Obfuscation Using Tensor Products

Craig Gentry and Charanjit S. Jutla
IBM T. J. Watson Research Center,
Yorktown Heights, NY 10598, USA

August 22, 2018

Abstract

We describe obfuscation schemes for matrix-product branching programs that are purely algebraic and employ matrix algebra and tensor algebra over a finite field. In contrast to the obfuscation schemes of Garg et al (SICOM 2016) which were based on multilinear maps, these schemes do not use noisy encodings. We prove that there is no efficient attack on our scheme based on re-linearization techniques of Kipnis-Shamir (CRYPTO 99) and its generalization called XL-methodology (Courtois et al, EC2000). We also provide analysis to claim that general Grobner-basis computation attacks will be inefficient. In a generic colored matrix model our construction leads to a virtual-black-box obfuscator for NC^1 circuits.

Contents

1 Introduction

Couple of year ago, Ye and Liu [YL16] posted a paper titled "Obfuscation without Multilinear Maps". The basic idea of that paper is to express binary circuits of depth d on n input bits as matrix-product programs using Barrington's theorem (the length of the matrix-product programs is exponential in d). Current obfuscation schemes (based on MMAPs [GGH$^+$16]) also use Kilian randomization of such a product program. However, if the underlying matrices are not securely encoded, then we have a problem. The new idea of [YL16] is to have a dynamic way of constructing new Kilian randomization matrices which depend on the full input x. This approach is called "dynamic fencing"' by Ye and Liu. Unfortunately, various attacks based on analyzing the underlying linear subspaces can be demonstrated on this scheme, and it is unlikely the scheme can be rescued without major changes. In this note, we will not go into these specific attacks as we will anyway highlight many relevant cryptanalytic techniques. The idea of dynamic-fencing was also discussed in [CEJvO02], although that work did not give any realizations of it.

To circumvent the possibly fatal flaws, our scheme moves to general linear transforms of the underlying Barrington matrices; the transforms we focus on are equivalent to simultaneously multiplying the underlying matrix by both its left and right randomizer matrix. The ring of $(k \times k)$ matrices over $\mathbb{F}$ is usually denoted by $M_k(\mathbb{F})$ (or M_k, for short). These matrices can also be viewed as k^2-dimensional vector spaces over $\mathbb{F}$, which we will denote by $\mathbb{F}^{k^2}$. The endomorphism-ring of any m-dimensional vector space V (i.e. the ring of homomorphisms from V to V) is itself the matrix ring M_m. Thus, we focus on the endomorphism-ring (or general linear transforms) M_{k^2} of the vector space $\mathbb{F}^{k^2}$. But, we will further focus on a subset of such transforms which can be represented as tensor product (Kronecker product) of two matrices from M_k, i.e. two $(k \times k)$ matrices. It turns out that these transforms, say $G \otimes H$, are exactly the transforms that result in multiplying a matrix by G on the left and H^T on the right. This subset of transforms form a multiplicative subgroup of the ring M_{k^2}, which is easily seen since matrix tensor product distributes over matrix multiplication.

Our construction requires each step i of the obfuscated program to consist of n stages, where n is the number of input bits. Each stage has a pair of such transforms associated with it. At "stage" j of step i of the program, let's say the pairs are $\langle G^i_{j,0} \otimes H^i_{j,0},\ G^i_{j,1} \otimes H^i_{j,1} \rangle$. The various stages in a step are used to build a dynamic-fence based on the input, and if the j-th input variable is zero, then the first element of the j-th pair is used, and likewise if the j-th input variable is one then the second element of the j-th pair is used. We must ensure that it should be impossible to obtain a mixed transform, say $G^i_{j,0} \otimes H^i_{j,1}$. But, if $G^i_{j,0} \otimes H^i_{j,0}$, $G^i_{j,1} \otimes H^i_{j,1}$ are given in the clear, then one easily obtains $G^i_{j,0}, H^i_{j,0}, G^i_{j,1}, H^i_{j,1}$ up-to scalar multiples, and hence one also obtains the mixed transforms (up-to scalar multiples). Hence, the construction only gives blinded versions of these tensor transforms. Thus, the (i,j)-th step tensor transforms are given masked by an invertible (linear transform) matrix $F_{i,j}$ from M_{k^4} (with $(G^i_{j,0} \otimes H^i_{j,0})$ and $(G^i_{j,1} \otimes H^i_{j,1})$ viewed as elements of $\mathbb{F}^{k^4}$).

We must however provide a gadget to take such a "masked dynamic-fence" computed for a step of the program, and apply the unmasked dynamic-fence to the Barrington matrix of the step[1]. Since the output of this gadget will *not* have a masking linear transform, a naive such gadget can be used

[1] Similarly, we also need to provide gadgets to multiply the tensor products of the different stages within a step.

to collect many outputs on related inputs, which then can allow computation of the linear transform $G^i_{j,0}{G^i_{j,1}}^{-1} \otimes H^i_{j,0}{H^i_{j,1}}^{-1}$, and hence also mixed gadgets. Thus, the gadget must compute an (invertible) quadratic form of the dynamic-fence before applying it to the Barrington matrix of the step. Since matrix multiplication is non-commutative, this then defeats the attack that computes the above linear transform.

It is well known that the space of homogeneous polynomials of degree d on a vector space V is isomorphic to the space of symmetric tensors of degree d on V. Thus, if we make the quadratic forms representing the gadgets homogeneous then these are completely specified by input-output behavior on $\binom{k^4+1}{2}$ linearly independent inputs. Since legitimate inputs to the gadgets are (masked) tensor-products, they belong to an affine algebraic variety, namely rank (at most) one matrices. A similar correspondence would then allow us to give a gadget that only works on the algebraic variety. Since our algebraic variety is of a particular special form (variety of an ideal with homogeneous multi-linear generators), we can indeed show such a correspondence and the gadget is completely specified (and easily computed) by input-output behavior on $\binom{k^2+1}{2}^2$ inputs of the (masked) tensor-product form. In fact, we can just specify the input-output behavior on legitimate inputs computed from the above described masked transforms associated with the different stages of a step. Thus, the gadget specifications can be considered black-box.

Security. Although the tensor transforms are hidden or masked by general linear transforms, say F, the Adversary gets many such samples masked by the same linear transform F. Calling one such sample $X_i = F \cdot \mathsf{vec}(A_i \otimes B_i)$, where vec is an operator that vectorizes a matrix, one can take tensor product of X_i with itself and then $X_i \otimes X_i$ has a non-trivial kernel, and belongs to what is called a symmetric tensor space. Many such samples yield many quadratic equations in F, the number of such equations equal to the product of the rank of the symmetric tensor space and the rank of a related skew-symmetric tensor space. A simple calculation shows that one gets about $k^{16}/16$ quadratic equations in k^8 variables representing F. The XL-methodology, a generalization of Kipnis-Shamir relinearlization technique [KS99, CKPS00], as well as what is known as Macaulay-matrix approach (see e.g. [BFS03, BFSS13]) are Grobner-Basis based cryptanalytic techniques for solving multi-variate polynomial equations. The basic idea of Grobner-basis based methodology is to multiply the original equations by monomials to get new equations (but in higher degree monomials)[2]. Since the monomials grow slower than the number of new equations obtained, at some point one expects more equations than monomials. However, the equations may not be linearly independent – if they were then one can solve for the monomials, and hence possibly the individual variables or at least their ratios.

We prove that this methodology fails for the $k^{16}/16$ quadratic equations above. In other words, we show that the number of linearly independent equations obtained at each grade (i.e. degree of the monomials) remains in large deficit of the number of monomials at that grade. This is not surprising as one can easily see that there are multiple solutions for F given the above equations, and F is at most determined modulo the tensor-product (multiplicative) subgroup.

Next, we add the equations obtained from the gadgets. Recall a gadget unmasks the linear-transform

[2]Grobner-basis methodology is more general than the XL-methodology, as in the former one can specify a monomial ordering and aim to eliminate low-ordered monomials.

F and outputs a quadratic form of the unmasked tensor-products. This yields further quadratic equations in variables representing F. However, as argued above the gadget is completely specified by k^8 inputs, and thus we only get that many new quadratic equations[3]. We then show that if one continues the XL-methodology and multiplies these additional equations also by monomials to get further equations, then the degree of the monomials must be at least k^2 before the total number of linearly independent equations approach the total number of monomials. This approach would then require computing an exponential number of additional equations. In particular, the number of equations required would be of the order of $(k^8)^{k^2} * k^{16}$.

While this rules out XL-methodology attacks, it is possible that a smaller set of monomials can yield equations such that there is a small-weight "codeword" spanned by the linear combination of the equations; possibility also exists of a not-so small "codeword" but with a linear factor. However, we have found no evidence suggesting that such anomalous codewords exist, and in Section 7 we give support for this claim. We consider proving lower bounds for weights of such codewords to be at par with proving lower bounds in proof-complexity [BP01] and computational complexity theory.

In Section 8 we also formulate and analyze a new *one-more masked tensor problem* and we believe progress on this hard problem can shed further light on the security of our scheme.

In the full version of the paper, we will show that in a generic colored matrix model [GGH$^+$16], our construction yields a virtual black-box (VBB) [BGI$^+$12] obfuscator for NC^1 circuits. The generic model disallows feeding the obfuscated program as input to another program, and hence the generic model proof of VBB-obfuscation does not contradict the known impossibility of VBB obfuscation of general circuits [BGI$^+$12] which does require self-feeding circuits. For this proof we employ another assumption called the pseudo-randomness of consistent matrix products.

Pseudo-randomness of Consistent Matrix-Products Assumption. For a security parameter s, and for any n, consider an oracle $\mathcal{O}$ which initializes itself by picking $s * n$ pairs of invertible matrices $\langle G_i^0, G_i^i \rangle$ from $M_k(\mathbb{F})$. On a query which is an assignment x from $[0..n-1]$ to $[0..1]$, the oracle responds with

$$\prod_{i' \in [0..s-1]} \prod_{i'' \in [0..n-1]} G_{i'*n+i''}^{x[i'']}.$$

The pseudo-randomness of consistent matrix products assumption states that no efficient adversary can distinguish with non-negligible probability the above oracle from another which just replies with random invertible matrices.

In other words, the consistent matrix product is a PRF on n-bit inputs with output that are invertible M_k matrices and with the key being the sequence of pairs of invertible matrices. The assumption is based on the fact that a Barrington matrix program of a small depth pseudo-random function would yield a similar consistent matrix product function.

[3]There may be additional multiplicative factor of k^2 equations, but then the equations also have an additional degree, e.g. in k^2 variables representing the pre-randomized Barrington matrix.

2 Preliminaries

Definition 1. Let vec be a vectorizing operator which takes an s by t matrix and represents it naturally as an $s * t$-length column vector (*by scanning row-wise*). Let $\mathsf{mat}_{s,t}$ denote the inverse operation, i.e. taking an $s * t$-length column vector and re-shaping it as an s by t matrix. Let $\Omega_{s,t}$ be the permutation such that $\Omega_{s,t} \cdot \mathsf{vec}(X) = \mathsf{vec}(X^{\mathsf{T}})$ for every $s \times t$-matrix X.

Lemma 1.

$$(A \otimes B^{\mathsf{T}}) \cdot \mathsf{vec}(C) = \mathsf{vec}(A \cdot C \cdot B), \tag{1}$$

for every A, C, B for which the product on the right is well-defined.

This is proven by simple algebraic manipulation.

Corollary 2. *For all M_t matrices A and B,*

$$\Omega_{t,t} \cdot (A \otimes B) \cdot \Omega_{t,t} \;=\; B \otimes A$$

Note $\Omega_{t,t}$ is symmetric, and since it is also a permutation, it implies that is is involutory, i.e. $\Omega_{t,t}\Omega_{t,t} = I$.

Similarly, for vectors we have,

Corollary 3. *For all t-vectors a and b,*

$$\Omega_{t,t} \cdot (a \otimes b) \;=\; b \otimes a$$

Lemma 4. *Define the permutation $\Pi_{s,t} = I^{s\times s} \otimes \Omega_{s,t} \otimes I^{t\times t}$. Consider any M_{st} matrix A, which can also be viewed as an s by s block matrix, with each block A_{ij} being an M_t matrix. Then, $\Pi_{s,t} \cdot \mathsf{vec}(A) = \mathsf{vec}(A')$, where A' is the matrix such that each M_t-blocks A_{ij} of A is replaced by $\mathsf{vec}(A_{ij})^{\mathsf{T}}$.*

The lemma is proved by applying the above fact twice. As an example application of the lemma, note

$$\Pi_{2,2} \cdot \mathsf{vec}\left(\begin{bmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{bmatrix}\right) = \mathsf{vec}\left(\begin{bmatrix} \mathsf{vec}(A_{00})^{\mathsf{T}} & \mathsf{vec}(A_{01})^{\mathsf{T}} \\ \mathsf{vec}(A_{10})^{\mathsf{T}} & \mathsf{vec}(A_{11})^{\mathsf{T}} \end{bmatrix}\right) = \begin{bmatrix} \mathsf{vec}(A_{00}) \\ \mathsf{vec}(A_{01}) \\ \mathsf{vec}(A_{10}) \\ \mathsf{vec}(A_{11}) \end{bmatrix} \tag{2}$$

where each A_{ij} is an M_2 matrix.

We really do not need the above lemma in this work; all we need is that the permutation $\Pi_{s,t}$ is defined to be the one that satisfies $\Pi_{s,t} \cdot \mathsf{vec}(A) \;=\; \mathsf{vec}(A')$.

Corollary 5. *For Y in M_s and Z in M_t, $\Pi_{s,t} \cdot \mathsf{vec}(Y \otimes Z)$ is same as $\mathsf{vec}(Y) \otimes \mathsf{vec}(Z)$.*

3 The Obfuscation Construction

We will assume that the circuit on n input-bits is given as a matrix branching program of length m:

$$P(x) = \prod_{i \in [0..m-1]} C_{i,x[l(i)]}$$

where $l(i) \in [1..n]$ is the index of the input bit used in the i-th branch. The matrices $C_{i,0/1}$ are in $\mathrm{SL}_k(\mathbb{F})$, where $k \geq 2$ is a security parameter.

3.1 Tensor-Product Encoding Scheme

We now describe an encoding scheme which is at the heart of the obfuscation construction. This encoding is just for one step, and in particular for one matrix $C_{i,0/1}$ above. Even though it is an encoding for one step, say i, which has input bit $l(i)$ associated with it, the encoding allows building dependence on all n variables. Alternative, and potentially more efficient (encoding) schemes will be discussed later (see Appendix A).

To aim for the required security, we employ two security parameters k and p. The parameter p is used to repeat the n variables p times. Assume that $p * n$ is a power of two, and let $d = \log(p * n)$.

The encoding scheme (with parameters $k, p, n, \mathtt{lmax}$) takes the following as input:

1. an $\mathrm{SL}_k(\mathbb{F})$ matrix C (called *center-piece*),
2. for each $j \in [0..p*n-1]$, a length $\mathsf{len}(j)$ ($\leq \mathtt{lmax}$), and a $\mathsf{len}(j)$-length list of *pairs* of M_k matrices, say $(\langle G_{j,0}, H_{j,0}\rangle, ..., \langle G_{j,\mathsf{len}(j)-1}, H_{j,\mathsf{len}(j)-1}\rangle)$. The matrices G, H will be generically referred to as *small Kilian matrices*.

In the immediate application, $\mathtt{lmax}$ will be two. We will also assume that len is consistent, i.e. for all j, $\mathsf{len}(j) = \mathsf{len}(j+n)$.

The output of the encoding is a collection of *masked leaf tensor-products*, *intermediate gadgets* and a *root gadget*.

We now describe the encoding procedure. Let $k' = k^4$.

- **Masked Leaf Tensor-Products:**
  1. Pick $p * n$ random and independent invertible $M_{k'}$-matrices $F_0, ..., F_{p*n-1}$.
  2. For each $j \in [0..p*n-1]$, for each $z \in [0..(\mathsf{len}(j)-1)]$, set $P_{j,z} = F_j \cdot \mathsf{vec}(G_{j,z} \otimes H_{j,z})$.

  The $P_{j,z}$ are the masked leaf tensor-products.
- **Intermediate Gadgets:** Building a binary tree from the masked leaves, the intermediate gadgets allow one to generate masked tensor-products which are a (quadratic) function of its two child nodes. Pick a random and independent invertible $M_{k'}$ matrix for each *internal* node of a binary tree of depth d, which we will refer to generically as F_{node}. The F matrix associated to the (2^d)

leaf nodes will just be the F_0 to F_{p*n-1} defined above. For each node, define $\mathsf{R1}(\text{node})$ to be the affine algebraic set[4]:

$$\mathsf{R1}(\text{node}) \;=\; \{v \in \mathbb{F}^{k^4} \mid \exists A, B \in M_k : v = F_{\text{node}} \cdot \mathsf{vec}(A \otimes B)\}$$

For each node in the binary tree, except for the leaf nodes and the root node, generate the following gadgets:

$$\begin{aligned}&\text{Gadget}_{\text{node}} : \mathsf{R1}(\text{left(node)}) \times \mathsf{R1}(\text{right(node)}) \to \mathsf{R1}(\text{node}) \text{ given by}\\ &\text{Gadget}_{\text{node}}(\mathbf{x}, \mathbf{y}) = F_{\text{node}} \cdot \mathsf{vec}(\,[\mathsf{mat}(F^{-1}_{\text{left(node)}} \cdot \mathbf{x}), \mathsf{mat}(F^{-1}_{\text{right(node)}} \cdot \mathbf{y})]\,)\end{aligned}$$

where left(node) and right(node) refer to the left and right children of the node respectively, and where the bracket $[\cdot, \cdot]$ is the commutator operator. The inverse of a matrix in the commutator can be replaced by adjugate of the matrix. Since we are aiming for quadratic functions, for $k > 2$, one can also consider replacing inverse of a matrix by transpose of the matrix, i.e. $[A, B]$ can then be considered to be $ABA^{\mathsf{T}}B^{\mathsf{T}}$. We will later describe how such gadgets can be efficiently implemented. A naive implementation can extend the gadget to be defined over all $\mathbb{F}^{k^2} \times \mathbb{F}^{k^2}$, in which case it can be given as coefficients of the resulting degree two polynomial in x and y (total degree four).

- **Root Gadget:** Let us call the root node of the binary tree "root". The root gadget is defined to be the following:

$$\begin{aligned}&\text{Gadget}_{\text{root}} : \mathsf{R1}(\text{left(root)}) \times \mathsf{R1}(\text{right(root)}) \to M_k \text{ given by}\\ &\text{Gadget}_{\text{root}}(\mathbf{x}, \mathbf{y}) = \mathsf{mat}(\,[\mathsf{mat}(F^{-1}_{\text{left(root)}} \cdot \mathbf{x}), \mathsf{mat}(F^{-1}_{\text{right(root)}} \cdot \mathbf{y})] \cdot \mathsf{vec}(C)\,)\end{aligned}$$

  Again, we will describe later how such a gadget can be implemented efficiently.

3.1.1 Using the Tensor-Product Gadgets

Recall that the encoding scheme produces as output a collection of *masked leaf tensor-products*, *intermediate gadgets* and a *root gadget*.

A legitimate use of these gadgets is to pick a (not necessarily consistent) assignment $x[j]$ for each $j \in [0..p*n-1]$ ($x[j] \in [0..(\mathsf{len}(j)-1)]$), and then pick the corresponding masked leaf tensor-product $P_{j,x[j]}$. Next, use the intermediate gadgets to build a masked tensor-product for each internal node of the binary tree. If to each node (including the leaves) we associate the computed (or picked) masked tensor-product by $v(\text{node})$, we then have:

1. For all $j \in [0..p*n-1]$, $v(j) = F_j \cdot \mathsf{vec}(G_{j,x[j]} \otimes H_{j,x[j]})$.

[4] It is not difficult to see that such sets are algebraic as these are exactly the v with the property that $(F_{node} \cdot \Pi_{k,k})^{-1} v$ are vectorized rank one (or zero) M_{k^2} matrices: Note, $F \cdot \mathsf{vec}(A \otimes B)$ is same as $F \cdot \Pi_{k,k} \cdot (\mathsf{vec}(A) \otimes \mathsf{vec}(B))$. Thus, $(F \cdot \Pi_{k,k})^{-1} v$ is $\mathsf{vec}A \otimes \mathsf{vec}B$ which can be re-shaped into $\mathsf{vec}A\,\mathsf{vec}B^{\mathsf{T}}$, a rank one matrix. for more details, See Section 4.2.

2. For each internal node, $v(\text{node}) = F_{\text{node}} \cdot \mathsf{vec}(\ [\mathsf{mat}(F^{-1}_{\text{left(node)}} \cdot v(\text{left(node)})), \mathsf{mat}(F^{-1}_{\text{right(node)}} \cdot v(\text{right(node)}))]\ )$.

Finally, the root gadget is used to compute the (unmasked) commutator of the children tensor-products applied to $\mathsf{vec}(C)$. Since the commutator is itself a tensor product, by Lemma 1 this leads to a dynamic-fence applied to C, with a function of the G matrices (picked according to assignment x) multiplied on the left and a function of the matched H matrices applied on the right. We will refer to this value as **sandwiched-centerpiece**.

The full evaluation of the obfuscated program is described after we describe the full obfuscated program itself. The consistency of the assignment x is enforced in the full evaluation.

3.2 The Obfuscated Program

We will assume that the circuit on n-input bits is given as a matrix branching program of length m:

$$P(x) = \prod_{i \in [0..m-1]} C_{i,x[l(i)]}$$

where $l(i) \in [1..n]$ is the index of the input bit used in the i-th branch. The matrices $C_{i,0/1}$ are in $\mathrm{SL}_k(\mathbb{F})$. We will also assume that if the circuit evaluates to false on an input x then $P(x)$ is the identity matrix, and if the circuit evaluates to true on an input x then $P(x)$ is not a scalar matrix. Usual application of Barrington's theorem would easily result in such a matrix product program.

In a first reading, the following pre-processing step can be skipped, and the reader can move directly to the beginning of the next paragraph.

Pre-Processing: The branching program above is first pre-processed to obtain another branching program (on n-bits input) by the usual Kilianization procedure. In other words, choose m matrices F_i from $\mathrm{SL}_k(\mathbb{F})$ ($i \in [0..m-1]$), and let the new matrices for step i be $C'_{i,b} = F_i \cdot C_{i,b} \cdot F^{-1}_{i+1}$, where $(i+1)$ is computed modulo m. The size of the program remains the same, as well as the labeling l. Next, each step of this pre-processed branching program is further pre-processed as follows. A new security parameter (natural number) p' is chosen, which can be zero. Each step is expanded into $1 + p' * n$ steps as follows: the new matrices for the $m' = m * (1 + p' * n)$ total steps will be called C'. The new labeling will be called l'. For each $i \in [0..m-1]$,

- Pick $p' * n$ random matrices $S_0, ..., S_{p'*n-1}$ from $\mathrm{SL}_k(\mathbb{F})$. Let $S_{p'*n}$ be the identity matrix.
- Let $C'_{i*(1+p'*n),b} = C_{i,b} * S_0$ (for $b \in [0..1]$). Let $l'(i * (1 + p' * n)) = l(i)$.
- for each $\kappa \in [0..p'-1]$, for each $j' \in [0..n-1]$, and each $b \in [0..1]$, let $C'_{i*(1+p'*n)+1+\kappa*n+j',b} = S^{-1}_{\kappa*n+j'} * S_{\kappa*n+j'+1}$. Let $l'(i * (1 + p' * n) + 1 + \kappa * n + j') = j'$.

That finishes the description of the pre-processing. For ease of presentation, we will refer to C', l' and m' as the new C, l and m below, i.e. the pre-processing is assumed implicit.

We now define the various components of the obfuscated program. This consists primarily of two encodings corresponding to each step ($i \in [0..m-1]$) of the matrix program, obtained by using the above

tensor-product encoding scheme. To obtain "dynamic fences", for each of the m steps, the encoding scheme is provided, as required, with pairs of $p * n$ left (small) Kilian matrices, and pairs of $p * n$ right (small) Kilian matrices. In each such pair $(G_{j,0}, G_{j,1})$ $(j \in [0..p * n - 1])$ of left Kilian matrices, one matrix is meant to be used if the j'-th variable has assignment zero, and the other if the assignment is one, where $j' = j \pmod n$. Similarly, for the right Kilian matrix pairs $(H_{j,0}, H_{j,1})$. Of course, these Kilian matrices are chosen so that the right Kilian matrices for one program step, say step i, are canceled by left Kilian matrices of the next program step $i + 1$. We also want to assure that if for the left Kilian matrix for variable j', the assignment zero is chosen, i.e. $G_{j',0}$ is chosen, then for the right Kilian matrix for variable j', the assignment zero is chosen as well, i.e. $H_{j',0}$ is chosen. To this end, the encoding scheme is called with pairs $(G_{j,0}, H_{j,0})$, and $(G_{j,1}, H_{j,1})$, which enforces this consistent assignment on left and right. Thus, the encoding scheme is called with a list of pairs of length two, i.e. $((G_{j,0}, H_{j,0}), (G_{j,1}, H_{j,1}))$, except when j' is same as $l'(i)$, where i is the step number. When j' is same as $l'(i)$, the encoding scheme is provided with (a list of) only one pair $(G_{j,b}, H_{j,b})$, where b is the centerpiece being encoded for this step. Recall, each step has two centerpieces, corresponding to matrices $C_{i,b}$ of the program (or $C'_{i,b}$ of the pre-processed program). The two centerpieces and its encodings obtained using the encoding scheme are considered as two different *tracks*. They share the same left and right Kilian matrices above, i.e. the G and the H matrices. But otherwise, their encodings are obtained independently.

We now give a formal description of the obfuscated program. Given a branching program on n variables of length m, specified by $\mathrm{SL}_k(\mathbb{F})$ matrices $C_{i,b}$ for each $i \in [0..m-1]$ and $b \in \{0,1\}$, and a step-to-variable mapping l from $[0..m-1]$ to $[0..n-1]$, the obfuscated program is obtained as follows:

- For each $i \in [0..m-1]$, each $j \in [0..p*n-1]$, and for each $z \in [0..1]$, generate a random (left small Kilian) invertible M_k matrix $G_{i,j,z}$.
- For each $i \in [0..m-1]$ do
  1. Define a mapping len as $\mathsf{len}(j) = 2$ if $j \neq l(i) \pmod n$, otherwise $\mathsf{len}(j) = 1$.
  2. For each $z \in [0..1]$, each $j \in [0..p*n-1]$, let (the right small Kilian matrix) $H_{i,j,z} = G^{-\mathsf{T}}_{i+1,j,z}$, where subscript addition is modulo m.
  3. For each track $b \in [0..1]$, and $j \in [0..p*n-1]$, prepare a list of pairs, $L_{i,b,j}$, of length $\mathsf{len}(j)$, as follows:
     - if $\mathsf{len}(j) = 1$, then $L_{i,b,j}[0] = \langle G_{i,j,b}, H_{i,j,b} \rangle$ (i.e. only the variable assignment consistent with track b is provided),
     - else, $L_{i,b,j}[z] = \langle G_{i,j,z}, H_{i,j,z} \rangle$, for $z \in [0..1]$,

     Note, that the lists (corresponding to the two tracks) are identical except when $j = l(i) \pmod n$. Next, invoke the tensor-product encoding scheme with input:
     - the $\mathrm{SL}_k(\mathbb{F})$ matrix $C_{i,b}$,
     - for each $j \in [0..p*n-1]$, the length $\mathsf{len}(j)$, and for each $j \in [0..p*n-1]$ the $\mathsf{len}(j)$-length lists $L_{i,b,j}$.

     to obtain as output a collection of *masked leaf tensor-products*, *intermediate gadgets* and a *root gadget*. Call these $P^{i,b}_{j,z}$, $\mathrm{Gadget}^{i,b}_{\mathrm{node}}$ and $\mathrm{Gadget}^{i,b}_{\mathrm{root}}$ resp.

4. When $j = l(i) \pmod n$, i.e. $\mathsf{len}(j) = 1$, since only one masked leaf tensor-product is returned, for convenience we set the second matrix to be same. Thus, set $P^{i,b}_{j,1} = P^{i,b}_{j,0}$ for $j = l(i) \pmod n$.

The **obfuscated program** consists of the masked leaf tensor-products $P^{i,b}_{j,z}$, intermediate gadgets $\text{Gadget}^{i,b}_{\text{node}}$ and root gadgets $\text{Gadget}^{i,b}_{\text{root}}$ ($i \in [0..m-1]$, $b \in [0..1]$, $j \in [0..2 * p * n - 1]$, $z \in [0..1]$ and node belonging to internal nodes of a binary tree of depth d).

3.3 Evaluating the Obfuscated Program

During an honest execution of the obfuscated program, an assignment, say $x[j']$, is chosen for each of the n variables, i.e. $j' \in [0..n-1]$.

Next, for each step of the pre-processed program, i.e. for $i \in [0..m-1]$, use the $x[l(i)]$-th track encoding, i.e. $\text{Gadget}^{i,x[l(i)]}_{\text{node}}$ and root gadgets $\text{Gadget}^{i,x[l(i)]}_{\text{root}}$ on the masked leaf tensor-products from that track and picking the $x[j]$-th choice for each leaf, i.e. $P^{i,x[l(i)]}_{j,x[j]}$ ($j \in [0..p * n]$) to compute the sandwiched-centerpiece as described in Section 3.1.1. Note that the centerpiece inside the sandwiched-centerpiece is $C_{i,x[l(i)]}$. For each i, we will refer to the so obtained sandwiched-centerpiece as $S^{i,x}$.

Compute $T = S^{0,x} \cdot S^{1,x} \cdot ... \cdot S^{m-1,x}$.

Output false if T is a scalar matrix (i.e. scalar multiple of the identity matrix), else output true.

It is an easy exercise to see the correctness of the evaluation of the obfuscated program.

4 Tensor Algebra

In this section we introduce the basics of tensor algebra. We refer the reader to books on tensor algebra such as [BG68, Gre78] for a more extensive treatment. In the following we will consider vector spaces over the field $\mathbb{F}$. If a basis of a vector space V of dimension t is fixed, the elements of V will be identified with t-column vector of coefficients with respect to the basis. If V is a linear sub-space of $\mathbb{F}^m$, then the basis can be represented as set of m-column vectors of $m \times t$ matrix. The *dual space* V^* of V is the linear space of linear functionals on V, i.e. linear functions $V \to \mathbb{F}$. It is well known that V^* is isomorphic to V and $V^{**} = V$. If a basis of V^* is fixed, then a linear functional will be identified with a t-row vector of coefficients with respect to this basis. Thus, if f is a linear functional in V^* and $x \in V$, then $f(x) = f \cdot x$.

A tensor over V is a $\mathbb{F}$-valued multilinear-function with all variables in V or V^*. The number of variables from V^* are called the contravariant degree, and the number of variables from V are called covariant degree of the tensor. The (linear) space of multilinear functions on, say, $V^* \times V \times V$ will be denoted by $V \otimes V^* \otimes V^*$ or $T^1_2(V)$. The linear space will also be called the space of tensors of type $(1, 2)$.

The *tensor product* of tensor a of type (r, s) and tensor b of type (p, q) is a tensor $a \otimes b$ of type

$(r+p, s+q)$ and defines a function on $(V^*)^{r+p} \times V^{s+q}$, given by

$$a \otimes b(\tau^i, ..., \tau^{r+p}, v_1, ..., v_{s+q}) = a(\tau^i, ..., \tau^r, v_1, ..., v_s) b(\tau^{r+1}, ..., \tau^{r+p}, v_{s+1}, ...v_{s+q}).$$

It is not difficult to see the tensor product distributes over addition.

Note that T_0^1, i.e. linear functionals from V^* to $\mathbb{F}$, is just V. In this work, we will focus mostly on tensor products of such tensors (or just column vectors aka contravariant vectors). Since a $p \times p'$ matrix can be represented as a $p * p'$-columns vector (e.g. using the vec operator), one can also consider tensor product of matrices A (of dimension $r \times s$) and B (of dimension $p \times q$) by $A \otimes B = \mathsf{mat}(\Pi \cdot (\mathsf{vec}(A) \otimes \mathsf{vec}(B)))$, where Π is an appropriate permutation as defined in Section 2. This then yields the familiar definition of matrix tensor product (also known as Kronecker product).

A basis for T_0^n is easily seen to be given by tensors $\{e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_n}\}$, where $\{e_i\}$ is a basis of V. Thus with respect to such a basis, a tensor $a \in T_0^n$ is given by components $a^{i_1,...,i_n}$ (represented as a $t * n$ vector). We will conveniently write $e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_n}$ as $e_{i_1,i_2,...,i_n}$.

4.1 Symmetric Tensors

A tensor a (in T_0^n) is **symmetric** in the p-th and q-th indices if a (i.e. the corresponding multilinear function $(V^*)^n \to \mathbb{F}$) is symmetric in the p-th and q-th variables. A tensor is *symmetric* if it is symmetric in every pair of indices. The symmetric tensors of type $(n, 0)$ form a subspace SYM^n of T_0^n. A symmetric tensor a is given by the components $a^{i_1,...,i_n}$ such that $i_1 \leq \cdots \leq i_n$, and the other components are given by symmetry. A convenient basis is given by the symmetrization operator, the basis consisting of following tensors: for every unordered choice of $i_1, ..., i_n$,

$$\sum_\sigma e_{\sigma(i_1,...,i_n)},$$

where σ is a permutation of n letters (henceforth called n-permutation).[5] Thus, the dimension of SYM^n is $\binom{t+n-1}{n}$.

It is well known that $\mathsf{SYM}^n(V)$ is isomorphic to homogeneous polynomials of degree n in t variables in field $\mathbb{F}$ (see e.g. Section 9.13 [Gre78]). By fixing a basis for V, any element of V is then specified by t elements in $\mathbb{F}$, and hence we also have an isomorphism to degree n homogeneous polynomial functions on V.

A tensor a (in T_0^n) is **skew-symmetric** in the p-th and q-th indices if a is skew-symmetric in the p-th and q-th variables. Again, it is not difficult to see that a skew-symmetric tensor a of type $(n, 0)$ is given by components $a^{i_1,...,i_n}$ such that $i_1 < i_2 < \cdots < i_n$. Thus the dimension of the space is $\binom{t}{n}$. A convenient basis is now given by anti-symmetrization, i.e. the basis consisting of following tensors: for every unordered choice of distinct $i_1, ..., i_n$,

$$\sum_\sigma \mathsf{sgn}\, \sigma \cdot e_{\sigma(i_1,...,i_n)}.$$

[5] Every permutation σ on n letters determines a permutation on $[k]^n$ (also called σ) and defined by $\sigma(i_1, i_2, ..., i_n) = (i_{\sigma^{-1}(1)}, i_{\sigma^{-1}(2)}, ..., i_{\sigma^{-1}(n)})$.

It is also not difficult to see that the skew-symmetric tensor space is the orthogonal complement of SYM^2, and for this reason we will call it coSYM^2, its basis being the co-kernel of the basis of SYM^2 (viewed as a $t \times \binom{t+n-1}{n}$ matrix)[6]

4.2 Rank One Matrices

Square matrices in M_t that have rank one (or zero) are exactly the matrices ab^T, where a and b are T_0^1 tensors (or simply, t-columns). Such a rank one matrix can also be viewed as $\mathsf{mat}(a \otimes b)$. Thus, we will identify rank one matrices with rank one T_0^2 tensors over $\mathbb{F}^t$. Rank one matrices also form an algebraic set, as these are exactly the matrices for which every 2×2 minor of the matrix has determinant zero, thus giving an algebraic representation. We now show an alternate way of proving that rank one matrices form an algebraic set (hint: defining determinant in terms of exterior product of columns).

Let $\mathsf{SYM}^{(1,3),(2,4)}$ (to be later renamed $\mathsf{SYM}^{2,2}$) be the space of tensors of degree four (i.e. subspace of T_0^4) for which the first and third indices are symmetric, and the second and fourth degrees are symmetric. Taking a cue from the basis of SYM^n above, it is the space spanned by all tensors in T_0^4 of the form $a \otimes b \otimes a \otimes b$.

It will be convenient to denote e_{i_1,j_1,i_2,j_2} by $\epsilon(\langle i_1, i_2\rangle, \langle j_1, j_2\rangle)$. As for symmetric tensors, it is not difficult to show that the linear sub-space spanned by $a \otimes b \otimes a \otimes b$ has as basis the following vectors (for every unordered choice of (i_1, i_2) and every unordered choice of (j_1, j_2)):

$$\sum_{\sigma_1} \sum_{\sigma_2} \epsilon(\sigma_1\langle i_1, i_2\rangle, \sigma_2\langle j_1, j_2\rangle) \tag{3}$$

where σ_1 and σ_2 are 2-permutations. This implies that the number of basis vectors is exactly the square of the number of monomials of degree two in t variables, which is $\binom{t+1}{2}^2$. Alternatively, one can view this space as tensor product of two symmetric tensor spaces of degree two. For this reason we will now just call it $\mathsf{SYM}^{2,2}$. Its orthogonal complement, which will have as its basis the co-kernel of the matrix representing the above basis of $\mathsf{SYM}^{2,2}$, will be denoted by $\mathsf{coSYM}^{2,2}$.

In the following, we will let $\mathbf{x} \otimes \mathbf{x}$ denote the Kronecker-product of a vector of t variables $\mathbf{x}$ with itself. In other words, $(\mathbf{x} \otimes \mathbf{x})_{\langle i,j\rangle} = x_i x_j$. We will also denote the vector of all degree two monomials in $\mathbf{x}$ by $\mathrm{Mon}^2(\mathbf{x})$, and in particular $(\mathrm{Mon}^2(\mathbf{x}))_{\langle i,j\rangle} = x_i x_j$ with $i \leq j$. Then it is easy to check that if we identify SYM^2 with its basis matrix $\{\sum_\sigma e_{\sigma(i,j)}\}_{i \leq j}$, then

$$\mathbf{x} \otimes \mathbf{x} = \mathsf{SYM}^2(\mathbb{F}^t) \cdot \mathrm{Mon}^2(\mathbf{x}). \tag{4}$$

Similarly, with $\mathbf{y}$ being an additional t-vector of variables, we have

$$\mathbf{x} \otimes \mathbf{y} \otimes \mathbf{x} \otimes \mathbf{y} = \mathsf{SYM}^{2,2} \cdot (\mathrm{Mon}^2(\mathbf{x}) \otimes \mathrm{Mon}^2(\mathbf{y})), \tag{5}$$

where $\mathsf{SYM}^{2,2}$ is identified with the matrix representing the basis (3) above with indices ranging ($i_1 \leq i_2, j_1 \leq j_2$).

[6] An $m \times n$ matrix T is also a linear transformation from $\mathbb{F}^n$ to $\mathbb{F}^m$. Thus, the orthogonal complement of the vector space with basis T is same as the cokernel of the linear transformation T, i.e. $\mathbb{F}^m/\mathrm{Im}(T)$.

Lemma 6. *The set of rank one (or zero) matrices is the subset of $\mathbb{F}^{t^2}$ which is the zero-set of the quadratic polynomials in (t^2-vector of variables) $\mathbf{x}$ given by $(\mathsf{coSYM}^{2,2}(\mathbb{F}^t))^{\mathsf{T}}(\mathbf{x}\otimes\mathbf{x})$.*

Proof. By definition of $\mathsf{coSYM}^{2,2}$ and the above characterization of rank one matrices as of the form $a\otimes b$, it is clear that rank one matrices are in the zero-set of these polynomials.

We next show that if for some t^2-vector c it is the case that $c\otimes c$ is in span of $\mathsf{SYM}^{2,2}$, then c must be a T_0^2 tensor, i.e. of the form $a\otimes b$. This is same as checking that if cc^{T} is in span of tensor product of symmetric matrices then c itself must be a tensor-product. To this end, divide c into blocks of vectors, each of size t. Call these blocks $c_1,...c_t$. Then the (i,j)-th M_t-block of cc^{T} is $c_ic_j^{\mathsf{T}}$. Next, we argue that every M_{t^2} matrix m that is in the span of tensor product of symmetric matrices has the property that each M_t-block of m is itself a symmetric matrix. This follows because symmetric matrices form an additive sub-group of M_t.

Hence, for each $i,j\in[1..t]$, the block $c_ic_j^{\mathsf{T}}$ is symmetric. For each vector c_i, we will let $c_{i,k}$ denote its k-th element. Next, we claim that if $c_ic_j^{\mathsf{T}}$ is symmetric then either (a) c_j is a scalar multiple of c_i, or (b) c_i is zero. So, suppose c_i is not zero. Let, k be the smallest index such that $c_{i,k}$ be non-zero. Then, for all $l\in[1..t]$, by the symmetry property of $c_ic_j^{\mathsf{T}}$, we have $c_{j,l}=(c_{j,k}/c_{i,k})*c_{i,l}$, and hence c_j is a scalar multiple of c_i.

Next, let s be the smallest index in $[1..t]$, such that c_s is non-zero. If there is no such s, then c is zero, and c is trivially a T_0^2 tensor. Otherwise, for all $s'\in[1..t]$, $c_{s'}$ is a scalar multiple of c_s by above claim. Thus, c is a T_0^2 tensor. □

Since, $\mathsf{coSYM}^2(\mathbb{F}^{t^2})$ is a subspace of $\mathsf{coSYM}^{2,2}(\mathbb{F}^t)$, and $(\mathsf{coSYM}^2)^{\mathsf{T}}(\mathbf{x}\otimes\mathbf{x})$ is identically zero, we are really looking at the zero-set of $(\mathsf{coSYM}^{2,2}/\mathsf{coSYM}^2)^{\mathsf{T}}(\mathbf{x}\otimes\mathbf{x})$.

4.3 Quadratic Polynomials Modulo Ideal of Rank One Matrices

For any invertible M_{t^2} matrix F, define

$$\mathsf{R1}(F) \;=\; \{v\in\mathbb{F}^{t^2} \mid \exists a,b\in T_0^1(\mathbb{F}^t) : v = F\cdot(a\otimes b)\}$$

Lemma 7. *Rank of span $Mon^2(\mathsf{R1}(F))$ is $\binom{t+1}{2}^2$.*

Proof. We will first prove rigorously that the span of $\mathrm{Mon}^2(\mathsf{R1}(F))$ is isomorphic to the span of $\{y\otimes y | y\in R1(F)\}$. When we identify a vector space V with its basis B, we drop the term "span" when saying $V=$ span B or $V\simeq$ span B, and just write $V=B$ or $V\simeq B$ resp.

Recall, for a t^2-vector of variables $\mathbf{x}$, $(\mathbf{x}\otimes\mathbf{x}) \;=\; \mathsf{SYM}^2(\mathbb{F}^{t^2})\cdot\mathrm{Mon}^2(\mathbf{x})$. Since, $\mathrm{Mon}^2(\mathbf{x})$ is just a subset of $\mathbf{x}\otimes\mathbf{x}$, there is another matrix INVSYM^2 such that $\mathrm{Mon}^2(\mathbf{x}) \;=\; \mathsf{INVSYM}^2\cdot(\mathbf{x}\otimes\mathbf{x})$. Thus, $(\mathsf{SYM}^2\cdot\mathsf{INVSYM}^2 - I^{t^2})\cdot\mathsf{SYM}^2 = 0$, as $\mathsf{SYM}^2(\mathbb{F}^{t^2})$ is same as span of $(c\otimes c)$, where c is an arbitrary t^2-vector. We also have

$$\text{span } \mathrm{Mon}^2(\mathsf{R1}(F)) \;=\; \mathsf{INVSYM}^2\cdot(F\otimes F)\cdot\mathsf{SYM}^{2,2}(\mathbb{F}^t). \tag{6}$$

Since, $(F \otimes F) \cdot \mathsf{SYM}^{2,2}(\mathbb{F}^t)$ is subspace of $\mathsf{SYM}^2(\mathbb{F}^{t^2})$, we also have

$$(\mathsf{SYM}^2 \cdot \mathsf{INVSYM}^2 - I^{t^2}) \cdot (F \otimes F) \cdot \mathsf{SYM}^{2,2}(\mathbb{F}^t) = 0. \tag{7}$$

Since matrix SYM^2 is full-ranked, i.e. has no right kernel, we have from (7),

$$\mathsf{INVSYM}^2 \cdot (F \otimes F) \cdot \mathsf{SYM}^{2,2}(\mathbb{F}^t) \simeq (F \otimes F) \cdot \mathsf{SYM}^{2,2}(\mathbb{F}^t). \tag{8}$$

Thus, by (6)

$$\text{span } \text{Mon}^2(\mathsf{R1}(F)) \simeq (F \otimes F) \cdot \mathsf{SYM}^{2,2}(\mathbb{F}^t) = \text{span } \{y \otimes y \mid y \in R1(F)\}. \tag{9}$$

Thus, rank of span $\text{Mon}^2(\mathsf{R1}(F))$ is $\binom{t+1}{2}^2$. □

It is worth noting that by the very definition of $\mathsf{INVSYM}^2 \cdot (\mathbf{x} \otimes \mathbf{x})$, the transform INVSYM^2 picks $\binom{t^2+1}{2}$ rows from $(\mathbf{x} \otimes \mathbf{x})$, and we have shown that when picking the same rows from $\mathsf{SYM}^{2,2}$ it preserves the rank of $\mathsf{SYM}^{2,2}$.

In the following, given a full-ranked $(\binom{t^2+1}{2} \times \binom{t+1}{2}^2)$-matrix M, we would like to identify a subset of indices from $[1..\binom{t^2+1}{2}]$, called "full", such that the sub-matrix of M consisting of rows with indices from this subset is invertible.

Theorem 8.

(a) The vector space of homogeneous quadratic functions on $\mathsf{R1}(F)$ *is isomorphic to the* $\mathbb{F}$*-linear span of* $\{y \otimes y \mid y \in R1(F)\}$.

(b) Any homogeneous quadratic function $f(\mathbf{x})$ *defined on* $\mathsf{R1}(F)$ *and given by* $\vec{f}^{\,\mathsf{T}} \cdot Mon^2(\mathbf{x})$*, is with high probability equivalent to the function*

$$\{f(X_i)\}_i \cdot \{Mon^2(X_i)_{full}\}_i^{-1} \cdot Mon^2(\mathbf{x})_{full},$$

where $\{X_i\}_i$ *are a set of* $\binom{t+1}{2}^2$ *random and independent samples from* $\mathsf{R1}(F)$*, and the subscript full denotes any subsequence of indices of size* $\binom{t+1}{2}^2$ *such that the resulting matrix* $\{Mon^2(X_i)_{full}\}_i$ *is invertible.*

It is well known that the space of degree d homogeneous polynomials over a vector-space V is isomorphic to degree d symmetric tensors of V. However, here we are claiming the same to be true for an algebraic set $\mathsf{R1}(F)$. This however is not surprising, since it can be checked that the ideal generator $(\mathsf{coSYM}^{2,2})^{\mathsf{T}}(\mathbf{x} \otimes \mathbf{x})$ of ideal of $\mathsf{R1}(F)$ is already a Grobner basis for any natural monomial ordering. Moreover the ideal generators are homogeneous quadratic polynomials, and hence using Macaulay's theorem (see e.g. Theorem 15.3 in [Eis95]) the above theorem follows. However, for sake of completeness, we give a direct proof here.

Proof. Now, we prove (a) and (b).

(a) A homogeneous quadratic function f on $\mathbb{F}^t$ is given by polynomial $\vec{f}^{\,\mathsf{T}} \cdot \mathrm{Mon}^2(\mathbf{x})$, where $\vec{f}$ is the coefficients of f. Also, by lemma 6, $\mathsf{R1}(F)$ is the zero-set of $(\mathsf{coSYM}^{2,2})^{\mathsf{T}} \cdot (F^{-1} \otimes F^{-1}) \cdot (\mathbf{x} \otimes \mathbf{x})$ or alternatively the zero set of $(\mathsf{coSYM}^{2,2})^{\mathsf{T}} \cdot (F^{-1} \otimes F^{-1}) \cdot \mathsf{SYM}^2 \cdot \mathrm{Mon}^2(\mathbf{x})$.

So, consider the kernel of $(\mathsf{coSYM}^{2,2})^{\mathsf{T}} \cdot (F^{-1} \otimes F^{-1}) \cdot \mathsf{SYM}^2$. We will call it W. By lemma 6 it follows that span of $\mathrm{Mon}^2(\mathsf{R1}(F))$ is a subspace of W. However, a quick check of the ranks shows that the spaces are identical: we will just show that rank of W is also $\binom{t+1}{2}^2$. First, note that $\mathsf{coSYM}^2(\mathbb{F}^{t^2})$ is a subspace of $\mathsf{coSYM}^{2,2}(\mathbb{F}^t)$. Thus,

$$\begin{aligned}\operatorname{rank} W &= \operatorname{rank} \mathsf{SYM}^2(\mathbb{F}^{t^2}) - (\operatorname{rank} \mathsf{coSYM}^{2,2}(\mathbb{F}^t) - \operatorname{rank} \mathsf{coSYM}^2(\mathbb{F}^{t^2})) \\ &= t^4 - \operatorname{rank} \mathsf{coSYM}^{2,2}(\mathbb{F}^t) \\ &= \operatorname{rank} \mathsf{SYM}^{2,2}(\mathbb{F}^t).\end{aligned}$$

Now, consider a reduced "column-echelon" basis of W which can be obtained from $\mathrm{Mon}^2(\mathsf{R1}(F))$ or $\mathsf{INVSYM}^2 \cdot (F \otimes F) \cdot \mathsf{SYM}^{2,2}$, by multiplying on the right by inverse of $(\mathsf{INVSYM}^2 \cdot (F \otimes F) \cdot \mathsf{SYM}^{2,2})_{\text{full}}$. Call this reduced-$W$. Since, $\mathbf{x} \in \mathsf{R1}(F)$ iff $(\mathsf{coSYM}^{2,2})^{\mathsf{T}} \cdot (F^{-1} \otimes F^{-1}) \cdot \mathsf{SYM}^2 \cdot \mathrm{Mon}^2(\mathbf{x})$ is zero, this implies that $\mathbf{x} \in \mathsf{R1}(F)$ iff $\mathrm{Mon}^2(\mathbf{x}) = \text{reduced-}W \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}}$ (see Appendix B for a detailed proof). Thus, the homogeneous quadratic function f on $\mathbb{F}^t$ given by polynomial $\vec{f}^{\,\mathsf{T}} \cdot \mathrm{Mon}^2(\mathbf{x})$, can be restricted to be well-defined on $\mathsf{R1}(F)$ by the function

$$\vec{f}^{\,\mathsf{T}} \cdot \text{reduced-}W \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}}.$$

Thus, the vector space of homogeneous quadratic functions on $\mathsf{R1}(F)$ is spanned by

$$\begin{aligned}&\text{reduced-}W \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}} \\ &= \text{reduced-}(\mathsf{INVSYM}^2 \cdot (F \otimes F) \cdot \mathsf{SYM}^{2,2}) \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}}.\end{aligned}$$

We already showed that linear span of $\{y \otimes y \mid y \in R1(F)\}$ is isomorphic to $\mathrm{Mon}^2(\mathsf{R1}(F))$ or $\mathsf{INVSYM}^2 \cdot (F \otimes F) \cdot \mathsf{SYM}^{2,2}$, which then completes the proof.

(b) It is not difficult to see that with high probability

$$\{X_i \otimes X_i\}_{i=1..\text{rank}\mathsf{SYM}^{2,2}},$$

with X_i chosen randomly and independently from $\mathsf{R1}(F)$, form a basis of span of $\{y \otimes y \mid y \in R1(F)\}$. Then, by the above isomorphism (i.e. INVSYM^2), it follows that with high probability $\{\mathrm{Mon}^2(X_i)\}_{i=1..\text{rank}\mathsf{SYM}^{2,2}}$ also form a basis of space spanned by $\mathrm{Mon}^2(\mathsf{R1}(F))$. Thus, we have with high probability,

$$\begin{aligned}&\vec{f}^{\,\mathsf{T}} \cdot \text{reduced-}W \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}} \\ =&\vec{f}^{\,\mathsf{T}} \cdot \text{reduced-}\{\mathrm{Mon}^2(X_i)\}_{i=1..\text{rank}\mathsf{SYM}^{2,2}} \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}} \\ =&\vec{f}^{\,\mathsf{T}} \cdot \{\mathrm{Mon}^2(X_i)\}_i \cdot \{\mathrm{Mon}^2(X_i)_{\text{full}}\}_i^{-1} \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}} \\ =&\{f(X_i)\}_i \cdot \{\mathrm{Mon}^2(X_i)_{\text{full}}\}_i^{-1} \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}}.\end{aligned}$$

□

The proof of theorem 8 easily extends to give the following theorem.

Theorem 9. *Any homogeneous function $f(\mathbf{x}, \mathbf{y})$ defined on $\mathsf{R1}(F^1) \times \mathsf{R1}(F^2)$, quadratic in both $\mathbf{x}$ and $\mathbf{y}$, and given by $\vec{f}^{\,\mathsf{T}} \cdot (Mon^2(\mathbf{x}) \otimes Mon^2(\mathbf{y}))$, is with high probability equivalent to the function*

$$\{f(X_i, Y_j)\}_{i,j} \cdot (\{Mon^2(X_i)_{xfull}\}_i^{-1} \otimes \{Mon^2(Y_j)_{yfull}\}_j^{-1}) \cdot (Mon^2(\mathbf{x})_{xfull} \otimes Mon^2(\mathbf{y})_{yfull}),$$

where $\{X_i\}_i$ are a set of $\binom{t+1}{2}^2$ random and independent samples from $\mathsf{R1}(F^1)$ and $\{Y_j\}_j$ are a set of $\binom{t+1}{2}^2$ random and independent samples from $\mathsf{R1}(F^2)$, and subscript "xfull" denotes any subsequence of indices of size $\binom{t+1}{2}^2$ such that the resulting matrix $\{Mon^2(X_i)_{xfull}\}_i$ is invertible, subscript "yfull" denotes any subsequence of indices of size $\binom{t+1}{2}^2$ such that the resulting matrix $\{Mon^2(Y_j)_{yfull}\}_j$ is invertible.

5 Implementation of Gadgets

In Section 3.1, we promised to give an efficient implementation of the gadgets (i.e. the intermediate gadgets and the root gadget).

Recall, the intermediate gadget is:

$$\begin{aligned}&\text{Gadget}_{\text{node}} : \mathsf{R1}(\text{left(node)}) \times \mathsf{R1}(\text{right(node)}) \to \mathsf{R1}(\text{node}) \text{ given by}\\ &\text{Gadget}_{\text{node}}(\mathbf{x}, \mathbf{y}) = F_{\text{node}} \cdot \mathsf{vec}(\ [\mathsf{mat}(F^{-1}_{\text{left(node)}} \cdot \mathbf{x}), \mathsf{mat}(F^{-1}_{\text{right(node)}} \cdot \mathbf{y})]\ )\end{aligned}$$

and the root gadget is:

$$\begin{aligned}&\text{Gadget}_{\text{root}} : \mathsf{R1}(\text{left(root)}) \times \mathsf{R1}(\text{right(root)}) \to M_k \text{ given by}\\ &\text{Gadget}_{\text{root}}(\mathbf{x}, \mathbf{y}) = \mathsf{mat}(\ [\mathsf{mat}(F^{-1}_{\text{left(root)}} \cdot \mathbf{x}), \mathsf{mat}(F^{-1}_{\text{right(root)}} \cdot \mathbf{y})] \cdot \mathsf{vec}(C)\ )\end{aligned}$$

Theorem 9 shows that the above gadgets can be given by giving out polynomially many input-output samples of the evaluations of the Gadgets. In particular, we can give evaluations on legitimate inputs, and before a certain depth there are enough legitimate inputs such that their input-output behavior completely determines the Gadgets on all inputs in R1. Near the leaves there may not be enough legitimate inputs, but then the Gadget can be given on just those (few) legitimate inputs.

6 Ranks of Certain Symmetric-Tensor Spaces

Since by corollary 5, $F \cdot \mathsf{vec}(A \otimes B)$ is same as $F \cdot \Pi_{k,k} \cdot (\mathsf{vec}(A) \otimes \mathsf{vec}(B))$, in this section we will just focus on vectors $F \cdot (a \otimes b)$, where a and b are k^2-vectors (and let F stand for $F \cdot \Pi_{k,k}$). In this section t should be thought of as $t = k^2$, where k is the security parameter used in Section 3.

Now, given arbitrarily many samples $X_i = F \cdot (a_i \otimes b_i)$, it follows by Marcus-Moyls Theorem [MM59] (or see [Fei03]) that modulo tensor-product multiplicative-subgroup, any G satisfying (for all i) $X_i = \mathsf{G} \cdot (x_i \otimes y_i)$ for some x_i and y_i is either F or $F\Omega$. Further, note that the same a_i maybe used in the neighbouring step hidden under a different F, say F^1. Thus, the adversary may also be given $Y_i = F^1 \cdot (a_i \otimes c_i)$, for some arbitrary c_i. We next show that given arbitrarily many X_i and Y_i, G (modulo tensor-product subgroup) must be F (and G^1 must be F^1).

Note, for all i, $F^{-1}X_i$ and $(F^1)^{-1}Y_i$ have the same columns (namely a_i) up to scaling (when re-shaped as matrices, using the operator mat). Let P and Q be "alternatives" to F and F^1 (modulo tensor-product subgroup) – that is, $P^{-1}X_i$ and $Q^{-1}Y_i$ are all rank-1 matrices (i.e. of the form xy^{T}). Firstly, by Marcus-Moyls Theorem, P is either F or $F\Omega$ (modulo tensor-product subgroup). Suppose (toward a contradiction) that P is $F\Omega$ (modulo tensor-product subgroup). Then $P^{-1}X_i = (R \otimes S)\Omega(a_i \otimes b_i) = (R \otimes S) \cdot (b_i \otimes a_i) = (Rb_i \otimes Sa_i)$.

Thus, the columns of $P^{-1}X_i$ (when re-shaped as a matrix) are multiples of Rb_i. It is impossible for the columns of $Q^{-1}Y_i$ to be multiples of some Rb_i, since these samples are independent of b_i, which leads to a contradiction.

Despite this information-theoretic result, we now argue that it is computationally-hard for the Adversary to obtain F (up to tensor-products and scalars). We will then extend the argument to the case where the Adversary is given additional equations, e.g. from the commutator gadgets, or if the Adversary tries to fix a representative modulo the tensor-product subgroup.

6.1 Rank of Macaulay Matrices

For a given non-singular M_{t^2}-matrix F, consider the sub-space $\mathcal{E}$ of $\mathbb{F}^{t^2}$ generated by vectors $F \cdot (a \otimes b)$, where each a and b are arbitrary t-vectors. While $\mathcal{E}$ as a linear sub-space of $\mathbb{F}^{t^2}$ has rank t^2, one does get non-trivial information about F by taking tensor product of $F \cdot (a \otimes b)$ with itself.

Given arbitrary samples of $F \cdot (a_i \otimes b_i)$, called X_i, $i \in [0..\mathrm{rank}\,\mathsf{SYM}^{2,2} - 1]$, the collection $\{X_i \otimes X_i\}_i$ can be written as $(F \otimes F) \cdot \mathsf{SYM}^{2,2} \cdot R$, for some matrix R in $M_{\mathrm{rank}\,\mathsf{SYM}^{2,2}}$. Given random samples, with high probability R is non-singular. Thus, by introducing free variables G for F^{-1}, we then have the following equations in the t^4 variables of G:

$$\mathsf{coSYM}^{2,2^{\mathsf{T}}} \cdot (\mathsf{G}F \otimes \mathsf{G}F) \cdot \mathsf{SYM}^{2,2} \ = 0$$

At face value, this would mean $\binom{t+1}{2}^2 \cdot (t^4 - \binom{t+1}{2}^2)$, or $O(t^8)$ quadratic equations in t^4 variables. However, since $\mathsf{G}F \otimes \mathsf{G}F$ have repeated entries, the number of linearly independent (quadratic) equations, i.e. linear equations in $\mathrm{Mon}^2(\mathsf{vec}(\mathsf{G}F))$, is smaller. From now on-wards, we will write $\mathsf{G}F$ as H. To get the correct number of independent equations in $\mathrm{Mon}^2(\mathsf{vec}(\mathsf{H}))$, it is convenient to write the above equations using Lemma 1 as

$$(\mathsf{coSYM}^{2,2^{\mathsf{T}}} \otimes \mathsf{SYM}^{2,2^{\mathsf{T}}}) \cdot \mathsf{vec}(\mathsf{H} \otimes \mathsf{H}) \ = 0 \tag{10}$$

Instead of using the relation (4), we will use a slightly different representation in this section. It will also be convenient to write an index in $[0..t^8 - 1]$ as $(\langle i, j\rangle, \langle k, l\rangle)$, with each coordinate in $[0..t^2 - 1]$,

such that $\mathsf{vec}(\mathsf{H} \otimes \mathsf{H})_{\langle i,j\rangle,\langle k,l\rangle}$ refers to the $(i * t^2 + j)$-th row and $(k * t^2 + l)$-th column of $\mathsf{H} \otimes \mathsf{H}$. This, of course, is just $\mathsf{H}_{i,k}\mathsf{H}_{j,l}$. Hence, we have equality relations

$$\mathsf{vec}(\mathsf{H} \otimes \mathsf{H})_{\langle i,j\rangle,\langle k,l\rangle} = \mathsf{vec}(\mathsf{H} \otimes \mathsf{H})_{\langle j,i\rangle,\langle l,k\rangle}$$

In other words, for every permutation π of pairs of integers in $[0..t^2 - 1]$,

$$\mathsf{vec}(\mathsf{H} \otimes \mathsf{H})_{\langle i,j\rangle,\langle k,l\rangle} = \mathsf{vec}(\mathsf{H} \otimes \mathsf{H})_{\pi\langle i,j\rangle,\pi\langle k,l\rangle} \tag{11}$$

We will use the basis of $\mathsf{SYM}^{2,2}$ characterized in Section 4.2. We will also use other notation from that section. Recall, the linear sub-space spanned by $a \otimes b \otimes a \otimes b$ has as basis the following vectors:

$$\sum_{\sigma_1}\sum_{\sigma_2} \epsilon(\sigma_1\langle i_1, i_2\rangle, \sigma_2\langle j_1, j_2\rangle) \tag{12}$$

where σ_1 and σ_2 are 2-permutations. More generally, the space spanned by $(a \otimes b)^{\otimes n}$, has as basis vectors:

$$\sum_{\sigma_1}\sum_{\sigma_2} \epsilon(\sigma_1\langle i_1, i_2, ..., i_n\rangle, \sigma_2\langle j_1, j_2, ..., j_n\rangle)$$

where the permutations are now n-permutations. Again, the number of such basis vectors is exactly the square of the number of monomials of degree n in t variables, which is $\binom{t+n-1}{n}^2$. It will be useful to note that the basis vectors are closed under an n^2-permutation $\sigma \otimes \sigma$ applied to all terms, i.e. a n-permutation σ applied to the i-indices and the same σ simultaneously applied to the j-indices. While one can consider generalizations of the attack where instead of tensoring X_i with itself, one does an n-fold tensor product of X_i with itself, we will focus on the base case. This more general case can be handled similarly, and yields more or less the same result.

The orthogonal complement of the above space can also be characterized similarly, although it is slightly more complicated as the sign of the permutation gets involved[7]. However, since we are only seeking the number of independent equations in (10), i.e. in $\mathrm{Mon}^2(\mathsf{vec}(\mathsf{H}))$, one need only consider the following closure. For a vector c, and a set of operations Γ, the additive closure of c under Γ is defined as $\sum_{\gamma\in\Gamma} \gamma(c)$.

Thus, for each row of $\mathsf{coSYM}^{2,2^\mathsf{T}} \otimes \mathsf{SYM}^{2,2^\mathsf{T}}$, given that $\mathsf{vec}(\mathsf{H} \otimes \mathsf{H})$ satisfies equality relations (11), we must take its additive closure under $\pi \otimes \pi$, for every 2-permutation π of pair of integers in $[0..t^2 - 1]$. Each such 2-permutation π can be viewed as a 2-permutation $\sigma \otimes \sigma$ (by first pairing indices as in going from e to ϵ as in Section 4.2). However, as remarked earlier, $\mathsf{SYM}^{2,2^\mathsf{T}}$ is already closed under such permutations π (viewed as $\sigma \otimes \sigma$). The basis of the cokernel however is not closed under $\sigma \otimes \sigma$, and indeed most elements of the cokernel are trivialized (i.e. become zero) by this closure operation, except when all indices i_1, i_2 are distinct, as well as j_1, j_2 are distinct. In such cases, the closure of the cokernel

[7] The orthogonal complement $\mathsf{coSYM}^{2,2}$ is generated by the following vectors: For each i_1, i_2, j_1, j_2, $\sum_\sigma \mathsf{sgn}\,\sigma \cdot \epsilon(\sigma\langle i_1, i_2\rangle, j_1, j_2)$ and $\sum_\sigma \mathsf{sgn}\,\sigma \cdot \epsilon(i_1, i_2, \sigma\langle j_1, j_2\rangle)$. It is easy to check that only $(4-1)\cdot\binom{t}{2}^2 + 2\cdot t\cdot\binom{t}{2}$ of these are linearly independent.

(under π) is characterized by basis vectors :

$$\sum_{\sigma_1}\sum_{\sigma_2} \mathsf{sgn}\,\sigma_1 \cdot \mathsf{sgn}\,\sigma_2 \cdot \epsilon(\sigma_1\langle i_1, i_2\rangle, \sigma_2\langle j_1, j_2\rangle). \tag{13}$$

Thus, the total number of linearly independent (quadratic) equations[8] in (10) is $\binom{t}{2}^2 \cdot \binom{t+1}{2}^2$. This is still $O(t^8)$ (quadratic) equations in t^4 variables.

The usual XL-methodology at this point would require one to multiply each of these equations with degree n monomials of H to get extra equations of degree $n+2$ in H-variables (the matrix comprising of coefficients of all the equations in terms of the degree $(n+2)$-monomials is called the **Macaulay matrix**). The number of monomials of degree n in H variables is $\binom{t^4+n-1}{n}$, and of degree $n+2$ is $\binom{t^4+n+2-1}{n+2}$. The total number of equations obtained by multiplying by degree n monomials is then

$$\binom{t}{2}^2 \cdot \binom{t+1}{2}^2 \cdot \binom{t^4+n-1}{n}.$$

It is not difficult to see that there is a small constant n, for which this number of equations exceeds $\binom{t^4+n+2-1}{n+2}$. The only problem is that these equations are not linearly independent (in the degree $(n+2)$ monomials). Actually, the problem of determining the number of linearly independent equations in grade $(n+2)$ (i.e. degree $(n+2)$ monomials of a graded polynomial ring) is the subject of the famous Hilbert function. For any ideal I, which in this case is the set of quadratic equations above, the Hilbert function of the ideal I, denoted H_I, evaluated at $(n+2)$, is exactly the number of linearly independent generators of the ideal restricted to the $(n+2)$-th grade (the independence is defined over the underlying field or ring of the polynomial ring). In general, it is inaccurate to just multiply the number of original equations and by the number of monomials of degree n to get $H_I(n+2)$ as the following analysis shows: Let the ideal I be generated by N quadratic equations $\{f_i\}$. We will denote the polynomial ring $R[\mathsf{G}]$ by RG. Then the natural map from the free RG-module $\mathsf{RG}^N = \bigoplus \mathsf{RG}\epsilon_i$ to RG given by $\epsilon_i \to f_i$ has a kernel. Included in this kernel are expressions $f_i\epsilon_j - f_j\epsilon_i$. In other words, equations $f_i = 0$ and $f_j = 0$ yield dependent equations in grade 4 when multiplied by polynomials f_j and f_i resp. These are some of the obvious generators of the kernel, and in general there could be more complicated generators. Of course, these kernel generators can themselves be dependent, and we need to take a further kernel of

[8] Note that although the number of distinct monomials is $\binom{t^4+2-1}{2}$, we are considering the equations represented as a subspace of $\mathbb{F}^{t^8}$. Alternatively, we are looking at the quotient space

$$(\mathsf{coSYM}^{2,2} \otimes \mathsf{SYM}^{2,2})/\text{cokernel } \Pi_{t^2,t^2} \cdot \mathsf{SYM}^2(\mathbb{F}^{t^2}).$$

Strictly speaking, cokernel $\Pi_{t^2,t^2} \cdot \mathsf{SYM}^2(\mathbb{F}^{t^2})$ is not a subspace of $(\mathsf{coSYM}^{2,2} \otimes \mathsf{SYM}^{2,2})$, so we must take their intersection before dividing. It can be shown that this space is same as

$$((\mathsf{coSYM}^{2,2}/\mathsf{coSYM}^2(\mathbb{F}^{t^2})) \otimes \mathsf{SYM}^{2,2},$$

where SYM^2 is the sub-space of symmetric tensors, i.e. t^2-vectors tensor-ed with itself. It is not difficult to see that $(\mathsf{coSYM}^{2,2}/\mathsf{coSYM}^2(\mathbb{F}^{t^2}))$ has as its basis $\sum_{\sigma_2}\sum_{\sigma} \mathsf{sgn}\,\sigma \cdot \epsilon(\sigma_2\sigma\langle i_1, i_2\rangle, \sigma_2\langle j_1, j_2\rangle)$, which is non-trivial only if $i_1 \neq i_2$ and $j_1 \neq j_2$.

these kernel generators. This is the subject matter of *Hilbert's syzygy theorem* (see e.g. [Eis95]) which shows that this process of *free resolution* of the original ideal terminates after a finite number of steps. Then, by inclusion-exclusion principle one can obtain the Hilbert function.

For general ideals, it is a difficult problem to accurately get the Hilbert function at different grades. However, we show that for our problem we can exactly upper bound the number of independent generators of the ideal at each grade.

First note that it suffices to consider additional equations obtained by multiplying by monomials in H instead of G. Next, note that the generators of $\mathsf{coSYM}^{2,2}$ as given in (13) can be written as

$$\sum_{\sigma_1}\sum_{\sigma_2} \mathsf{sgn}\,\sigma_2 \cdot \epsilon(\sigma_1\langle i_1, i_2\rangle, \sigma_1\sigma_2\langle j_1, j_2\rangle). \tag{14}$$

Now, consider multiplying an original equation with coefficients

$$\sum_{\sigma_1,\sigma_2} \mathsf{sgn}\,\sigma_2 \cdot \epsilon(\sigma_1\langle i_1, i_2\rangle, \sigma_1\sigma_2\langle j_1, j_2\rangle)^{\mathsf{T}} \cdot \sum_{\sigma_1',\sigma_2'} \epsilon(\sigma_1'\langle i_1', i_2'\rangle, \sigma_2'\langle j_1', j_2'\rangle)^{\mathsf{T}} \tag{15}$$

by a (degree one) monomial $\mathsf{H}_{\langle i_3,j_3\rangle,\langle i_3',j_3'\rangle}$. Since monomials satisfy symmetry relations, the resulting equation is closed under a 3-permutation $\bar{\sigma}$ to give coefficients

$$\begin{aligned}
&\sum_{\bar{\sigma}}\sum_{\sigma_1,\sigma_2} \mathsf{sgn}\,\sigma_2 \cdot \epsilon(\bar{\sigma}(\sigma_1\langle i_1, i_2\rangle, i_3), \bar{\sigma}(\sigma_1\sigma_2\langle j_1, j_2\rangle, j_3))^{\mathsf{T}} \cdot \sum_{\sigma_1',\sigma_2'} \epsilon(\bar{\sigma}(\sigma_1'\langle i_1', i_2'\rangle, i_3'), \bar{\sigma}(\sigma_2'\langle j_1', j_2'\rangle, j_3'))^{\mathsf{T}} \\
=&\sum_{\bar{\sigma}}\sum_{\sigma_2} \mathsf{sgn}\,\sigma_2 \cdot \epsilon(\bar{\sigma}(i_1, i_2, i_3), \bar{\sigma}(\sigma_2\langle j_1, j_2\rangle, j_3))^{\mathsf{T}} \cdot \sum_{\sigma_1',\sigma_2'} \epsilon(\bar{\sigma}(\sigma_1'\langle i_1', i_2'\rangle, i_3'), \bar{\sigma}(\sigma_2'\langle j_1', j_2'\rangle, j_3'))^{\mathsf{T}}
\end{aligned} \tag{16}$$

When multiplying an original equation (15) by a degree $(n-2)$ monomial in H, we similarly get an equation with coefficients

$$\sum_{\bar{\sigma}}\sum_{\sigma_2} \mathsf{sgn}\,\sigma_2 \cdot \epsilon(\bar{\sigma}(i_1, i_2, i_3, .., i_n), \bar{\sigma}(\sigma_2\langle j_1, j_2\rangle, j_3, .., j_n))^{\mathsf{T}} \cdot \sum_{\sigma_1',\sigma_2'} \epsilon(\bar{\sigma}(\sigma_1'\langle i_1', i_2'\rangle, i_3', .., i_n'), \bar{\sigma}(\sigma_2'\langle j_1', j_2'\rangle, j_3', .., j_n'))^{\mathsf{T}} \tag{17}$$

Note that the expression

$$\sum_{\bar{\sigma}} \epsilon(\bar{\sigma}(i_1, i_2, i_3, .., i_n), \bar{\sigma}(j_1, j_2, j_3, .., j_n))^{\mathsf{T}} \cdot \epsilon(\bar{\sigma}(i_1', i_2', i_3', .., i_n'), \bar{\sigma}(j_1', j_2', j_3', .., j_n'))^{\mathsf{T}}$$

corresponds to a monomial $\mathsf{H}_{\langle i_1,j_1\rangle,\langle i_1',j_1'\rangle}\cdot\ldots\cdot\mathsf{H}_{\langle i_n,j_n\rangle,\langle i_n',j_n'\rangle}$ From now on, when we talk about a monomial in H, we mean the equivalence class modulo the commutation relations, as in xy and yx are in the same equivalence class. Similarly,

$$\sum_{\bar{\sigma}}\sum_{\sigma_2} \mathsf{sgn}\,\sigma_2 \cdot \epsilon(\bar{\sigma}(i_1, i_2, i_3, .., i_n), \bar{\sigma}(\sigma_2\langle j_1, j_2\rangle, j_3, .., j_n))^{\mathsf{T}} \cdot \epsilon(\bar{\sigma}(i_1', i_2', i_3', .., i_n'), \bar{\sigma}(j_1', j_2', j_3', .., j_n'))^{\mathsf{T}}$$

corresponds to $\mathsf{H}_{\langle i_1,j_1\rangle,\langle i'_1,j'_1\rangle}\cdot\mathsf{H}_{\langle i_2,j_2\rangle,\langle i'_2,j'_2\rangle}\cdot....\mathsf{H}_{\langle i_n,j_n\rangle,\langle i'_n,j'_n\rangle} - \mathsf{H}_{\langle i_1,j_2\rangle,\langle i'_1,j'_1\rangle}\cdot\mathsf{H}_{\langle i_2,j_1\rangle,\langle i'_2,j'_2\rangle}\cdot....\mathsf{H}_{\langle i_n,j_n\rangle,\langle i'_n,j'_n\rangle}$. Such terms will be referred to as j-transposed anti-symmetric terms. Now, note that all the equations of the form (17) are linear combinations of j-transposed anti-symmetric terms. We now give a basis which spans all the equations of the form (17).

We first partition the set of degree n monomials in H into disjoints sets as follows: fix $\langle i_1, i'_1, j'_1\rangle$, $\langle i_2, i'_2, j'_2\rangle$,...,$\langle i_n, i'_n, j'_n\rangle$ Note that each of these indices, i.e. $i_1, ..., i_n, i'_1, ..., i'_n, j'_1, ..j'_n$ takes value in $[1..t]$ and possibly with repetition. Next, pick $j_1, ..., j_n$, again possibly with repetition. Consider all possible permutations of the picked $j_1, ..., j_n$ and fill the j-slots in tuples above to get a set of monomials. This set is then characterized by unordered multi-set $(\langle i_1, i'_1, j'_1\rangle, ..., \langle i_n, i'_n, j'_n\rangle)$ and unordered multi-set $(j_1, j_2, ..., j_n)$, and consists of all (equivalence classes of) monomials $\mathsf{H}_{\langle i_1,j^*_1\rangle,\langle i'_1,j'_1\rangle}\cdot ...\cdot \mathsf{H}_{\langle i_n,j^*_n\rangle,\langle i'_n,j'_n\rangle}$, where $j^*_1, ..., j^*_n$ is a permutation of $j_1, ..., j_n$. We will refer to this set of monomials as $\text{Mon}^n(\langle i_1, i'_1, j'_1\rangle, ..., \langle i_n, i'_n, j'_n\rangle, j_1, j_2, ..., j_n)$.

Each monomial belongs to at least one of these sets, and by construction the sets are pair-wise disjoint. Next, consider each such set of monomials above, and consider its *transposition graph*, the nodes of which are the monomials in the set. Note that each node is characterized by a different ordering of multi-set $(j_1, ..., j_n)$. Two nodes have an edge if their characterizing orderings are a transposition of each other, i.e. one ordering is obtained from the other by exchanging two *different* entries in $(j_1, ..., j_n)$. It is well-known that the transposition graph of orderings of a multi-set is Hamiltonian [Cha73]. Further, as noted above, the equations of the form (17) are linear combinations of j-transposed anti-symmetric terms, which in this new formulation means the anti-symmetric term involving *two monomials from the same set* and with an edge between the two monomials in the transposition-graph of the set. A basis of such j-transpose anti-symmetric terms is then easily obtained by focusing on each set, and in particular focusing on a Hamiltonian path in its transposition-graph: for each edge in the Hamiltonian path introduce a basis element consisting of a j-transposed anti-symmetric term. Any transposition T in this set can be obtained by following the the sub-path of the Hamiltonian path starting from one node and ending in the other. A linear combination of the basis elements corresponding to the edges of the sub-path also yields the j-transposed anti-symmetric term for T. Thus, we have a basis for the j-transposed anti-symmetric terms associated with this set. The size of this basis is one less than the number of nodes in the transposition-graph corresponding to this particular set, or alternatively one less than the number of monomials in this particular set. We will refer to a basis for $\text{Mon}^n(...)$ as $\text{Basis}^n(...)$, and the size of the basis $|\text{Basis}^n(...)|$ is one less than $|\text{Mon}^n(...)|$.

For all equations of the form (17), we then have a basis which is of size at most the total number of monomials minus the total number of disjoint sets of monomials. Thus, the number of linearly-independent equations obtained at grade n is less than the total number of monomials of degree n by at least the total number of disjoint sets at grade n. The total number of disjoint sets above is

$$\binom{t^3+n-1}{n}\binom{t+n-1}{n}, \tag{18}$$

as the unordered multi-set $(\langle i_1, i'_1, j'_1\rangle, ..., \langle i_n, i'_n, j'_n\rangle)$ corresponds to a degree n monomial in t^3 variables, and the unordered multi-set $(j_1, ..., j_n)$ corresponds to a degree n monomial in t variables.

It is not surprising that for every grade n, the number of linearly-independent equations is much less than the monomials, since there are multiple solutions to H (or G) given the original equations (15). In

particular, any G which is same as F^{-1} modulo tensor-products is a solution for G.

6.2 Additional Equations from the Neighbouring Steps

While one particular step of the obfuscated program yields information about F via the equations considered above, the neighbouring steps yield additional information about F. In particular, the Adversary not only gets samples of the form $X_i = F \cdot (a_i \otimes b_i)$, but also $Y_i = \hat{F} \cdot (b_i \otimes c_i)$, where $\hat{F}$ is another M_{t^2} matrix, and c_i is an arbitrary t-vector. Since by Corollary 3, $(b_i \otimes c_i) = \Omega(c_i \otimes b_i)$, without loss of generality we will assume that $Y_i = \hat{F} \cdot (c_i \otimes b_i)$. Tensoring X_i with Y_i then yields $X_i \otimes Y_i = (F \otimes \hat{F}) \cdot (a_i \otimes b_i \otimes c_i \otimes b_i)$. We can then consider the space $\mathsf{SYM}^{2,1,1}$ generated by arbitrary $a \otimes b \otimes c \otimes b$, as well as its cokernel $\mathsf{coSYM}^{2,1,1}$, and get new equations

$$(\mathsf{coSYM}^{2,1,1^{\mathsf{T}}} \otimes (\mathsf{SYM}^{2,1,1})^{\mathsf{T}}) \cdot \mathsf{vec}(\mathsf{G}F \otimes \hat{\mathsf{G}}\hat{F}) \ = 0, \tag{19}$$

where $\hat{\mathsf{G}}$ are new variables for $(\hat{F})^{-1}$. Again, we will let $\hat{\mathsf{H}}$ stand for $\hat{\mathsf{G}}\hat{F}$.

This time, the coefficients in the original equations (cf. (15)) can be written as

$$\sum_{\sigma} \mathsf{sgn}\,\sigma \cdot \epsilon(i_1, i_2, \sigma\langle j_1, j_2\rangle)^{\mathsf{T}} \cdot \sum_{\sigma'} \epsilon(i'_1, i'_2, \sigma'\langle j'_1, j'_2\rangle)^{\mathsf{T}} \tag{20}$$

Note $\mathsf{H} \otimes \hat{\mathsf{H}}$ has no kernel, and hence the above equation has no outer permutation. If we multiply the above equation by a (degree one) monomial $\mathsf{H}_{\langle i_3, j_3\rangle, \langle i'_3, j'_3\rangle}$, we get coefficients

$$\begin{aligned}\sum_{\sigma} &[\epsilon(\sigma\langle i_1, i_3\rangle, i_2, \sigma\langle j_1, j_3\rangle, j_2)^{\mathsf{T}} - \epsilon(\sigma\langle i_1, i_3\rangle, i_2, \sigma\langle j_2, j_3\rangle, j_1)^{\mathsf{T}}] \cdot \\ &[\epsilon(\sigma\langle i'_1, i'_3\rangle, i'_2, \sigma\langle j'_1, j'_3\rangle, j'_2)^{\mathsf{T}} + \epsilon(\sigma\langle i'_1, i'_3\rangle, i'_3, \sigma\langle j'_2, j'_3\rangle, j'_1)^{\mathsf{T}}]\end{aligned} \tag{21}$$

and if instead we multiply by $\hat{\mathsf{H}}_{\langle i_3, j_3\rangle, \langle i'_3, j'_3\rangle}$, we get

$$\begin{aligned}\sum_{\sigma} &[\epsilon(i_1, \sigma\langle i_3, i_2\rangle, j_1, \sigma\langle j_3, j_2\rangle) - (\epsilon(i_1, \sigma\langle i_3, i_2\rangle, j_2, \sigma\langle j_3, j_1\rangle)] \cdot \\ &[\epsilon(i'_1, \sigma\langle i'_3, i'_2\rangle, j'_1, \sigma\langle j'_3, j'_2\rangle) + (\epsilon(i'_1, \sigma\langle i'_3, i'_2\rangle, j'_2, \sigma\langle j'_3, j'_1\rangle)]\end{aligned} \tag{22}$$

Similarly, we can consider multiplying by higher degree monomials of degree $m-2$, with possibly both H and $\hat{\mathsf{H}}$ variables. All such equations can again be seen as linear combinations of *cross-transposed anti-symmetric terms* which are of the form:

$$\begin{aligned}&\mathsf{H}_{\langle i_1, j_1\rangle, \langle i'_1, j'_1\rangle} \cdot \ldots \cdot \mathsf{H}_{\langle i_n, j_n\rangle, \langle i'_n, j'_n\rangle} \cdot \hat{\mathsf{H}}_{\langle \hat{i}_1, \hat{j}_1\rangle, \langle \hat{i}'_1, \hat{j}'_1\rangle} \cdot \ldots \cdot \hat{\mathsf{H}}_{\langle \hat{i}_{\hat{n}}, \hat{j}_{\hat{n}}\rangle, \langle \hat{i}'_{\hat{n}}, \hat{j}'_{\hat{n}}\rangle} \\ -\ &\mathsf{H}_{\langle i_1, \hat{j}_1\rangle, \langle i'_1, j'_1\rangle} \cdot \ldots \cdot \mathsf{H}_{\langle i_n, j_n\rangle, \langle i'_n, j'_n\rangle} \cdot \hat{\mathsf{H}}_{\langle \hat{i}_1, j_1\rangle, \langle \hat{i}'_1, \hat{j}'_1\rangle} \cdot \ldots \cdot \hat{\mathsf{H}}_{\langle \hat{i}_{\hat{n}}, \hat{j}_{\hat{n}}\rangle, \langle \hat{i}'_{\hat{n}}, \hat{j}'_{\hat{n}}\rangle}\end{aligned}$$

where $n \geq 1, \hat{n} \geq 1$ and $n + \hat{n} = m$.

So, these cross-transposed anti-symmetric terms are essentially j-transposed anti-symmetric terms considered in the previous sub-section, except that the transposition goes across between H and $\hat{\mathsf{H}}$

variables. We must also consider original equations (15) multiplied by mixed-variable monomials, and these would yield j-transposed anti-symmetric terms, except the transpositions are restricted to be between H variables.

As in the previous section, we now split the set of monomials of degree $(n, \hat{n})$, $n + \hat{n} = m$, into a union of disjoint sets: each set is characterized by an unordered multi-set $(\langle i_1, i'_1, j'_1\rangle, ..., \langle i_n, i'_n, j'_n\rangle)$, an unordered multi-set $(\langle \hat{i}_1, \hat{i}'_1, \hat{j}'_1\rangle, ..., \langle \hat{i}_{\hat{n}}, \hat{i}'_{\hat{n}}, \hat{j}'_{\hat{n}}\rangle)$ and unordered multi-set $(j_1, ..., j_m)$, and consists of all (equivalence classes of) monomials

$$\mathsf{H}_{\langle i_1, j^*_1\rangle, \langle i'_1, j'_1\rangle} \cdot ... \cdot \mathsf{H}_{\langle i_n, j^*_n\rangle, \langle i'_n, j'_n\rangle} \cdot \hat{\mathsf{H}}_{\langle \hat{i}_1, j^*_{n+1}\rangle, \langle \hat{i}'_1, \hat{j}'_1\rangle} \cdot ... \cdot \hat{\mathsf{H}}_{\langle \hat{i}_{\hat{n}}, j^*_m\rangle, \langle \hat{i}'_{\hat{n}}, \hat{j}'_{\hat{n}}\rangle},$$

where $j^*_1, ..., j^*_m$ is a permutation of $j_1, ..., j_m$.

Each monomial belongs to at least one of these sets, and by construction the sets are pair-wise disjoint. Then, reasoning as before, the deficit of the number of linearly-independent equations from the total number of monomials of degree m is same as the number of disjoint sets considered above. The number of disjoint sets is at least

$$\sum_{n:m>n\geq 2} \binom{t^3+n-1}{n} \binom{t^3+(m-n)-1}{m-n} \binom{t+m-1}{m}.$$

6.3 Additional Equations from Gadgets

In this section we show that if there are additional $\mathrm{O}(t^2)$ linear or affine equations in G available, or additional $\mathrm{O}(t^4)$ quadratic equations in G available, one would need $n = \Omega(t)$ so as to make the number of equations exceed the number of monomials.

So, first consider the case of $\mathrm{O}(t^2)$ affine equations in G that one can get by just fixing about $2 \cdot t^2$ G-variables to arbitrary values. Since the equations (10) and (19) only determine G modulo tensor-product subgroup, one can indeed set about $2 \cdot t^2$ variables of G to arbitrary constants (and no more) so as to fix a representative of G modulo the tensor-product subgroup (see Section 7 for more details).

Now, if we multiply these equations E by monomials of degree $(n-1)$, we will get equations with monomials of degree $(n-1)$ and n. However, given that the equations (15) could be generated[9] as linear combinations of j-transposed anti-symmetric terms (dropping one for each set of disjoint monomials $\mathrm{Mon}^n(...)$), a careful analysis shows that the total number of new independent equations obtained by multiplying the new equations E by monomials is at most one per equation from E and a disjoint set $\mathrm{Mon}^{n-1}(...)$. To be more precise, consider an equation $E_{i,j} = \mathsf{G}_{i,j} - c_{i,j}$ in E. Now, consider two monomials m_1 and m_2 of degree $(n-1)$ such that they are in some same disjoint monomials set $\mathrm{Mon}^{n-1}(...)$. Then we claim that $m_1 E_{i,j}$ and $m_2 E_{i,j}$ are dependent given the basis for j-transposed anti-symmetric terms of degree n. First, check that for any x, y, $m_1\mathsf{H}_{x,y}$ and $m_2\mathsf{H}_{x,y}$ are in some same disjoint set $\mathrm{Mon}^n(...)$. Thus, $(m_1 - m_2)\mathsf{H}_{x,y}$ is a j-transpose anti-symmetric term of degree n. Since $\mathsf{G}_{i,j}$ is same as $\mathsf{H}F^{-1}$, the claim follows. Hence for each equation $E_{i,j}$ in E we only obtain (at most) as many new independent equations as there are disjoint Monomial sets of degree $n-1$.

[9]For simplicity we will ignore equations coming from the neighbouring step, although one can also analyze these equations together without getting a different result.

Hence, the total number of new independent equations we obtain is

$$O(t^2)\binom{t^3+n-2}{n-1}\binom{t+n-2}{n-1}.$$

Since, the deficit from the total number of degree n monomials was as given in (18), for the total number of independent equations to equal (or approach) the number of monomials we must have

$$O(t^2)\binom{t^3+n-2}{n-1}\binom{t+n-2}{n-1} = \binom{t^3+n-1}{n}\binom{t+n-1}{n}.$$

This implies $n = \Omega(t)$.

A similar analysis shows that if we get $O(t^4)$ additional quadratic equations in G, then we must have

$$O(t^4)\binom{t^3+n-3}{n-2}\binom{t+n-3}{n-2} = \binom{t^3+n-1}{n}\binom{t+n-1}{n}.$$

This implies that n continues to be $\Omega(t)$.

Note that the number of equations we get from each commutator gadget is equal to the number of monomials of degree two in both x and y, where x and y are vectors of length t^2 each. Thus, in variables corresponding to F^{-1}_{left} and F^{-1}_{right}, the number of degree four (two in each variable) equations[10] coming from a commutator gadget is of the order of $O(t^8)$. Again, by analysis similar to above we have $n = \Omega(t)$.

7 Further Cryptanalysis

In this section, we investigate the possibility of obtaining small hamming-weight "codewords" using Grobner basis computation, especially for the case where some components of G are fixed to random values (as in Section 6.3). Recall F is a random invertible matrix in M_{t^2}.

Since the equations (10) in G (or even the additional equations (19)) only determine F^{-1} modulo (left) tensor-product subgroup, an Adversary may fix some of the components of G to particular or random values, so as to obtain a representative of F^{-1} (modulo the (left) tensor-product of invertible matrices group). So, we first show that only $O(t^2)$ components of G can be fixed arbitrarily.

Since, G must still be in the same left-coset as F^{-1}, we must have $\mathsf{G}F = \mathsf{U} \otimes \mathsf{V}$. Recall, we let $\mathsf{H} = \mathsf{G}F$, So, fixing components of G is same as fixing components of $\mathsf{H}F^{-1}$, or $(\mathsf{U} \otimes \mathsf{V}) \cdot F^{-1}$. We will let T stand for F^{-1}. Now, assume that the full left column of G, i.e. G^0 is fixed to some values R^0. Hence, $(\mathsf{U} \otimes \mathsf{V})T^0 = R^0$. Or, $\mathsf{U}\,\mathsf{mat}(T^0)\mathsf{V}^{\mathsf{T}} = \mathsf{mat}(R^0)$. We will write $\mathsf{mat}(T^0)$ as T^0, and $\mathsf{mat}(R^0)$ as R^0, and hence $\mathsf{U}\mathsf{T}^0\mathsf{V}^{\mathsf{T}} = \mathsf{R}^0$. So, this then just determines U in terms of $\mathsf{V}^{-\mathsf{T}}$ or vice versa. We can try to fix another full column of G, say $\mathsf{G}^1 = R^1$, for some R^1. This would then

[10] Technically, there are t^2 output components, and hence $t^2 * O(t^8)$ equations. But, the degree of the equations goes up by one because of the additional variables for F_{node}. Similarly, for the root commutator-gadget we have an additional degree corresponding to the variables for the center-piece.

also imply $\mathsf{U}\mathsf{T}^1\mathsf{V}^\mathsf{T} = \mathsf{R}^1$. But, since T is random, and if we pick R^0 and R^1 at random, then this would lead to a contradiction with high probability (by just taking determinants of all quantities). Hence, we cannot set another full column of G to random values. We can, try to set all but one (last) component of G^1, and by letting z be a free variable for G_{1,t^2-1}, we get $\mathsf{G}^1 = R^1||z$, where now R^1 is a random (t^2-1)-column, and the operator "||" stands for column-wise concatenation. Now, we get $\mathsf{U}\mathsf{T}^1(\mathsf{T}^0)^{-1}\mathsf{U}^{-1}\mathsf{R}^0 = \mathsf{mat}(R^1||z)$, or $\mathsf{U} = \mathsf{mat}(R^1||z)(\mathsf{R}^0)^{-1}\mathsf{U}\mathsf{T}^0(\mathsf{T}^1)^{-1}$. For U to be not all zero, we must have that $\det((\mathsf{mat}(R^1||z)(\mathsf{R}^0)^{-1}) \otimes (\mathsf{T}^0(\mathsf{T}^1)^{-1})^\mathsf{T} - I)$ is zero. However, this with high probability also leads to $\det(\mathsf{U})$ being zero, so this is also not a viable way to set some components of G.

It turns out that one can set all but t components of G^1, and $(t-1)$ components of G^2, and then the t-th component of G^2 (say, z) has t solutions (in particular, we get a degree t equation in z, as can be seen from computing the determinant of the resulting linear system in $\mathsf{vec}(\mathsf{U})$). So, we have

$$\begin{aligned}
&\mathsf{U}\mathsf{T}^0\mathsf{V}^\mathsf{T} = \mathsf{R}^0\\
&\mathsf{U}_{0..t-2}\mathsf{T}^1\mathsf{V}^\mathsf{T} = \mathsf{R}^1\\
&\mathsf{U}_0\mathsf{T}^2\mathsf{V}^\mathsf{T} = \mathsf{R}^2,
\end{aligned} \tag{23}$$

where R^0 is a random M_t matrix, R^1 is a random $(t-1)\times t$ matrix and R^2 is a random row vector of length $t-1$ followed by a singleton free variable z. The determinant that determines the degree t equation in z is of the following matrix

$$(\mathsf{R}^1 \otimes (\mathsf{T}^1)^{-\mathsf{T}} \;||\; \mathsf{R}^2 \otimes (\mathsf{T}^2)^{-\mathsf{T}}) \cdot ((\mathsf{R}^0)^{-1} \otimes (\mathsf{T}^0)^\mathsf{T}) - I^{t^2}.$$

So, now consider the problem of determining this degree t polynomials in z from the equations (10) in G (or even (19)) and fixing some components of G as above. We are interested in finding if there is an *efficient* Grobner-basis computation of $p(z)$, and in fact $p(z)$ must lie in the ideal of these system of equations, and a Grobner-basis computation that eliminates all variables except z will eventually output $p(z)$. However, we want to determine if there is an efficient computation of $p(z)$ or even any polynomial with $p(z)$ as a factor using Grobner-basis computation. To this end, we first define what we mean by Grobner-basis computation and secondly, when do we consider such a method to be efficient.

Note that the system of equations is in free variables G and z defined over a field $\mathbb{F}$. We call a computation of $p(z)$ to be $\mathbb{F}[\mathsf{G}, z]$-**linear** if $p(z) = 0$ is in the $\mathbb{F}[\mathsf{G}, z]$-linear span of the equations. In other words, $p(z) = 0$ is obtained by multiplying each of the original equations by some polynomial in G and z and then adding them all up. We will refer to these multiplying polynomials as Macaulay factors. The computation is considered **efficient** if each of these Macaulay factors has only polynomial (in t) many monomials – each Macaulay factor is to be considered as a sum of monomials in G and z, and not as an arithmetic circuit or arithmetic formula. The reason for this last requirement is that this is the only known way that general Grobner basis computations work, as Grobner basis by definition deals with monomial ideals.

To reiterate, we have the following equations in G and z

$$\begin{aligned}
&(\mathsf{coSYM}^{2,2^{\mathsf{T}}} \otimes \mathsf{SYM}^{2,2^{\mathsf{T}}})\mathsf{vec}((\mathsf{G}F) \otimes (\mathsf{G}F)) \ = 0\\
&\mathsf{G}^0_{0..t^2-1} = R^0\\
&\mathsf{G}^1_{0..t^2-t-1} = R^1\\
&\mathsf{G}^2_{0..t-1} = R^2,
\end{aligned} \tag{24}$$

where R^0 is a random t^2-column, R^1 is a random $t(t-1)$- column and R^2 is a random $(t-1)$-column followed by free variable z. We now *conjecture*[11] that there is no efficient $\mathbb{F}[\mathsf{G}, z]$-linear computation of $p(z)$. To provide support for this conjecture, we first strengthen the above system and hence give more power to the Adversary, and replace the first set of equations by just $(\mathsf{G}F) = \mathsf{U} \otimes \mathsf{V}$ – note that this definitely is a solution of the first set of equations but is not necessarily the only solution (despite the Marcus-Moyls Theorem and its extension discussed at the beginning of Section 6). So, then we have

$$\begin{aligned}
&\mathsf{G}F = \mathsf{U} \otimes \mathsf{V}\\
&\mathsf{G}^0_{0..t^2-1} = R^0\\
&\mathsf{G}^1_{0..t^2-t-1} = R^1\\
&\mathsf{G}^2_{0..t-1} = R^2,
\end{aligned}$$

or more simply (recalling, $T = F^{-1}$)

$$\begin{aligned}
&(\mathsf{U} \otimes \mathsf{V})T^0_{0..t^2-1} = R^0\\
&(\mathsf{U} \otimes \mathsf{V})T^1_{0..t^2-t-1} = R^1\\
&(\mathsf{U} \otimes \mathsf{V})T^2_{0..t-1} = R^2.
\end{aligned}$$

and thus we seek an efficient $\mathbb{F}[(\mathsf{U} \otimes \mathsf{V})T, z)]$ computation of $p(z)$ from these equations. A computation of $p(z)$ linear in $(\mathsf{U} \otimes \mathsf{V})T$ is impossible if we treat each component of $(\mathsf{U} \otimes \mathsf{V})T$ as an independent term. Thus, we next check if an efficient $\mathbb{F}[\mathsf{U} \otimes \mathsf{V}, z]$-linear or more liberal $\mathbb{F}[\mathsf{U}, \mathsf{V}, z]$-linear computation of $p(z)$ is possible from the above equations. We remind the reader that the distinction in these alternate computations comes from the fact that the efficiency definition counts the number of monomials in the Macaulay factors and not the size of the arithmetic circuit or arithmetic formula.

Writing the above equations as (23), we note that multiplying the first of these equations, i.e. $(\mathsf{R}^0)^{-1}\mathsf{U}\mathsf{T}^0\mathsf{V}^{\mathsf{T}} = I$, into the second and third we get in an $\mathbb{F}[\mathsf{U}, \mathsf{V}, z]$-linear fashion the equations

$$\begin{aligned}
&\mathsf{U}_{0..t-2}\mathsf{T}^1\mathsf{V}^{\mathsf{T}} = \mathsf{R}^1(\mathsf{R}^0)^{-1}\mathsf{U}\mathsf{T}^0\mathsf{V}^{\mathsf{T}}\\
&\mathsf{U}_0\mathsf{T}^2\mathsf{V}^{\mathsf{T}} = \mathsf{R}^2(\mathsf{R}^0)^{-1}\mathsf{U}\mathsf{T}^0\mathsf{V}^{\mathsf{T}}.
\end{aligned}$$

However, removing V^{T} from these equations in an $\mathbb{F}[\mathsf{U}, \mathsf{V}, z]$-linear computation, so as to get the determinant of the resulting linear-system in $\mathsf{vec}(\mathsf{U})$, seems to require multiplication by inverse of V (or adjugate of V), which has exponential in t many monomials.

[11] The conjecture should also hold for additional equations (19) added to the above system of equations.

7.1 Computing Mixed Gadgets

If one does indeed manage to compute $p(z)$ *efficiently* from equations (24) (and additional equations (19)), then we show that, under reasonable assumptions, computing mixed gadgets is efficient as well. In other words, given $X_1 = F \cdot (a_1 \otimes b_1)$ and $X_2 = F \cdot (a_2 \otimes b_2)$, one can compute $F \cdot (a_1 \otimes b_2)$ upto scalar factors.

We will need the following lemma. Let $\mathbb{K}$ be the algebraic closure of $\mathbb{F}$.

Lemma 10. *For any ideal I of $\mathbb{F}[z, \mathbf{x}]$, where z is a single variable and $\mathbf{x}$ is a vector of independent variables, if a polynomial $p(z) \in \mathbb{F}[z]$ is the generator of $I \cap \mathbb{F}[z]$, an $\mathbb{F}[z]$ ideal, then for every root ζ of $p(z)$ (possibly in extension ring $\mathbb{E} = \mathbb{F}[z]/(p(z)) \subseteq \mathbb{K}$), there is a vector of values in $\mathbb{K}$, say $\mathbf{x}_\zeta$, such that $(\zeta, \mathbf{x}_\zeta)$ is in the zero-set of I.*

Proof. Since $\mathbb{F}[z, \mathbf{x}]$ is Noetherian, I is finitely generated. Let $I = (p(z)) + J$, where $J \cap \mathbb{F}[z] \subseteq (p(z))$, and J is finitely generated. For any root ζ of $p(z)$, consider the ideal of $\mathbb{K}[z, \mathbf{x}]$ given by $(z - \zeta) + J$. Now, either $(z - \zeta)$ and J are coprime or $(z - \zeta) + J$ is a non-trivial ideal of $\mathbb{K}[z, \mathbf{x}]$. In the latter case, by Hilbert's nullstellensatz (see e.g. Proposition 1.2 [Har77]), the zero-set of $(z - \zeta) + J$ is non-empty. But, if $(\zeta', \mathbf{x}')$ is such a member of the zero-set, ζ' must be same as ζ. And the claim of the theorem holds.

On the other hand, if $(z-\zeta)$ and J are coprime, i.e. $(z-\zeta)+J = (1)$, then multiplying by $p(z)/(z-\zeta)$, a polynomial in $\mathbb{K}[z]$, we get that $((p(z)) + J)\mathbb{K}[z, \mathbf{x}]$, i.e. $I\mathbb{K}[z, \mathbf{x}]$, contains $p(z)/(z - \zeta)$. We next claim that if $p(z)$ is the generator of $I \cap \mathbb{F}[z]$, then it is also the generator of $\mathbb{K}[z]$-ideal $I\mathbb{K}[z, \mathbf{x}] \cap \mathbb{K}[z]$, which would then contradict that I contains $p(z)/(z - \zeta)$. The claim is proved using Grobner basis theory as follows: let σ be a monomial order of $\mathbb{F}[z, \mathbf{x}]$ such that monomials in z are lower ranked than all other monomials except 1. Then a reduced Grobner basis G_σ of I will contain $p(z)$ as a basis element, since initial terms of G_σ must generate initial terms of I, which includes $p(z)$. It is well known that G_σ is also the Grobner basis of $I\mathbb{K}[z, \mathbf{x}]$ (see e.g. [Eis95]), and hence $p(z)$ also generates $I\mathbb{K}[z, \mathbf{x}] \cap \mathbb{K}[z]$. □

So, suppose a degree t polynomial $p'(z)$ is the generator of ideal $I \cap \mathbb{F}[z]$, where I is the ideal generated by polynomials in equations 24 (as well as additional equations (19)). Since, $p'(z)$ is zero at the zero set of I, and roots of $p(z)$ are in the zero-set of I, we must have that $p'(z)$ is a multiple of $p(z)$, and since they have the same degree, they must be same (up to scalars). Instead of $\mathsf{G}^2_{\langle 0, t-1 \rangle}$ being set to z, one can also consider any other G^i_j being set to z, and assume that a polynomial $p_{i,j}(z)$ can be computed efficiently as well. The lemma above relates $p(z)$ (i.e. $p_{2,\langle 0,t-1\rangle}(z)$) to $p_{i,j}(z)$ by roots. If $p_{i,j}(\zeta')$ has t solutions, one for each solution ζ of $p(z)$, then it can be shown that $\zeta' + f(\zeta) = 0$ where f is a degree $t - 1$ polynomial. Essentially, the two polynomials $p(z) = 0$ and $z' + f(z) = 0$ are zero at the same points as $p(z)$ and $p_{i,j}(z')$ (this can be seen by forming Vandermonde matrices). Thus, we get all of G in terms of a single parameter z (and modulo $p(z)$). Since the mixed gadget is uniquely determined up to scalar factors, this then implies that if one does arithmetic in the ring $\mathbb{F}[z]/(p(z))$, the ratio of components of the mixed gadget will be in the base field $\mathbb{F}$.

8 The One-More Masked Tensor Problem

In this section, we describe the so-called "One-More Masked Tensor Problem". The hardness of the One-More Masked Tensor Problem is neither necessary nor sufficient for the security of our scheme. However, the problem is easy to understand, and we believe that progress on it could shed light on the security of our scheme.

Let $m = t \times t$. Let $F \in M_m$ be a matrix over the field $\mathbb{F}$. Consider a distribution of samples $w^{(i)} \leftarrow F \cdot x^{(i)}$ where each $x^{(i)}$ is (the vectorization of) a random rank-1 matrix (a tensor) – i.e., $x^{(i)} = y^{(i)} \otimes z^{(i)}$ where $y^{(i)}, z^{(i)} \in \mathbb{F}^t$. With this setup, let us define the One-More Masked Tensor Problem.

Definition 2 (One-More Masked Tensor Problem)**.** Given as many samples $w^{(i)} \leftarrow F \cdot x^{(i)}$ as desired, output a "new" sample $w = F \cdot x$, where x is another tensor. "New" means that w is not simply a scalar multiple of one of the previous samples.

Observe that all valid samples satisfy certain nontrivial quadratic equations. Let $\bar{F}$ be the adjugate of F, such that $\bar{F} \cdot F = \det(F) \cdot I$. Then, for each $w^{(i)}$, we have $\bar{F} \cdot w^{(i)} = \det(F) \cdot x^{(i)}$, where $x^{(i)}$ is a tensor which satisfies certain quadratic equations. Specifically, let us index the $m = t \times t$ coefficients of $x^{(i)}$ by $(j,k) \in [t] \times [t]$ in the natural way, so that $x^{(i)}_{j,k} = y^{(i)}_j \cdot z^{(i)}_k$. Then, we have that $x^{(i)}_{(j_1,k_1)} \cdot x^{(i)}_{(j_2,k_2)} = x^{(i)}_{(j_1,k_2)} \cdot x^{(i)}_{(j_2,k_1)}$ for every "rectangle" (j_1, k_1, j_2, k_2). Letting $\bar{F}_{(j,k)}$ be the (j,k)-th row of $\bar{F}$, we have:

$$\langle \bar{F}_{(j_1,k_1)}, w^{(i)} \rangle \cdot \langle \bar{F}_{(j_2,k_2)}, w^{(i)} \rangle = \langle \bar{F}_{(j_1,k_2)}, w^{(i)} \rangle \cdot \langle \bar{F}_{(j_2,k_1)}, w^{(i)} \rangle \tag{25}$$

and thus:

$$\langle \bar{F}_{(j_1,k_1)} \otimes \bar{F}_{(j_2,k_2)} - \bar{F}_{(j_1,k_2)} \otimes \bar{F}_{(j_2,k_1)}, w^{(i)} \otimes w^{(i)} \rangle = 0 \tag{26}$$

So, every tensored sample $w^{(i)} \otimes w^{(i)}$ falls in a linear subspace defined by the vectors $\bar{F}_{(j_1,k_1)} \otimes \bar{F}_{(j_2,k_2)} - \bar{F}_{(j_1,k_2)} \otimes \bar{F}_{(j_2,k_1)}$, and further defined by fact that each $w^{(i)} \otimes w^{(i)}$ is (the vectorization of) a symmetric matrix. Alternatively, see Lemma 6 and the remark after the proof of the lemma. We can rephrase the One-More Masked Tensor Problem in terms of this subspace.

Definition 3 (One-More Masked Tensor Problem - Subspace Version)**.** Given as many samples $w^{(i)} \leftarrow F \cdot x^{(i)}$ as desired, output a "new" sample w such that $w \otimes w$ is in the subspace generated by the $w^{(i)} \otimes w^{(i)}$'s. "New" means that w is not simply a scalar multiple of one of the previous samples.

We can also consider a related problem that may be much harder.

Definition 4 (Masked Tensor Equations Problem)**.** Again, consider samples $w^{(i)} \leftarrow F \cdot x^{(i)}$. Given only a canonical representation of the quadratic equations satisfied by all of these samples, output nontrivial $\mathbf{w}$ that satisfies these quadratic equations. In other words, given $p(\mathbf{x}) = (\mathsf{coSYM}^{2,2}/\mathsf{coSYM}^2)^{\mathsf{T}} \cdot (F^{-1} \otimes F^{-1}) \cdot \mathsf{SYM}^2 \cdot \mathrm{Mon}^2(\mathbf{x})$, output non-trivial $\mathbf{w}$ such that $p(\mathbf{w}) = 0$.

We conjecture that these problems are hard for appropriate parameters. More concretely, we speculate that it may be possible to prove that these problems are impossible to solve using a generic algorithm. "Generic" means that only black-box operations $(+,-,\times,\div)$ are allowed, and also that the characteristic of the field is unknown. To put it another way, a generic solution must be a rational polynomial over the coefficients of the initial samples that is a *formal* solution.

Below, we describe some attacks on these problems. There is a generic attack for the case $t = 2$. (We explain why this attack does not extend to higher dimensions.) We also describe a few non-generic attacks, including an attack on the Masked Tensor Equations Problem when $t = 2$ and the "field" is $\mathbb{Z}_N$ for composite integer N, and some attacks on higher dimensions when the field has characteristic zero.

8.1 Attacks

8.1.1 Generic Solution to the One-More Masked Tensor Problem when $t = 2$

First, we explain the generic attack on the One-More Masked Tensor Problem when $t = 2$.

Consider the dimension of the subspace generated by the $w^{(i)} \otimes w^{(i)}$'s. We have $w^{(i)} \otimes w^{(i)} = (F \otimes F) \cdot (x^{(i)} \otimes x^{(i)})$ for invertible F, so the dimension is the same as the subspace generated by the $x^{(i)} \otimes x^{(i)}$'s. Writing $x^{(i)} \otimes x^{(i)}$ as $y^{(i)} \otimes z^{(i)} \otimes y^{(i)} \otimes z^{(i)}$, and observing that the rank of the $y^{(i)} \otimes y^{(i)}$'s is 3 due to symmetry, the rank of the $w^{(i)} \otimes w^{(i)}$'s is $3^2 = 9$. Since this is within a $4^2 = 16$ dimensional space, the kernel should have dimension $16 - 9 = 7$. Indeed, when $t = 2$, Equation 26 gives a single nontrivial vector orthogonal to the $w^{(i)} \otimes w^{(i)}$'s – namely:

$$\bar{F}_{(1,1)} \otimes \bar{F}_{(2,2)} - \bar{F}_{(1,2)} \otimes \bar{F}_{(2,1)},$$

and 6 more dimensions come from the subspace of vectorized anti-symmetric 4×4 matrices, which are orthogonal to $w^{(i)} \otimes w^{(i)}$ because it is symmetric. Given enough random samples, one can generate a basis for this kernel.

Pick any vector u in this kernel that is not in the subspace generated by the antisymmetric matrices. If you like, you can pick u to be a symmetric matrix: start with an initial choice of u, split it into symmetric and asymmetric parts ($u = u^+ + u^-$), set $u \leftarrow u^+$, and send u^- back into the subspace of antisymmetric matrices whence it came.[12] Now, view the equation $\langle u, w^{(i)} \otimes w^{(i)} \rangle = 0$ as saying the quadratic polynomial $u(w)$ has all valid samples as roots. If we find a new w such that $u(w) = 0$, then it will be a valid new sample, as $w \otimes w$ will be orthogonal to the subspace of antisymmetric matrices automatically, and will be orthogonal to $\bar{F}_{(1,1)} \otimes \bar{F}_{(2,2)} - \bar{F}_{(1,2)} \otimes \bar{F}_{(2,1)}$ since it is orthogonal to u.

All that remains is to explain how to generate a new solution to $u(w) = 0$ from old ones. For "most" quadratic equations over two or more variables, it is straightforward to generate a new solution from an old one. First, fix all but two variables to be equal to the old solution, and then diagonalize the equation over the remaining variables to obtain $ax^2 + by^2 = c$, an equation to which we already have

[12] Interestingly, this u^+, as the only symmetric component of the kernel subspace, must equal (up to scaling) the symmetrization of the nontrivial vector – namely, $\bar{F}_{(1,1)} \otimes \bar{F}_{(2,2)} + \bar{F}_{(2,2)} \otimes \bar{F}_{(1,1)} - \bar{F}_{(1,2)} \otimes \bar{F}_{(2,1)} - \bar{F}_{(2,1)} \otimes \bar{F}_{(1,2)}$. We can thus obtain this function of F from the samples when $t = 2$.

one solution (x_0, y_0) from the old sample. Suppose we are "lucky" and $a \neq 0 \neq c$. Then we can divide by a to get $x^2 - By^2 = C$ for $B = -b/a$ and nonzero $C = c/a$. View this equation as saying that the norm of $x_0 + y_0\sqrt{B}$ is C in the number field $\mathcal{V}(\sqrt{B})$. Then, the norm of $(x_0 + y_0\sqrt{B})^3/C$ must also be C in $\mathcal{V}(\sqrt{B})$, yielding the (likely different) solution $(x_1, y_1) = ((x_0^3 + 3Bx_0y_0^2)/C, (3x_0^2y_0 + By_0^3)/C)$ to $x^2 - By^2 = C$, which gives a new solution to $u(w) = 0$.

The main reason this attack works is that, when $t = 2$, a valid sample w only needs to be a root of a single quadratic polynomial, and it is easy to generate new rational solutions of a single multivariate quadratic polynomial given an initial solution. When $t > 2$, a sample must satisfy a larger system of quadratic equations.

8.1.2 Non-Generic Solution to Masked Tensor Equations Problem when $t = 2$

As we saw above, when $t = 2$, finding a valid sample is tantamount to solving a single multivariate quadratic equation. Somewhat surprisingly, the equation $ax^2 + by^2 = 1 \bmod N$ can be solved efficiently, even when N is composite, under the minimal assumption that $a - b \neq 0 \neq ab$.[13] The algorithm works by lifting to a problem $Ax^2 + By^2 - z^2 = 0$ over the integers, where $A = a \bmod N$, $B = b \bmod N$, A and B are distinct primes that are quadratic residues of each other, A is odd, and $B = 1 \bmod 4$. Suitable A and B can be found in probabilistic polynomial time. Once found, the equation $Ax^2 + By^2 - z^2 = 0$ is guaranteed to have a solution over the integers. A solution (x, y, z) to $Ax^2 + By^2 - z^2 = 0$ yields a solution $(x/z, y/z)$ to $ax^2 + by^2 = 1 \bmod N$.

For A and B satisfying the above conditions, Cremona and Rusin show how to reduce solving $Ax^2 + By^2 - z^2 = 0$ over the integers to finding the shortest vector in a 3-dimensional lattice. See [BGH07] for an exposition of this algorithm.[14] This algorithm is non-generic in a couple of ways. First, lifting the problem from one "field" to another (from $\mathbb{Z}_N$ to $\mathbb{Z}$ (or $\mathcal{V}$)) is non-generic. Second, lattice reduction crucially uses the *size* of elements to make progress, and *size* is a non-generic notion.[15]

One may point to this algorithm as a reason not to place any faith in generic proofs of impossibility, as it is "ruled out" by our proof that solving the Masked Tensor Equations Problem is generically impossible. Indeed, we cannot rule out non-generic algorithms for the Masked Tensor Equations Problem that work even when $t > 2$. We merely note that the algorithm for $t = 2$ appears quite specialized, and does not seem readily extensible to solving systems of quadratic equations (even when the system has a lot of structure, as in our case).

[13] Boneh, Gentry and Hamburg described an identity-based encryption scheme that actually uses this algorithm during the encryption process [BGH07].

[14] Just to give a taste of the algorithm, it is analogous to a lattice-based approach for expressing a prime $p \equiv 1 \bmod 4$ as the sum of two squares: Let r and s be such that $r^2 = 1 \bmod p$ and $s^2 = -1 \bmod p$ (these exist since $p \equiv 1 \bmod 4$). Consider the lattice generated by the rows $(1, s/r \bmod p)$ and $(0, p)$. Note that, for any vector $\vec{v} = (v_1, v_2)$ in this lattice, p divides its squared length $v_1^2 + v_2^2$. Suppose $\vec{v}$ is the shortest nonzero vector in the lattice. Since the lattice determinant is p, by Minkowski's theorem the squared length of $\vec{v}$ is less than $2p$, and therefore can only be p.

[15] Another example of a non-generic algorithm that uses size is the computation of a Jacobi symbol modulo N.

8.1.3 Attacks on the One-More Masked Tensor Problem in Characteristic Zero

Suppose we are given a basis of some subspace of matrices, together with the promise that there is basis of this subspace consisting entirely of rank-1 matrices; can we recover a rank-1 basis? If so, then we can solve the One-More Masked Tensor Problem: given a canonicalized basis generated by some $w^{(i)} \otimes w^{(i)}$'s, the algorithm will output a (likely different) basis of symmetric rank-1 matrices. Over the reals, it appears that this problem is solvable for large parameter sizes. (See [YSU17]).

8.2 Generic Solution to One-More Masked Tensor Problem Must Have High Degree

Let τ be minimal such that there is a formal arithmetic circuit that computes a new sample w from old samples $w^{(1)}, \ldots, w^{(\tau)}$. First, we show that $\tau \geq m$ (where m is the dimension of the w vectors). Next, we show that the total degree of the arithmetic circuit must be at least $\tau/2$.

This result certainly does not prove that computing a new sample is generically impossible, or even infeasible. But it does rule out naive application of certain low-degree attacks – for example, low-degree relinearization attacks a la Kipnis-Shamir and Courtois et al [KS99, CKPS00], and low-degree Grobner basis attacks. A successful attack will need to be more sophisticated.

To show that $\tau \geq m$, the basic idea is simple: given only $m-1$ samples, F is underdetermined, even if the values of $x^{(1)}, \ldots, x^{(m-1)}$ are also given. In particular, $F^{-1} \cdot w$ is a random and independent vector in the field if w is linearly independent of $w^{(1)}, \ldots, w^{(m-1)}$. Hence $F^{-1} \cdot w$ is likely not a tensor.

But suppose w is in the subspace of $w^{(1)}, \ldots, w^{(m-1)}$. Then we have $w = \sum_i f_i(\{w^{(j)}\}) \cdot w^{(i)}$ for some functions f_i. If $w = F \cdot x$, we have $x = \sum_i f_i(\{w^{(j)}\}) \cdot x^{(i)}$. But, viewing only $m-1$ samples $w^{(1)}, \ldots, w^{(m-1)}$ (and not knowing F *a priori*), the $w^{(j)}$'s appear independent of the $x^{(i)}$'s, and so we can replace $f_i(\{w^{(j)}\})$ with constant c_i, obtaining $x = \sum_{i \in [m-1]} c_i \cdot x^{(i)}$. But as the c_i's are independent of the tensors $\{x^{(i)}\}$, the formal rank of x is clearly equal to the number of nonzero c_i's. If only one c_i is nonzero, the new sample w is not new: it equals an old sample up to scaling.

Now, we establish that the degree must be at least $\tau/2$. Consider the "projected" arithmetic circuit that results when we set one of the old samples to 0. Clearly this projected arithmetic circuit outputs a sample that is *valid*, in the sense that it is F times a tensor. We consider two cases.

Case 1: There is some j such that zeroizing the j-th sample gives a projected formal polynomial is not a multiple of one of the original samples. Then, we can construct a new sample from $\tau - 1$ initial samples (contradiction).

Case 2: For all j, the projected formal polynomial is a (possibly zero) multiple of one of the old samples.

It remains to address Case 2. Suppose zeroizing the j-th sample leads to a projected polynomial that is a multiple of the i-th sample – that is,

$$C(w^{(1)}, \ldots, w^{(j-1)}, 0, w^{(j+1)}, \ldots, w^{(t)}) = f(\{w's\}) \cdot w^{(i)}.$$

Consider what happens to the output of the projected polynomial when we zeroize the j'-th sample for $j' \neq j$. It only changes the value of $f(\{w's\})$, and does not alter the fact that the output is some

multiple of $w^{(i)}$. Now, suppose we zeroize the j'-th sample and then j-th sample (in the opposite order of before). Zeroizing the j'-th sample gives a multiple of the i'-th sample for some i', and then zeroizing the j-th sample gives another multiple of the i'-th sample. If $i \neq i'$, since the i-th and i'-th samples are different, zeroizing the j-th and j'-th samples must set the entire formal polynomial to zero. Thus, every monomial of the formal polynomial must be divisible by either a coefficient of the j-th sample or of the j'-th sample for any (j, j') whose annihilation projects down to different (i, i'). (Note that if annihilating the j-th sample projects down to the zero polynomial, then every monomial must be divisible by some coefficient of the j-th polynomial.)

Consider a graph G, in which each vertex is associated to an index $j \in [\tau]$. Partition the vertices V into subsets $V_0, V_1, \ldots, V_t$, where V_0 consists of those indices j such that annihilating the j-th sample projects down to the zero polynomial, and otherwise V_i consists of j's that project down to a nonzero multiple of the i-th sample. Within V_0 draw an edge from each vertex to itself; otherwise, draw an edge between j and j' if they are in subsets V_i and $V_{i'}$ for $i \neq i'$. By the discussion above, the degree of the formal polynomial is lower-bounded by the size of the smallest vertex cover of this graph. Since the complement of a vertex cover is an independent set, the degree is lower-bounded by $\tau - |V_{i^*}|$, where V_{i^*} is the largest subset with $i^* \neq 0$.

Let S_{i^*} be the set of j's whose annihilation leads to a multiple of the i^*-th sample, where i^* is the index with the largest number of associated j's. (S_{i^*} corresponds to $V_0 \cup V_{i^*}$.) We lower-bound the degree by $|S_{i^*}|$. Together with the above result, this gives a lower-bound on the degree of $\tau/2$. To prove this result, we use inclusion-exclusion. For a subset $X \subseteq [\tau]$, let p_X be the portion of the formal polynomial containing monomials that are divisible by some coefficient of the i-th sample for *every* $i \in X$. Let q_X be the portion of the formal polynomial consisting of monomials that are divisible by a coefficient of the i-th sample for *some* $i \in X$. By inclusion-exclusion, for any set X, we have:

$$q_X = \sum_{j \in X} p_{\{j\}} - \sum_{(j_1,j_2) \subset X} p_{(j_1,j_2)} + \sum_{(j_1,j_2,j_3) \subset X} p_{(j_1,j_2,j_3)} - \cdots \pm p_X$$

Suppose X has cardinality k. Since $(1-1)^k \cdot C = 0$, we obtain:

$$(C - q_X) = \sum_{j \in X} (C - p_{\{j\}}) - \sum_{(j_1,j_2) \subset X} (C - p_{(j_1,j_2)}) + \cdots \pm (C - p_X) \tag{27}$$

Assume inductively that we have proven that $(C - p_X)$ is a multiple of the i^*-th sample for all $X \subseteq S_{i^*}$ with $|X| \leq k-1$; we prove it for k. From Equation 27, observe that our induction hypothesis implies that all of the addends are multiples of the i^*-th sample except possibly $C - q_X$ and $C - p_X$. But $C - q_X$ corresponds to the polynomial we obtain when we annihilate all of the samples associated to indices in X, and this result must be a multiple of the i^*-th sample. Therefore $C - p_X$ must be as well. Ultimately, we obtain that $C - p_{S_{i^*}}$ is a multiple of the i^*-th sample. Since C must output a *new* sample, $p_{S_{i^*}}$ must be nonzero. That is, there is a non-empty set of monomials in C that are divisible by coefficients from the j-th sample for every $j \in S_{i^*}$.

9 Security Parameters

We suggest the following parameters for target security level of 80 bits.

- Size of $\mathbb{F}$: The field size should be about 2^{80}, prime or otherwise. It is possible that a field size of about 2^{64} suffices, but this needs more analysis.
- Security parameter k: The parameter k, which is the size of the centerpiece matrices is recommended to be 3. This also means $t = k^2 = 9$ (see Section 6).
- Security parameter p: The repetition parameter p can be set to 10.
- The pre-procsseing parameter $p' = 1$ (see Section 3.2).

References

[BFS03] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. Complexity of gröbner basis computation for semi-regular overdetermined sequences over f2 with solutions in f2. *[Research Report] INRIA*, RR-5049, 2003. 1

[BFSS13] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Pierre-Jean Spaenlehauer. On the complexity of solving quadratic boolean systems. *J. Complexity*, 29(1):53–75, 2013. 1

[BG68] Richard L. Bishop and Samuel I. Goldberg. *Tensor Analysis on Manifolds*. Dover Publications, 1968. 4

[BGH07] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryption without pairings. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 647–657, 2007. 8.1.2, 13

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012. 1

[BP01] Paul Beame and Toniann Pitassi. Propositional proof complexity: Past, present, and future. In *Current Trends in Theoretical Computer Science*, pages 42–70. 2001. ECCC TR98-067. 1

[CEJvO02] Stanley Chow, Philip A. Eisen, Harold Johnson, and Paul C. van Oorschot. White-box cryptography and an AES implementation. In *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers*, pages 250–270, 2002. 1

[Cha73] P. J. Chase. Transposition graphs. *SIAM J. Comput.*, 2:128–133, 1973. 6.1

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, pages 392–407, 2000. 1, 8.2

[Eis95] David Eisenbud. *Commutative Algebra with a View toward Algebraic Geometry*. Springer-Verlag, 1995. 4.3, 6.1, 7.1

[Fei03] Wang Fei. Some problems on linear preservers. *Undergraduate Thesis, Dept. of Mathematics, National Univ. of Singapore*, 2003. 6

[GGH+16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM J. Comput.*, 45(3):882–929, 2016. 1, 1

[Gre78] Werner Greub. *Multilinear Algebra, 2nd Edition*. Springer-Verlag, 1978. 4, 4.1

[Har77] Robin Hartshorne. *Algebraic Geometry*. Springer-Verlag, 1977. 7.1

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, pages 19–30, 1999. 1, 8.2

[MM59] M. Marcus and B. N. Moyls. Transformations on tensor product spaces. *Pacific J. Math.*, 9:1215–1221, 1959. 6

[YL16] Dingfeng Ye and Peng Liu. Obfuscation without multilinear maps. *IACR Cryptology ePrint Archive*, 2016:95, 2016. 1

[YSU17] Nakatsukasa Yuji, Tasuku Soma, and André Uschmajew. Finding a low-rank basis in a matrix subspace. *Mathematical Programming*, 162(1-2):325–361, 2017. 8.1.3

A Tensor-Product Encoding Scheme II

We now describe an alternate, more efficient, tensor-product encoding scheme. In the full version of the paper, we will show that the security analysis (as in Section 6) of the Section 3.1 encoding scheme continues to hold for this scheme.

As before, we employ two security parameters k and p. Instead of a binary-tree fashioned encoding, this scheme will be flat above the root gadget.

The encoding scheme (with parameters $k, p, n, \texttt{lmax}$) takes the following as input:

1. an $\mathrm{SL}_k(\mathbb{F})$ matrix C (called *center-piece*),

2. for each $j \in [0..p*n-1]$, a length $\mathsf{len}(j)$ ($\leq \mathtt{lmax}$), and a $\mathsf{len}(j)$-length list of *pairs* of M_k matrices, say $(\langle G_{j,0}, H_{j,0}\rangle, ..., \langle G_{j,\mathsf{len}(j)-1}, H_{j,\mathsf{len}(j)-1}\rangle)$. The matrices G, H will be generically referred to as *small Kilian matrices*.

In the immediate application, $\mathtt{lmax}$ will be two. We will also assume that len is consistent, i.e. for all j, $\mathsf{len}(j) = \mathsf{len}(j+n)$.

Let $k' = k^2$.

The output of the encoding will include for each $j \in [0..p*n-1]$ a list of matrices $(P_{j,0}, ..., P_{j,\mathsf{len}(j\%n)-1})$ where each $P_{j,z}$ is a $M_{k'}$ matrix. The output will also include a *root gadget*.

We now describe the encoding procedure.

- Pick $p*n+1$ random and independent invertible $M_{k'}$ matrices $F_0, ..., F_{p*n}$.
- For each $j \in [0..p*n-1]$, for each $z \in [0..\mathsf{len}(j)-1]$, let

$$P_{j,z} = F_{j+1} \cdot (G_{j,z} \otimes H_{j,z}) \cdot F_j^{-1},$$

- **Root Gadget:** Define $\mathsf{R1}$ to be the affine algebraic set:

$$\mathsf{R1} \;=\; \{v \in \mathbb{F}^{k^4} \mid \exists A, B \in M_k : v = (F_{p*n} \otimes F_0^{-\mathsf{T}}) \cdot \mathsf{vec}(A \otimes B)\}$$

  The root gadget is defined to be the following:

$$\begin{aligned}&\mathrm{Gadget}_{\mathrm{root}} : \mathsf{R1} \to M_k \text{ given by}\\ &\mathrm{Gadget}_{\mathrm{root}}(\mathbf{x}) = \mathsf{mat}(\mathsf{mat}((F_{p*n}^{-1} \otimes F_0^{\mathsf{T}}) \cdot \mathbf{x}) \cdot \mathsf{mat}((F_{p*n}^{-1} \otimes F_0^{\mathsf{T}}) \cdot \mathbf{x})^{\mathsf{T}} \cdot \mathsf{vec}(C)))\end{aligned}$$

  As shown in Theorem 8 (b), the above gadget can be implemented efficiently.

A.1 Using the Tensor-Product Encoding II

Recall that the encoding scheme produces as output a collection of matrices $P_{j,z}$ and a *root gadget*.

A legitimate use of these gadgets is to pick a (not necessarily consistent) assignment $x[j]$ for each $j \in [0..p*n]$ ($x[j] \in [0..(\mathsf{len}(j)-1)]$), and then pick the corresponding masked tensor-product $P_{j,x[j]}$. Multiply them all in sequence to get a matrix whose vectorization is in $\mathsf{R1}$. Next, the root gadget is used to compute the (unmasked) quadratic function (in this case, MM^{T}) of the tensor-products applied to $\mathsf{vec}(c)$. This leads to a dynamic-fence applied to C, with a function of the G matrices (picked according to assignment x) multiplied on the left and a function of the matched H matrices applied on the right. We will refer to this value as **sandwiched-centerpiece**.

The full obfuscated program is as in Section 3.2, and the full evaluation of the obfuscated program is as in Section 3.3.

A.2 Security Parameters

We suggest the following parameters for target security level of 80 bits.

- Size of $\mathbb{F}$: The field size should be about 2^{80}, prime or otherwise. It is possible that a field size of about 2^{64} suffices, but this needs more analysis.
- Security parameter k: The parameter k, which is the size of the centerpiece matrices is recommended to be 3. This also means $t = k^2 = 9$ (see Section 6).
- Security parameter p: The repetition parameter[16] p can be set to 10.
- The pre-processing parameter $p' = 0$ (see Section 3.2).

A.3 Back of the envelope calculations

1. Branching Program Size $m = 10^6$.
2. Inputs $n = 100$.
3. Each step has $n * p = 1000$ leaves.
4. Obfuscated Program Size
   - Each leaf has 4 matrices, two in each track. Total matrices 4000.
   - One gadget in each track. Each gadget has size $t * \binom{t+1}{2}^2 \approx 9 * 45^2$. Total gadget size per step = 36000 field elements.
   - Total size per step = $81 * 4000 + 36000 = 360000$ field elements.
   - Total size of obfuscated program $36 * 10^9$ field elements or 288 GBytes (assuming field size of 8 bytes).
5. Time to compute using the Obfuscated Program
   - Computing using the Gadget: compute $\text{Mon}^2(x)_{\text{full}}$ requires $\binom{t+1}{2}^2$ multiplications, which is about 2000 multiplications. Next, a matrix of size 9×2000 is multiplied into this vector, which is about $9 * 2000$ multiplications. So, total 20,000 field multiplications.
   - Multiplying the leaf matrices. There are 1000 leaves, so that many matrix multiplications of size 9×9. Assuming $n^{2.5}$ time for multiplication, that means 100 multiplications per matrix multiplication. Thus, total multiplications is 10^5, which dominates the gadget computation time.
   - Thus, total multiplications for the whole program is of the order of 10^{11}.
   - Assuming a 2GHz modern Intel single core computer, and assuming 64-bit modular multiplication (in a near Mersenne-prime field) requires 10 cycles in a pipelined fashion, the total time is $10^{11} * 10/(2 * 10^9)$, which is 500 secs, or 10 minutes. The more aggresive setting of $p = 1$ would result in time of 1 minute.

[16] A more aggresive implementation can even try $p = 1$, $p' = 0$.

B Further Details of Proof of Theoerm 8

In the proof of theorem 8, it was claimed that $\mathbf{x} \in \mathsf{R1}(F)$ iff $\mathrm{Mon}^2(\mathbf{x}) = \text{reduced-}W \cdot \mathrm{Mon}^2(\mathbf{x})_{\text{full}}$.

For matrices X and Y with the same number of columns we will use $X \mid\mid Y$ to denote the matrix where Y is stacked below X. Similarly, if A and B have the same number of rows, then $A \mid B$ denotes matrix A appended with columns of B.

First note that if a full-ranked matrix $(A \mid B)$, such that B is a square matrix, has kernel $(X \mid\mid Y)$, where X is an invertible square matrix, then B is also invertible. Since, X is invertible, the kernel is also spanned by $(I \mid\mid YX^{-1})$. If B is not invertible then there is a non-zero vector u such that $Bu = 0$. then $(0 \mid\mid u)$ is in the kernel of $(A \mid B)$, but is not in the span of $(I \mid\mid YX^{-1})$. Contradiction.

Also, note that since $AX + BY = 0$, we have

$$B^{-1}A + YX^{-1} = 0. \tag{28}$$

Thus, $(A \mid B) \cdot \mathrm{Mon}^2(\mathbf{x}) = 0$ is equivalent to $(I \mid\mid -B^{-1}A) \cdot (\mathrm{Mon}^2(\mathbf{x}))_{\text{top}} = \mathrm{Mon}^2(\mathbf{x})$, which is equivalent to $(X \mid\mid Y) \cdot X^{-1} \cdot (\mathrm{Mon}^2(\mathbf{x}))_{\text{top}} = \mathrm{Mon}^2(\mathbf{x})$.

If X is not invertible, then since $(X \mid\mid Y)$ is the kernel of a matrix it must be full-ranked and hence there is some subet of indices of rows such that the resulting sub-matrix is invertible. Calling this subset of indices "full", the above proof extends to show that $(X \mid\mid Y) \cdot (X \mid\mid Y)^{-1}_{\text{full}} \cdot (\mathrm{Mon}^2(\mathbf{x}))_{\text{full}} = \mathrm{Mon}^2(\mathbf{x})$.

C Diagonalization of Direct Sum of Different Ranked Spaces

In this section we give a polynomial time algorithm to diagonalize Kilian matrices F, when given many samples X_i of the form $F \cdot B_i \cdot G$, where B_i are block diagonal matrices with two blocks such that the top block is a random symmetric matrix, and the bottom block is a general random matrix. This algorithm illustrates that if the XL-methodology gives a polynomial time algorithm then one can usually get an even faster polynomial time algorithm with direct rank analysis. We leave it as an exercise to the reader to figure out why this algorithm does not generalize in a straightforward manner to the masked tensor problems considered in the main sections.

The two diagonal blocks will be called BTL_i and BBR_i resp. (for top-left and bottom-right resp.). Let $D = F^{-1}$ and $E = G^{-1}$. Thus, $B_i = D \cdot X_i \cdot E$. Letting DT stand for the top half rows of D and RL stand for the left half columns of E, we have that $DT \cdot X_i \cdot EL$ is a symmetric matrix BTL_i. Moreover, $DT \cdot X_i \cdot ER = 0$, where ER is the right half columns of E. Further, let ELT and ELB denote the top and bottom rows of EL (and similarly for ERT, ERB). Let XL_i be the left half columns of X_i and XR_i be the right half columns of X_i. Thus,

$$DT \cdot \mathsf{XL}_i \cdot ELT \; + \; DT \cdot \mathsf{XR}_i \cdot ELB = \mathsf{BTL}_i.$$

Also,

$$DT \cdot \mathsf{XL}_i \cdot ERT = -DT \cdot \mathsf{XR}_i \cdot ERB.$$

Thus from these two we get,

$$DT \cdot \mathsf{XL}_i \cdot ELT \; - \; DT \cdot \mathsf{XL}_i \cdot ERT \cdot ERB^{-1} \cdot ELB = \mathsf{BTL}_i.$$

Or, letting H stand for $ELT - ERT \cdot ERB^{-1} \cdot ELB$, we get that

$$DT \cdot \mathsf{XL}_i \cdot H = \mathsf{BTL}_i \tag{29}$$

is a symmetric matrix.

To be able to diagonalize F, what we really want is $F_{10} * F_{00}^{-1}$, where F has blocks F_{00} etc. This is equivalent to getting $DTR * DTL^{-1}$ (again using the above naming convention), or getting linear relations between columns of DT. If BTL_i is an m by m matrix, then we seek m relations between columns of DT. Focusing on any two rows of DT, say j and k, we have (using the symmetric matrix property) that

$$DT^j \cdot \mathsf{XL}_i \cdot H_k = DT^k \cdot \mathsf{XL}_i \cdot H_j.$$

Here superscripts denote row indices, and subscripts denote column indices. Multiplying both sides by H_j^{T}, we get

$$DT^j \cdot \mathsf{XL}_i \cdot H_k H_j^{\mathsf{T}} = DT^k \cdot \mathsf{XL}_i \cdot H_j H_j^{\mathsf{T}}.$$

Now, $H_j H_j^{\mathsf{T}}$ is symmetric. For simplicity, we now focus on the case $m = 3$. The argument and algorithm generalizes to arbitrary m. We now seek scalars α_i, β_i such that in the equation (let S_i denote $\begin{bmatrix} 0 \\ \alpha_i \\ \beta_i \end{bmatrix}$),

$$\sum_i DT^j \cdot \mathsf{XL}_i \cdot H_k H_j^{\mathsf{T}} \cdot S_i = \sum_i DT^k \cdot \mathsf{XL}_i \cdot H_j H_j^{\mathsf{T}} \cdot S_i \tag{30}$$

the right hand side is identically zero, and the left hand side is not identically zero. In other words,

$$\sum_i \mathsf{XL}_i \cdot H_j H_j^{\mathsf{T}} \cdot S_i$$

is zero, but

$$\sum_i \mathsf{XL}_i \cdot H_k H_j^{\mathsf{T}} \cdot S_i$$

is not zero. This is possible because $H_j H_j^{\mathsf{T}}$ is a symmetric matrix and hence has a lower rank, as we elaborate below. Let $\mathsf{XL}_{t,i}$ denote the t-th column of XL_i. Then, for the above to hold we necessarily and sufficiently require that the following hold non-trivially.

$$\sum_i \alpha_i \mathsf{XL}_{0,i} = 0, \tag{31}$$

$$\sum_i \beta_i \mathsf{XL}_{0,i} = 0, \tag{32}$$

$$\sum_i \alpha_i \mathsf{XL}_{1,i} = 0, \tag{33}$$

$$\sum_i \beta_i \mathsf{XL}_{2,i} = 0, \tag{34}$$

$$\sum_i \alpha_i \mathsf{XL}_{2,i} + \beta_i \mathsf{XL}_{1,i} = 0 \tag{35}$$

In other words the following matrix needs to be non-full-ranked:

$$\chi = \begin{bmatrix} \mathsf{XL}_0^\mathsf{T} & 0 & \mathsf{XL}_1^\mathsf{T} & 0 & \mathsf{XL}_2^\mathsf{T} \\ 0 & \mathsf{XL}_0^\mathsf{T} & 0 & \mathsf{XL}_2^\mathsf{T} & \mathsf{XL}_1^\mathsf{T} \end{bmatrix},$$

where XL_t ($t \in [0..2]$) denotes the matrix composed of columns $\mathsf{XL}_{t,i}$. First note that the rank of the space of matrices of the form X_i above is $m * (m+1)/2 + m^2$ (the two terms coming from symmetric BTL and general BBR). Thus, for $m = 3$, the indices i range from [0..14]. We now show that the above 30×30 matrix has nullity (at least) three (or, m in general).

First, note that $\mathsf{XL}_{t,i} = F \cdot B_i \cdot G_t$. Thus, the rank of the above will remain same if we replace $\mathsf{XL}_{t,i}$ by $\mathsf{YL}_{t,i} = B_i \cdot G_t$ in the above matrix χ (call this new matrix Ψ). Since, BTL_i component of B_i are symmetric, consider the matrix

$$K = \begin{bmatrix} \mathsf{GT}_2^\mathsf{T} & 0 & \mathsf{GT}_1^\mathsf{T} & 0 & 0 & 0 & 0 & 0 & -\mathsf{GT}_0^\mathsf{T} & 0 \\ 0 & 0 & \mathsf{GT}_2^\mathsf{T} & 0 & 0 & 0 & -\mathsf{GT}_0^\mathsf{T} & 0 & 0 & 0 \\ \mathsf{GT}_1^\mathsf{T} & 0 & 0 & 0 & -\mathsf{GT}_0^\mathsf{T} & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

(Here GT is the top half rows of G.) Then, the above matrix Ψ (i.e. the one with YL) right multiplied by K^T is zero.

So, now let α_i, β_i be any sequence of scalars from the left-kernel of χ. Note, we have three linearly-independent choices for such scalars. Then, using equation (30), we have

$$\sum_i DT^j \cdot \mathsf{XL}_i \cdot H_k H_j^\mathsf{T} \cdot S_i = 0$$

Further, we can just take the anti-symmetric part of $H_k H_j^\mathsf{T}$ in this equation, as the symmetric part would also annihilate, and hence

$$\sum_i DT^j \cdot \mathsf{XL}_i \cdot (H_k H_j^\mathsf{T} - H_j H_k^\mathsf{T}) \cdot S_i = 0$$

By taking, $j = 1$ and $k = 2$ (recall, $t \in [0..2]$), we have

$$\sum_i DT^1 \cdot \mathsf{XL}_i \cdot (H_2 H_1^\mathsf{T} - H_1 H_2^\mathsf{T}) \cdot S_i = 0$$

Again, by properties (31) and (32) above, we get

$$\sum_i DT^1 \cdot (\mathsf{XL}_{1,i} \cdot \beta_i - \mathsf{XL}_{2,i} \cdot \alpha_i) \cdot (H_{2,2} H_{1,1} - H_{1,2} H_{2,1}) = 0$$

And hence,

$$\sum_i DT^1 \cdot (\mathsf{XL}_{1,i} \cdot \beta_i - \mathsf{XL}_{2,i} \cdot \alpha_i) = 0.$$

We get three different such equations. The same equations hold for DT^0 and DT^2 as well, and that diagonalizes D (with some more work).