebraic Membership Obfuscation

We explain our techniques via the affine case. The projective case is similar.

We obfuscate an a-ASMF $f_{A,b}$ as follows. We first do a primary randomization by performing a change of basis on (A, b) to get $(\bar{A}, \bar{b}) := R(A, b) = (RA, Rb)$, where $R \in \mathbb{F}_q^{m \times m}$ is a random matrix sampled from the set $Invt(\mathbb{F}_q^{m \times m})$ of invertible matrices in $\mathbb{F}_q^{m \times m}$. We then sample k random rows $(A', b') \leftarrow \mathbb{F}_q^{k \times (\ell+1)}$ such that $m + k \gg \ell$. 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 $s = (s_1, \ldots, s_{m+k}) \in \{0,1\}^{m+k}$ 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 O_S be an input-hiding SSF obfuscator (e.g., the obfuscator in [BCJJLMMMR19] or in Chapter 4). We publish $(A^*, b^*, O_S(f_s))$ as the obfuscated function. The details are given in Algorithm 7.

Algorithm 7 AMSF 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: $(A^* \in \mathbb{F}_q^{(m+k)\times \ell}, b^* \in \mathbb{F}_q^{(m+k)}, O_S(f_s))$

1: choose a polynomial size a such that $2^{\lambda/m} < (1+a)/(1+a/q+o(1)) < q$
2: set $k = am$, $t = \lceil (1+a/q+o(1))m \rceil$
3: sample $R \leftarrow Invt(\mathbb{F}_q^{m\times m})$ 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 $O_S(f_s)$
9: **return** $(A^*, b^*, O_S(f_s))$

Let us look at some parameter examples. 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 $a = 794$.

The evaluation is as follows. It takes as input an x, evaluate all $m + k$ equations on x 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 x is a solution of the i-th equation, for all $i \in \{1, \ldots, m+k\}$. It eventually outputs what $O_S(f_s)$ outputs on s'. The algorithm is the following.

Algorithm 8 AMSF Evaluation (with embedded data $(M, A^*, b^*, O_S(f_s))$

Input: $x \in \{0,1\}^n$

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** $O_S(f_s)(s')$

Functionality Preservation

THEOREM 14. *The obfuscator O_A given by Algorithm 7-8 is approximate-functionality-preserving.*

Proof. 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 a proper number for "$o(1)$" we can ensure that the probability that x satisfies more than t rows is negligible in λ. (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 $\approx 2.008 \times 10^{-46} < 1/2^{128} \approx 2.939 \times 10^{-39}$.) Hence s' is a small superset of s with overwhelming probability and thus $O_S(f_s)(s') = 1$ 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 $O_S(f_s)(s') = 0$ and the obfuscated a-ASMF will correctly output 0. □

Polynomial Slowdown

To show polynomial slowdown compared to the unobfuscated function, it is sufficient to show polynomial complexity of the obfuscated function.

THEOREM 15. *The time complexity of the obfuscated function given by Algorithm 8 is polynomial in λ.*

Proof. Algorithm 8 evaluates $m + k = (1 + a)m$ polynomials of bounded degree, generates a vector s' of polynomial length, and evaluates $O_S(f_s)$ on s'. Due to the polynomial size of a and the polynomial complexity of O_S, Algorithm 8 has polynomial complexity. □

Security

We first show input-hiding.

THEOREM 16. *The obfuscator O_A given by Algorithm 7 is input-hiding on evasive algebraic set membership functions assuming input-hiding of the small superset obfuscator O_S.*

Proof. Let $\mathcal{A}$ be any PPT algorithm that breaks input-hiding of the ASMF obfuscator O_A given by Algorithm 7 with probability $\mu(\lambda)$. We construct an algorithm $\mathcal{B}$ against

input-hiding of the SSF obfuscator O_S with success probability $\nu(\lambda)$ and show that $\nu(\lambda) = \mu(\lambda)$.

Given an obfuscated SSF $O_S(f_s)$ with its parameters satisfying the requirements in Algorithm 7 (we will come back to it at end of the proof to discuss its existence), 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 set the parameters m, k and ℓ as in the above paragraph. Then $\mathcal{B}$ samples a random $(m+k)\times(\ell+1)$ matrix (R^*, r^*) and calls $\mathcal{A}$ with $(R^*, r^*, O_S(f_s), \alpha, \beta)$ for a solution of the equation $Rx = r$, where (R, r) are the rows indicated by s, and $\beta \in \{0,1\}^{poly(\lambda)}$ is the global auxiliary information of the ASMF family. Since the distribution of (R^*, r^*) and (A^*, b^*) (as in Algorithm 7) 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 O_S is equal to the probability that $\mathcal{A}$ breaks input-hiding of O_A. Hence $\nu(\lambda) = \mu(\lambda)$. By the input-hiding property of O_S, $\nu(\lambda)$ 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.

To show the existence of $O_S(f_s)$, we consider the SSF obfuscator in Chapter 4. Note that the parameters q, m, k, ℓ for our ASMF obfuscation is $k = am$ such that $2^{\lambda/m} < (1+a)/(1+a/q+o(1)) < q$, and $t = \lceil (1+a/q+o(1))m \rceil$. Also $n' = m+k$. On the other hand, a suggested family of SSF in Chapter 4 is $(n' \geq 6\lambda, t \leq m + n'/\ln n')$, where n' is the secret length, m is the secret weight, and t is the "small" threshold. Now, note that $t = \lceil (1+a/q+o(1))m \rceil \leq m + n'/\ln n'$ is always true for $q \geq 2$. The only requirement left is $n' \geq 6\lambda$, which is easy to satisfy by choosing a large enough a. (A concrete setting for ASMF obfuscation and the SSF obfuscation its uses is: $q = 4, m = 2\lambda, a = 2$, $k = am = 4\lambda$, $\ell := 3m = 6\lambda$, $n' = m + k = 6\lambda$ and $t \approx (1 + a/q)m = 3\lambda = n'/2$.) □

We then show the obfuscator hides the span of the coefficient matrix A.

THEOREM 17. *The obfuscator given by Algorithm 7 hides the span of the coefficient matrix $A \in \mathbb{F}_q^{m\times(\ell-1)}$ if $q^{\ell-m} \geq 2^\lambda$, assuming input-hiding of the small superset obfuscator O_S .*

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ which, with noticeable probability, computes the Hermite normal form $H(A)$ of A, then there exists a PPT algorithm $\mathcal{B}$ which breaks input-hiding of the obfuscator given by Algorithm 7 hence (by Theorem 16) breaks input-hiding of the SSF obfuscator O_S.

Given an obfuscated ASMF $(A^*, b^*, O_S(f_s), \alpha)$, $\mathcal{B}$ works as follows: It directly queries $\mathcal{A}$ with $(A^*, b^*, O_S(f_s), \alpha)$. By assumption, $\mathcal{A}$ outputs $H(A)$. $\mathcal{B}$ then tests which rows of A^* are in $\mathrm{span}(H(A))$, and solves for a point x that satisfies the corresponding rows in (A^*, b^*) (we call these rows the selected rows). $\mathcal{B}$ outputs x.

We show that with overwhelming probability the solution x exists and is an accepting input. It is easy to see that if x exists then it is an accepting input since all the m real rows are being selected by $\mathrm{span}(H(A))$ and if x satisfies all selected rows, it must satisfy all the real rows. To see its existence, just notice that the only case that the selected rows are not consistent is when there exists a row (A_i^*, b_i^*) which is a linear combination of the m real rows except that the constant entry b_i^* is flipped, but this happens with negligible probability. This is because $q^{\ell-m} \geq 2^\lambda$ and the probability that a random ℓ-dimensional vector falls into $\mathrm{span}(H(A))$ is $\leq 1/2^\lambda$. Hence in the k dummy rows (polynomially many) of A^*, we expect < 1 row are in $\mathrm{span}(H(A))$. Hence with overwhelming probability the selected rows are precisely the m real rows. Hence with overwhelming probability $\mathcal{B}$ computes a correct solution x to $Ax = b$. This contradicts Theorem 16. Hence $\mathcal{A}$ does not exist and $\mathrm{span}(A)$ is hidden. □

Theorem 17 means that our obfuscation not only hides the algebraic set, but also hides its shape, i.e., the shape of the geometric locus, which is given by $\mathrm{span}(A)$.

Parameters

Now we explain the parameters of our obfuscator (as given by Algorithm 7) 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 (4.2).

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 (5.6). We therefore only need to consider condition (3). It gives a restriction on m as

$$m \geq \lambda/(\log_2(1+a)/c). \tag{5.7}$$

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 (5.6)). 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 7, 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 $q \geq spoly(\lambda)$ for any super-polynomial $spoly(\lambda)$. 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 (5.7).

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 (5.6).

Chapter 6

A GENERAL METHOD

This chapter derives a general obfuscation methodology from the algebraic membership obfuscation. This gives us a new view to the relations between other open problems.

The key point is to view algebraic sets as closed sets of the Zariski topology on the variety, and that an algebraic set $V(I)$ of an ideal $I = (f_1, \ldots, f_m)$ is the intersection of the algebraic sets $V(f_1), \ldots, V(f_m)$ of the ideal generators $f_1, \ldots, f_m$, i.e., $V(I) = V(f_1) \cap \cdots \cap V(f_m)$. Then the natural imagination beyond closed sets in a Zariski topology is that we can talk about closed sets (or more naturally, open sets) of a generic topology.

In the following we call a membership testing function of any object obj an *object membership function*, denoted $f_{obj} : X \to Y$, where $f_{obj}(x) = 1$ if and only if $x \in obj$. We call obj the *object* of the function. For example, an algebraic set membership function $f_{V(I)}$ is an object membership function with its object the algebraic set $V(I)$.

Now we replay the algebraic set membership obfuscation to gain intuition of the general method. The algebraic set membership obfuscation includes three process: algebraic set representation, algebraic set randomization and small superset obfuscation.

- Algebraic set representation: Represent an algebraic set $V(I)$ of an ideal $I = (f_1, \ldots, f_m)$ as $V(I) = V(f_1) \cap \cdots \cap V(f_m)$.
- Algebraic set randomization: Mix the real generating algebraic sets $V(f_1), \cdots, V(f_m)$ with k dummy algebraic sets $V(f'_1), \ldots, V(f'_k)$ to get a sequence of algebraic sets $S = (V(f^*_1), \ldots, V(f^*_{m+k}))$, where each f^* is either an f or an f'.
- Small superset obfuscation: Obfuscate the small superset function f_s with respect to the characteristic vector $s \in \{0,1\}^{m+k}$ which indicates the positions of the real algebraic sets in the sequence S.

Now replace the algebraic sets with generic objects, we have a generic object membership obfuscator $O := (A_{rep}, A_{ran}, O_{mat})$, where

- A_{rep}: is an object representation finding algorithm which takes as input an object obj of an object membership function f_{obj} and outputs a set of generators of obj: $\{obj_1, \ldots, obj_m\}$ such that $obj = (obj_1, \ldots, obj_m)$.

- A_{ran}: is an object randomization algorithm which takes as input a sequence of objects $\{obj_1, \ldots, obj_m\}$ and outputs a permuted sequence of objects $(obj_1^*, \ldots, obj_m^*)$ containing the m real objects $obj_1, \ldots, obj_m$ and k dummy objects $obj_1', \ldots, obj_k'$.
- O_{mat}: is a matching obfuscator which takes as input a matching function f_s and outputs an obfuscated function $O(f_s)$ of f_s.

6.1 Object Representation

Note that it is not secure to mix a single object with a polynomial number of other objects. This is because brute forcing one element from polynomially many is easy. Hence the first question is how to represent an object as many different objects.

Unlike Zariski topology where the generators $V(f_1), \ldots, V(f_m)$ of an algebraic set $V(I)$ are naturally given by its ideal $I = (f_1, \ldots, f_m)$, there is no handy generators for an open set of a generic topology. For example, in a metric space, given a metric ball B, there is no handy superballs $B_1, \ldots, B_m$ of B such that B is precisely the intersection of these superballs.

Moreover, the concept of topology is an unnecessary restriction. To better find generators for an object, we generalize topology and define "genology" to capture the intuition of "generation" in mathematics.

Genology

Recall that a topology τ on a set X is a collection of subsets of X which (1) contains $\varnothing$ and X; (2) is closed under arbitrary unions; and (3) is closed under finite intersections. We generalize it by preserving only the last axiom, i.e., closure under finite intersections.

DEFINITION 39. *A genology γ on a set X is a collection of subsets of X which is closed under finite intersection.*

We call the sets in a genology *genosets*, meaning "generating sets". Similar to topological space, we call the pair (X, γ) (or simply X) a *genological space*. We also call the genosets of a topology the *genosubsets* of the topological space.

The following remarks give intuitions of Definition 39.

REMARK 1 (Essence of Generation). *In most contexts of mathematics, by "generation" we essentially mean "set intersection". Specifically, let X be a set, when we say that a set $S \subset X$ is generated by a set $T \subset S$, we usually mean that S*

is the intersection of all subsets $S_1 \cdots, S_\ell$ of X that contain T. We can instead say that S is generated by a minimal subset of $\{S_1, \ldots, S_\ell\}$ that is enough to give T. For example, when we say that the algebraic set $V(f_1, \ldots, f_m)$ is generated by $V(f_1), \ldots, V(f_m)$, we mean that $V(f_1, \ldots, f_m)$ is precisely the intersection of $V(f_1), \ldots, V(f_m)$.

REMARK 2 (Finiteness of Intersections). *The reason to require closure under "finite" intersections comes from the requirement of computational feasibility by function obfuscation. In fact, this makes the definition more general because a structure that is closed under infinite intersections must be closed under finite intersections hence is a genology.*

REMARK 3 (Closure Under Intersections). *The reason to require "closure" under finite intersections is that we want to ensure that genologies are "dense" in the sense that for every subset S of a genology γ, there exists an element $\alpha \in \gamma$ which is the intersection of the elements in S. On the contrary, if we do not require this, then the worst case is that γ is a very "sparse" structure such that the intersection of any two elements is empty. For example, the collection $\{\varnothing, \{a\}, \{b\}, \{c\}, \ldots\}$ of single element subsets together with the empty set $\varnothing$. This makes the definition meaningless for "generation'."*

Common Genological Spaces

An immediate observation is that topology and σ-algebra are all genologies.

PROPOSITION 3. *Every topology is a genology. Every σ-algebra is a genology.*

Proof. By definition. I.e., the axioms of topology (resp., σ-algebra) imply the axiom of genology. □

By the definitions of the corresponding spaces, we have the following corollary.

COROLLARY 1. *Every topological space is a genological space. Every measurable space is a geological space.*

Another immediate observation is that vector spaces and modules are all genological spaces.

PROPOSITION 4. *Every vector space is a genological space. Every module is a genological space.*

Proof. Let V be a vector space (resp. module). Let γ be a set of all vector subspaces (resp., submodules) of V. Note that the intersection of two subspaces (resp., submodules) is still a subspace (resp., submodule). I.e., γ is closed under intersection. Hence γ is a genology and V is a genological space. □

Finding Representation of Objects by Defining Genology

The generality of genology gives us a more flexible way to find generators for an object. Let $\mathcal{F} = \{f_{obj} : X \to Y\}_{obj}$ be a function family. We find generators for the objects obj by defining a genology γ on the function domain X such that each obj in the function family is the intersection of some genosets in γ, i.e., $obj = obj_1 \cap \cdots \cap obj_m$.

Note also that a basic requirement from input-hiding is that none of the generators obj_i individually leaks a member of obj. Otherwise one can immediately distinguish the real generators from the dummy ones by membership testing for the leaked member. More precisely, what we really require is that the intersection of any efficiently recoverable generators is large enough so that a random point in the intersection has low probability to lie in obj.

Take algebraic set membership as an example. We represent an algebraic set as: $V(I) = V(f_1) \cap \cdots \cap V(f_m)$. We require that the intersection of any $d < m$ algebraic sets $V(f_i)$ is exponential times larger than $V(I)$ hence does not leak a point in $V(I)$.

6.2 Object Randomization

As we have seen in the algebraic set membership obfuscation, the randomization method is very simple. The basic idea is the exponential blow up of combinations, i.e., the hardness of finding m objects from $m + k$ randomly permuted objects. We call the randomization process the "mix & permute" randomization.

The "mixing" process is easy. It simply means to put the m real generators with k dummy objects together. As to "permutation", the simplest one is uniform permutation. I.e., to permute the $m + k$ objects uniformly, as what we did in the algebraic set membership obfuscation.

A basic requirement for the mix & permute randomization is that m and k should be chosen such that (1) the $m + k$ choose m attack is inefficient; (2) the accepting input finding attack by choosing $d < m$ generators from $m + k$ objects is inefficient; and (3) satisfy the restriction from the matching obfuscation.

6.3 Matching Obfuscation

The simplest way to evaluate the randomized sequence $(obj_1^*, \ldots, obj_{m+k}^*)$ is to perform pattern matching. The intuition is: If and only if the input $x \in obj$ can it spot the positions of the real objects $obj_1, \ldots, obj_m$ from $(obj_1^*, \ldots, obj_{m+k}^*)$. Take algebraic set membership obfuscation as an example, the intuition of evaluation is: If an only if $x \in V(I) = V(f_1, \ldots, f_m)$ can it satisfy (a small superset of) the m generators $f_1, \ldots, f_m$ in the randomized sequence $(f_1^*, \ldots, f_{m+k}^*)$.

The requirement for pattern matching is that the target object obj is represented in an "independently evaluable" way, meaning that the evaluation of the function can be reduced to a "summary" of the "independent" evaluations on the generators $obj_1, \cdots, obj_m$. For example, in the algebraic set membership obfuscation, an input $x \in V(I)$ if and only if x is a root of every generator $f_1, \ldots, f_m$ of the ideal I independently. The "summary" in this case is the logic AND.

For a counter example, consider ideal membership obfuscation. An ideal membership function with respect to an ideal $I = (f_1, \ldots, f_m)$ is a function $f_I(h)$ such that $f_I(h) = 1$ if and only if $h \in I$. In this case we cannot perform independent evaluation on the randomized sequence $(f_1^*, \ldots, f_{m+k}^*)$. This is because even if h is not in any single ideal (f_i), it can still be in I, as long as $h = g_1 f_1 + \cdots + g_m f_m$ is a "linear combination" of the generators $f_1, \ldots, f_m$ over $k[x_1, \ldots, x_n]$. This "combinatory evaluation" violates the requirement of "independently evaluable".

To perform independent evaluation, we also require the existence of an efficient membership testing algorithm. For example, given a metric ball $B(x, r)$, where x is the center of the metric ball and r is the radius of the ball, we require an efficient algorithm to compute the distance $d(x, y)$ so that we can decide if $y \in B(x, r)$.

Chapter 7

NEW VIEWS AND NEW PROBLEMS

This chapter discusses problems that are related to the algebraic set membership obfuscation problem. Note that the relations between most of the problems are not obvious if without the general obfuscation method. This reflects the meaning of the general method as well as the algebraic membership problem.

7.1 Vector Subspace and Lattice Membership

Two problems naturally follow from the algebraic set membership problem are the vector subspace membership and lattice membership problems.

Vector Subspace Membership Let K be a field and K^n be a vector space. A vector subspace membership function with respect to a vector subspace $V = (v_1, \ldots, v_m) \subseteq K^n$ is a Boolean function $f_V : K^n \to \{0, 1\}$ such that $f_V(v) = 1$ if and only if $v \in V$. Let σ be a collection of vector subspaces of V. The vector subspace membership function family with respect to σ is $\mathcal{F} = \{f_V\}_{V \in \sigma}$.

Its obfuscation is similar to the homogeneous linear case of the algebraic set membership problem. The only difference is that the algebraic set membership problem tests orthogonality against the bases, while the vector subspaces membership problem asks for linear combinations of the bases. Due to the "independently evaluable" requirement, we cannot directly encode the bases of the vector subspaces. The solution is to instead encode the basis $\{\hat{v}_1, \ldots, \hat{v}_{n-m}\}$ of its orthogonal complement. Then we can perform orthogonality testing: $\hat{v}_i \cdot x = 0$.

Lattice Membership Let $\mathbb{R}^m$ be the m-dimensional Euclidean space. A lattice membership function with respect to a lattice $L(B) = \{Bx : x \in \mathbb{Z}^n\}$ with the lattice basis $B = (b_1, \ldots, b_n) \in \mathbb{R}^{m \times n}$ is a Boolean function $f_L : \mathbb{Z}^n \to \{0, 1\}$ such that $f_L(z) = 1$ if and only if $z \in L(B)$. Let σ be a collection of lattices of $\mathbb{R}^m$. The lattice membership function family with respect to σ is $\mathcal{F} = \{f_L\}_{L \in \sigma}$.

Similar to algebraic set membership and vector subspace membership, lattice membership is another natural problem revealing the main idea of the obfuscation method.

At a high level, algebraic set membership and vector subspace membership perform "zero-testing", while lattice membership performs "integral-testing". The solution is to encode the dual basis $\{\hat{b}_1, \ldots, \hat{b}_n\}$ then perform "integral-testing": $\hat{b}_i \cdot x \in \mathbb{Z}$.

7.2 Subscheme and Submanifold Membership

Two kinds of objects that have deep relation to algebraic sets are subschemes and submanifolds.

Subscheme Membership With a sheaf a topological space can be attached with different datas (sets, groups, rings) hence being turned into richer kinds of topological spaces. Then we can talk about open subset membership obfuscation in the new topological space. A classic example is the spectrum $\mathrm{Spec}(R)$ of a commutative ring R with the Zariski topology being turned into a locally ringed space, which gives an affine scheme.

Let X be a scheme. A subscheme membership function with respect to a subscheme $U \subset X$ is a Boolean function $f_U : X \to \{0, 1\}$ such that $f_U(x) = 1$ if and only if $x \in U$. Let σ be a collection of subschemes of X. The subscheme membership function family with respect to σ is $\mathcal{F} = \{f_U\}_{U \in \sigma}$.

Submanifold Membership Let M be a projective complex manifold. An submanifold membership function with respect to a submanifold $U \subset M$ is a Boolean function $f_U : M \to \{0, 1\}$ such that $f_U(x) = 1$ if and only if $x \in U$. Let σ be a collection of submanifolds of M. The submanifold membership function family with respect to σ is $\mathcal{F} = \{f_U\}_{U \in \sigma}$.

Serre's GAGA principle allows us to map a projective complex manifold to an algebraic set. Hence a strategy to obfuscate submanifold membership is to firstly map it to an algebraic set then obfuscate algebraic set membership instead.

7.3 Metric Ball and Measure Box Membership

Metric Ball Membership An interesting and hard problem is the edit ball membership problem, or fuzzy edit distance matching problem. By edit distance we mean Leveshtein distance which involves three operations: insertion, deletion and substitution. This problem has wide applications in other disciplines such as gene editing in

biology and error correcting code construction in coding theory.

An edit ball membership function with respect to an edit ball $B(x, r) \in \{0,1\}^n$, where $x \in \{0,1\}^n$ is a point and $r < n \in \mathbb{N}$ is a threshold, is a boolean function $f_{B(x,r)} : \{0,1\}^n \to \{0,1\}$ such that $f_{B(x,r)}(y) = 1$ if and only if the edit distance between x and y is not greater than r, i.e., $y \in B(x, r)$. Let σ be a collection of edit balls in $\{0,1\}^n$. The edit ball membership function family with respect to σ is $\mathcal{F} = \{f_B\}_{B \in \sigma}$.

Measure Box Membership In contrast to metric "ball" membership, another problem of interest is measure "box" membership. An example is high dimensional (dim > 1) interval membership. Let $a_1, \ldots, a_n, b_1, \ldots, b_n \in \mathbb{Z}$ and $v = ([a_1, b_1], \ldots, [a_n, b_n])$. An n-dimensional interval function is a function $f_v : \mathbb{Z}^n \to \{0,1\}$ such that $f_v(x) = 1$ if and only if $x_i \in \{a_i, \ldots, b_i\}$ for all $i \in \{1, \ldots, n\}$. Let σ be a collection of n-dimensional intervals. An n-dimensional interval function family with respect to σ is $\mathcal{F} = \{f_v\}_{v \in \sigma}$.

Note that for a 1-dimensional interval it is not secure to represent it as the intersection of multiple intervals, because the number of all intervals induced by the representing intervals on a line grows very slowly and the attacker can easily find an accepting input by testing membership for all the induced intervals on the line. However, the maximum number of parts that N n-dimensional hypersphere divide a space is $\binom{N-1}{n} + \sum_{i=1}^{n} \binom{N}{i}$. Hence when the dimension is high enough we expect exponential blow up of induced intervals to hide the real interval.

7.4 Subgroup and Ideal Membership

Subgroup Membership Let G be a group. A subgroup membership function with respect to a subgroup $H \leq G$ is a function $f_H : G \to \{0,1\}$ such that $f_H(x) = 1$ if and only if $x \in H$. Let σ be a collection of subgroups of G. The subgroup membership function family with respect to σ is $\mathcal{F} = \{f_H\}_{H \in \sigma}$.

Ideal Membership A "dual" problem to the algebraic membership problem is the ideal membership problem. Let k be a field and $k[x_1, \ldots, x_n]$ be a polynomial ring. An ideal membership function with respect to an ideal $I \subset k[x_1, \ldots, x_n]$ is a function $f_I : k[x_1, \ldots, x_n] \to \{0,1\}$ such that $f_I(p) = 1$ if and only if $p \in I$. Let σ be a collection of ideals of $k[x_1, \ldots, x_n]$. The ideal membership function family with respect to σ is $\mathcal{F} = \{f_I\}_{I \in \sigma}$.

Note that there is a special kind of subgroups and ideals that are easy to find rep-

resentations. We call them prime-based objects. These include subgroups of integers, ideals of a univariate polynomial ring and ideals of the ring of integers. Let $n\mathbb{Z}_q$ be a subgroup of $\mathbb{Z}_q$ (the additive group modulo q). It can be represented by its supergroups as $n\mathbb{Z}_q = p_1\mathbb{Z}_q \cap \cdots \cap p_m\mathbb{Z}_q$. Let $I = (f_1 \cdots f_m)$ be an ideal of a univariate ring $k[x]$. It can be represented by its super ideals as $I = (f_1 \cdots f_m) = (f_1) \cdots (f_m) = (f_1) \cap \cdots \cap (f_m)$. Let $\mathfrak{a} = \mathfrak{p}_1 \cdots \mathfrak{p}_m = (p_1) \cdots (p_m)$ be an ideal of the ring of integers $\mathfrak{O}_{\mathbb{Q}} = \mathbb{Z}$. It can be represented by its super ideals as $\mathfrak{a} = (p_1) \cap \cdots \cap (p_m)$.

Another interesting example is Lie groups, which have wide applications in physics. People in physics mainly interested in a particular type of Lie groups, the matrix Lie groups, which can be defined using the general linear group $\mathrm{GL}_n \mathbb{C}$. A common case is the real general linear group $\mathrm{GL}_n \mathbb{R}$, which is the group of invertible $n \times n$ real matrices. A way to define Lie subgroups is by specifying the determinant of $n \times n$ matrices. Then to represent a Lie subgroup, one can just represent its determinant.

For example, let $a \in \mathbb{N}$ and let $\sigma_a = \{A \in \mathrm{GL}_n \mathbb{R} \mid \det(A) = a^i, i \in \mathbb{Z}\}$. Then σ_a is a Lie subgroup of $\mathrm{GL}_n \mathbb{R}$. Too see this, by the closed subgroup theorem, we only need to see that σ_a is a closed subgroup of $\mathrm{GL}_n \mathbb{R}$. To see σ_a is a subgroup of $\mathrm{GL}_n \mathbb{R}$, just to notice that the inverse A^{-1} of any matrix $A \in \sigma_a$ has determinant $\det(A^{-1}) = \det(A)^{-1} = a^{-i}$ hence is also in σ_a. By the two-step subgroup test, σ_a is subgroup of $\mathrm{GL}_n \mathbb{R}$. Again, to see σ_a is closed, let $\varphi : \mathbb{R}^{n^2} \to \mathbb{R}, A \mapsto \det(A)$ be the determinant function. Since φ is continuous, we only need to see that $\varphi(\sigma_a)$ is closed in $\mathbb{R}$. This is obvious since the elements in $\varphi(\sigma_a)$ are discrete points and its complement $\mathbb{R} \backslash \varphi(\sigma_a) = \cdots (1/a^i, 1/a^{i-1}) \cup \cdots \cup (a^i, a^{i+1}) \cdots$ is a union of open intervals hence is open.

Chapter 8

FUTURE WORK

We have given input-hiding obfuscation for the algebraic set membership problem in great generality. An open problem is to determine whether the obfuscator provides VBB security.

We have proposed a general obfuscation method for object membership functions. This method gives a new view to various open problems which were seemingly unrelated before. Due to lack of time, the development of complete solutions to these problems is a problem for future work.

It is also interesting to study the post-quantum hardness and applications of SPE, TSPE, MSPE, and ISPE.

BIBLIOGRAPHY

[AJ15] Prabhanjan Ananth and Abhishek Jain. "Indistinguishability Obfuscation from Compact Functional Encryption". In: *Advances in Cryptology – CRYPTO 2015*. Ed. by Rosario Gennaro and Matthew Robshaw. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 308–326. ISBN: 978-3-662-47989-6.

[BBCKPS14] Boaz Barak, Nir Bitansky, Ran Canetti, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. "Obfuscation for Evasive Functions". In: *Theory of Cryptography*. Ed. by Yehuda Lindell. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 26–51. ISBN: 978-3-642-54242-8.

[BCJJLMMMR19] James Bartusek, Brent Carmer, Abhishek Jain, Zhengzhong Jin, Tancrède Lepoint, Fermi Ma, Tal Malkin, Alex J. Malozemoff, and Mariana Raykova. "Public-Key Function-Private Hidden Vector Encryption (and More)". In: *Advances in Cryptology – ASIACRYPT 2019*. Ed. by Steven D. Galbraith and Shiho Moriai. Cham: Springer International Publishing, 2019, pp. 489–519. ISBN: 978-3-030-34618-8.

[BGIRSVY01] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. "On the (Im)possibility of Obfuscating Programs". In: *Advances in Cryptology — CRYPTO 2001*. Ed. by Joe Kilian. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 1–18. ISBN: 978-3-540-44647-7.

[BKMPRS18] Allison Bishop, Lucas Kowalczyk, Tal Malkin, Valerio Pastro, Mariana Raykova, and Kevin Shi. "A simple obfuscation scheme for pattern-matching with wildcards". In: *Annual International Cryptology Conference (CRYPTO)*. Springer. 2018, pp. 731–752.

[BLMZ19] James Bartusek, Tancrède Lepoint, Fermi Ma, and Mark Zhandry. "New Techniques for Obfuscating Conjunctions". In: *Advances in Cryptology – EUROCRYPT 2019*. Ed. by Yuval Ishai and Vincent Rijmen. Cham: Springer International Publishing, 2019, pp. 636–666. ISBN: 978-3-030-17659-4.

[BV15] N. Bitansky and V. Vaikuntanathan. "Indistinguishability Obfuscation from Functional Encryption". In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. 2015, pp. 171–190. DOI: 10.1109/FOCS.2015.20.

[BVWW16] Zvika Brakerski, Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. "Obfuscating Conjunctions Under Entropic Ring LWE". In: *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*. ITCS '16. Cambridge, Massachusetts, USA: ACM, 2016, pp. 147–156. ISBN: 978-1-4503-4057-1. DOI: 10.1145/2840728.2840764. URL: http://doi.acm.org/10.1145/2840728.2840764.

[BW19] Ward Beullens and Hoeteck Wee. "Obfuscating Simple Functionalities from Knowledge Assumptions". In: *Public-Key Cryptography – PKC 2019*. Ed. by Dongdai Lin and Kazue Sako. Cham: Springer International Publishing, 2019, pp. 254–283. ISBN: 978-3-030-17259-6.

[Can97] Ran Canetti. "Towards realizing random oracles: Hash functions that hide all partial information". In: *Advances in Cryptology — CRYPTO '97*. Ed. by Burton S. Kaliski. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 455–469. ISBN: 978-3-540-69528-8.

[CRV10] Ran Canetti, Guy N Rothblum, and Mayank Varia. "Obfuscation of hyperplane membership". In: *Theory of Cryptography Conference (TCC)*. Springer. 2010, pp. 72–89.

[Duj04] Andrej Dujella. "Continued fractions and RSA with small secret exponent". In: *arXiv preprint cs/0402052* (2004).

[Duj09] Andrej Dujella. "A variant of Wieners attack on RSA". In: *Computing* 85.1-2 (2009), pp. 77–83.

[FRS16] Benjamin Fuller, Leonid Reyzin, and Adam Smith. "When Are Fuzzy Extractors Possible?" In: *Advances in Cryptology – ASIACRYPT 2016*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 277–306. ISBN: 978-3-662-53887-6.

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. "Candidate Multilinear Maps from Ideal Lattices". In: *Advances in Cryptology – EUROCRYPT 2013*. Ed. by Thomas Johansson and Phong Q. Nguyen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 1–17. ISBN: 978-3-642-38348-9.

[GGHRSW13] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. "Candidate Indistinguishability Obfuscation and Functional Encryption for all Circuits". In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. 2013, pp. 40–49. DOI: 10.1109/FOCS.2013.13.

[GK05] Shafi Goldwasser and Yael Tauman Kalai. “On the impossibility of obfuscation with auxiliary input”. In: *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*. IEEE. 2005, pp. 553–562.

[GKW17] R. Goyal, V. Koppula, and B. Waters. “Lockable Obfuscation”. In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 2017, pp. 612–621.

[GR07] Shafi Goldwasser and Guy N Rothblum. “On best-possible obfuscation”. In: *Theory of Cryptography Conference (TCC)*. Springer. 2007, pp. 194–213.

[GZ19] Steven D. Galbraith and Lukas Zobernig. “Obfuscated Fuzzy Hamming Distance and Conjunctions from Subset Product Problems”. In: *Theory of Cryptography*. Ed. by Dennis Hofheinz and Alon Rosen. Cham: Springer International Publishing, 2019, pp. 81–110. ISBN: 978-3-030-36030-6.

[GZ20] Steven D. Galbraith and Lukas Zobernig. “Obfuscating Finite Automata”. In: *Selected Areas in Cryptography - SAC 2020 - 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers*. Ed. by Orr Dunkelman, Michael J. Jacobson Jr., and Colin O'Flynn. Vol. 12804. Lecture Notes in Computer Science. Springer, 2020, pp. 90–114. DOI: 10.1007/978-3-030-81652-0_4. URL: https://doi.org/10.1007/978-3-030-81652-0_4.

[Hur91] Adolf Hurwitz. “Über die angenäherte Darstellung der Irrationalzahlen durch rationale Brüche”. In: *Mathematische Annalen* 39.2 (1891), pp. 279–284.

[IN96] Russell Impagliazzo and Moni Naor. “Efficient cryptographic schemes provably as secure as subset sum”. In: *Journal of cryptology* 9.4 (1996), pp. 199–216.

[Lin16] Huijia Lin. “Indistinguishability obfuscation from constant-degree graded encoding schemes”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2016, pp. 28–57.

[LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. “Factoring polynomials with rational coefficients”. In: *Mathematische Annalen* 261.4 (1982), pp. 515–534.

[LM09] Vadim Lyubashevsky and Daniele Micciancio. “On Bounded Distance Decoding, Unique Shortest Vectors, and the Minimum Distance Problem”. In: *Advances in Cryptology - CRYPTO 2009*. Ed. by Shai Halevi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 577–594. ISBN: 978-3-642-03356-8.

[LPS04] Benjamin Lynn, Manoj Prabhakaran, and Amit Sahai. “Positive Results and Techniques for Obfuscation”. In: *Advances in Cryptology - EUROCRYPT 2004*. Ed. by Christian Cachin and Jan L. Camenisch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 20–39. ISBN: 978-3-540-24676-3.

[LSL13] Laura Luzzi, Damien Stehlé, and Cong Ling. “Decoding by embedding: correct decoding radius and DMT optimality”. In: *IEEE Transactions on Information Theory* 59.5 (2013), pp. 2960–2973.

[MM11] Daniele Micciancio and Petros Mol. “Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions”. In: *Annual Cryptology Conference*. Springer. 2011, pp. 465–484.

[Nec94] Vassiliy Ilyich Nechaev. “Complexity of a determinate algorithm for the discrete logarithm”. In: *Mathematical Notes* 55.2 (1994), pp. 165–172.

[Pol78] John M Pollard. “Monte Carlo methods for index computation (mod p)”. In: *Mathematics of computation* 32.143 (1978), pp. 918–924.

[Sha73] Daniel Shanks. “Five number-theoretic algorithms”. In: *Proceedings of the Second Manitoba Conference on Numerical Mathematics (Winnipeg), 1973*. 1973.

[SW14] Amit Sahai and Brent Waters. “How to use indistinguishability obfuscation: deniable encryption, and more”. In: *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. ACM. 2014, pp. 475–484.

[Wee05] Hoeteck Wee. “On obfuscating point functions”. In: *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*. ACM. 2005, pp. 523–532.

[WZ17] Daniel Wichs and Giorgos Zirdelis. “Obfuscating compute-and-compare programs under LWE”. In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 600–611.