Chinese Journal of Electronics
Vol.32, No.2, Mar. 2023

Code-Based Conjunction Obfuscation

ZHANG Zheng[1,2], ZHANG Zhuoran[1,2], and ZHANG Fangguo[1,2]

(1. *School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China*)

(2. *Guangdong Province Key Laboratory of Information Security Technology, Guangzhou 510006, China*)

Abstract — The pattern-matching problem with wildcards can be formulated as a conjunction where an accepting string is same as the pattern for all non-wildcards. A scheme of conjunction obfuscation is a algorithm that "encrypt" the pattern to prevent some adversary from forging any accepting string. Since 2013, there are abundant works about conjunction obfuscation which discussed with weak/strong functionality preservation and distributed black-box security. These works are based on generic group model, learning with error assumption, learning with noise assumption, etc. Our work proposes the first conjunction obfuscation with strong functionality preservation and distributed black-box security from a standard assumption. Our scheme with some parameter constraints can also resist some related attacks such as the information set decoding attack and the structured error arrack.

Key words — Obfuscation, Conjunction, Random linear code, General decoding problem, Virtual black-box security.

I. Introduction

Obfuscation has long been a question of great interest in a wide range of cryptography [1]–[3]. An obfuscator conceals the internal structure and privacy information of a program while holding its functionality. It is a powerful tool which is applied to functional encryption [4], multiparty key exchange [5], deniable encryption [6] and so on.

Nevertheless, most published studies of general indistinguishability obfuscation scheme are limited to multilinear maps [7]–[9], but all known multilinear maps are attacked [10]–[15]. Little knowing about multilinear maps leads to the difficulties in designing an indistinguishability obfuscator effectively and securely. Gratifyingly, the obfuscation of the specific function, especially evasive circuit family, has sprouted quantitative works in recent years.

An evasive circuit family consists of those functions which have sparse accepting inputs [16]. Point functions, hyperplanes and conjunctions are all evasive functions. The obfuscation for evasive functions will resist the adversary to guess the accepting inputs. In the early stage, much of the research has explored how to design an obfuscator of point functions with a one-bit point or a multi-bit point [17], [18], an auxiliary input or not [19], [20]. Until 2016, Bellare and Stepanovs [21] first gave a brief overview of point-function obfuscation with a formal framework and generic constructions. Besides the obfuscations for point functions, Canetti, Rothblum and Varia used the generic group model (GGM) to design an obfuscation of hyperplane membership [22]. In 2017, Goyal *et al.* proposed a lockable obfuscation under the learning with errors (LWE) assumption [23] while Wichs and Zirdelis obfuscated compute-and-compare programs under the same assumption [24].

We focus on a specific evasive function – conjunctions in this work [25]–[28]. A conjunction is denoted by a Boolean function $f(x_1,\dots,x_n)=\wedge_{i\in S}\ell_i$ where S is a subset of $[n]$ and ℓ_i is either x_i or $\neg x_i$. From another point of view, it is an instance of pattern-matching with wildcards. A pattern with wildcard is denoted by a string $\boldsymbol{pat}\in\{0,1,*\}^n$. The character $*$ is a wildcard which does not require matching. Let the set $\{i\mid i\in[n]$ and $i\notin S\}$ be the position indexes of wildcards. It implies that a pattern-matching with wildcards outputs 1 if and only if $\ell_i=x_i$ for every $i\in S$. For example, a Boolean function $f(x_1,\dots,x_5)=\ell_1\wedge\ell_2\wedge\ell_4=x_1\wedge\neg x_2\wedge x_4$ with $S=\{1,2,4\}$ is equal to $\boldsymbol{pat}=10*1*$ while the input $\boldsymbol{x}=10110$ matches it and $\boldsymbol{x}=00110$ does not. It is easy to know that a conjunction outputs 0 on most of inputs $\boldsymbol{x}\in\{0,1\}^n$ which is a evasive function.

Manuscript Received Nov. 15, 2020; Accepted July 18, 2022. This work was supported by Guangdong Major Project of Basic and Applied Basic Research (2019B030302008) and the National Natural Science Foundation of China (61972429).

DOI:10.23919/cje.2020.00.377

The obfuscation for pattern-matching with wildcards was beginning with Brakerski and Rothblum's work in CRYPTO 2013 [25]. However, this work does not separate oneself from the multilinear map. In a follow-up study, Brakerski *et al.* proposed a new construction under entropic ring LWE [26]. In 2013, Bishop *et al.* (denoted by BKM18) proposed one ingenious construction of conjunction obfuscation using GGM [27]. They encode the output of a degree $(n-1)$ polynomial or not depending on each pattern's character. When one input matches the pattern, it picks the corresponding codes and reconstructs the polynomial by Lagrange interpolation method. This work is ingenious and elegant. Following this, Bartusek *et al.* (denoted by BLM19) conducted an extensive study of this conjunction obfuscation's techniques [28]. Concretely, they considered how to embed the pattern in the error positions of a noisy Reed-Solomon codeword and proposed three schemes, one over exponential size alphabets (denoted by BLM19-1), one from the decisional exact learning parity with noise (DxLPN) problem (denoted by BLM19-2) and one satisfied information-theoretic security (denoted by BLM19-3). Recently, Steven and Lukas [29] (denoted by SL19) proposed a reduction between an obfuscator for fuzzy matching for Hamming distance and conjunction obfuscations, which is based on the distributional modular subset product problem (SPP). The concrete comparison of these schemes are shown in Table 1 where "FP" means functionality preservation.

Table 1. The complexity comparison between our constructions and related works

	Standard model	Assumption	Alphabet	FP	Distribution
BKM18	×	GGM	Binary	Strong	$m < 0.774n$
BLM19-1	×	GGM	Exponential	Strong	$m < n - \omega(\log n)$
BLM19-2	✓	DxLPN	Binary	Weak	$m = cn, c < 1$
BLM19-3	✓	–	Binary	Weak	$H_\infty(b \mid \boldsymbol{pat}^{-1}(*)) \geq n^{1-\gamma}$
SL19	✓	SPP	Binary	Strong	$m < n/2$
This work	✓	GD	Binary	Strong	$m = cn, c < 1$

Note: The definitions of notations about distribution are the same as that in [28].

In this work, we will solve the open problem in [28]: How to construct an obfuscator for conjunctions with strong functionality preservation under the standard assumption? The strong functionality preservation requires the obfuscated program is simultaneous correct with overwhelming probability on all inputs while the weak one only needs correctness for every input. It means that the weak functionality preservation still needs the negligible mistake for every input does not hold for a large fraction of inputs. When it comes to strong functionality preservation, it needs the following equation:

$$
\begin{aligned}
&Pr[Obf(1^n, C)(x) = C(x) \forall x \in \{0,1\}^n] \\
&= \sum_{x \in \{0,1\}^n} \mid Pr[Obf(1^n, C)(x) = C(x) \mid \\
&= 1 - \mathrm{negl}(n)
\end{aligned}
$$

Thus, the strong functionality preservation is simultaneous correct with overwhelming probability on all inputs.

The BKM18 scheme and BLM19-1 scheme can achieve strong correctness by using the group with sufficiently large size of 2^{2n}. When it comes to code-based construction, the BLM19-2 scheme only achieves weak functionality preservation. The reason is that the BLM19-2 scheme is based on binary code and depends on the LPN assumption, which results in the invalidness of expanding the size of the code domain for BLM19-2. Thus, we will try to construct a conjunction obfuscation based on the general decoding problem. When the domain $\mathbb{F}_q$ is expanded to $q > 2^{2n}$, the obfuscator can reach strong functionality preservation.

Our scheme begins with a $[2n, n-k]$ random linear code $\mathcal{C}$ over $\mathbb{F}_q$ with generator matrix $\boldsymbol{G} \leftarrow \mathbb{F}_q^{(n-1)\times 2n}$. The pattern $\boldsymbol{pat} \in \{0,1,*\}^n$ will be replaced with an error vector $\boldsymbol{e} \in \mathbb{F}_q^{2n}$ and each element $\boldsymbol{pat}_i$ corresponds to two elements e_{2i-1}, e_{2i}. For every wildcard element $\boldsymbol{pat}_i = *$, e_{2i-1}, e_{2i} are set as zero elements. Otherwise, we set zero element $e_{2i-b} = 0$ and random element $e_{2i-(1-b)} \leftarrow \mathbb{F}_q$ for $\boldsymbol{pat}_i = b$. Each matching input $\boldsymbol{x} \in \{0,1\}^n$ corresponds to an all-zero subvector $\boldsymbol{e} = (e_{2-x_1}, e_{4-x_2}, \dots, e_{2n-x_n})$. Our obfuscator invokes $\boldsymbol{c} = \boldsymbol{mG} + \boldsymbol{e}$ to hide the error vector where it also hides the pattern. Then the evaluation algorithm only needs to determine whether the subcode $\boldsymbol{c_x} = (c_{2-x_1}, c_{4-x_2}, \dots, c_{2n-x_n})$ is error-free.

The new evaluation algorithm guarantees the probability of the obfuscated program outputs 1 on an unmatching input is $1/q + \mathrm{negl}(n)$. Then we can select the parameter $q > 2^{2n}$ to achieve strong functionality preservation.

Our results This work intends to explore the construction of conjunction obfuscations with simultaneous correctness under standard assumption. We choose the general decoding problem as our security assumption. Our scheme is proven to satisfy strong functionality preservation and distributional virtual black-box security. Given some parameter constrains, the obfuscation can resist information set decoding (ISD) over $\mathbb{F}_q$ and the structured error attack. Beyond that, we also provide some optimized schemes to reduce the size of the obfuscated program, the complexity of the evaluation algorithm and expand the functionality which outputs multi-bit.

Organization This paper was organised in the following way. After introducing the background of this paper, the next section will show some notations and formal security definitions used for this work. The third section presents detailed construction and security analysis. It will then go on to the related attacks and parameter analysis. On Section IV, there are some optimized schemes and the conclusion is in the last part.

II. Preliminaries

The following is a brief description of the notations and security definitions used in this study, consisting of distributional virtual black-box obfuscation, linear codes and security assumption.

1. Notations

The set $\{1, 2, \ldots, n\}$ is denoted by $[n]$. The boldface character will be used in this study to refer to vector, such as $\boldsymbol{u}$, $\boldsymbol{v}$. The i-th element of one vector $\boldsymbol{u}$ is defined as u_i. That is to say, a vector is denoted by $\boldsymbol{u} = (u_1, \ldots, u_n)$. The capital letters are used to denote the matrix, such as $\boldsymbol{G}$. And $(\boldsymbol{G})_i$ refers to the i-th column of the matrix $\boldsymbol{G}$. Let $\mathbb{F}_q$ denote a finite field of q elements. In this essay, the negligible function $\text{negl}(\cdot)$ is (asymptotically) smaller than any inverse polynomial.

Throughout this paper, a conjunction is a Boolean function $f(x_1, \ldots, x_n) = \wedge_{i \in S} l_i$ for some $S \subseteq [n]$, where each literal l_i is either x_i or $\neg x_i$. Let n be the length of a pattern with exactly m wildcards. Thus, $\mid S \mid= n - m$. A conjunction corresponds to a pattern $\boldsymbol{pat} \in \{0, 1, *\}^n$ one by one. We use a pattern as one input of the obfuscation algorithm instead of one conjunction.

2. Distributional VBB obfuscation

This work considers the distributional virtual black-box (VBB) obfuscation. It can be traced back to Brakerski and Rothblum's work in [25], which is denoted by average-case secure virtual black-box obfuscation. Informally, obfuscation is a distributional VBB obfuscation when the adversary will not be able to distinguish an obfuscation of a uniformly picked pattern from an obfuscation of a function that always outputs 0. In other words, the adversary can not guess one pattern-matching input with overwhelming probability. We use the definition in [28] as following.

Definition 1 (Distributional VBB obfuscation)

Let $\mathcal{C}_n$ is a set of Boolean circuits with inputs of length n. For all length n, $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ is a family of circuits with polynomial size. We denote a probabilities polynomial-time (PPT) algorithm by Obf. Obf takes some circuits $C \in \mathcal{C}_n$ as inputs and outputs another Boolean circuit $\mathcal{O} = Obf(C)$. Assume that each input circuit $C \in \mathcal{C}_n$ is samples following on a distribution $\mathcal{D}_n$ for each n. Let $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ be an ensemble of distribution families $\mathcal{D}_n$. The algorithm Obf is a distributional virtual black-box obfuscator for the distribution class $\mathcal{D}$ over the circuit family $\mathcal{C}$ if it has the following properties:

1) Functionality preservation (FP): It has three variants:

- The algorithm Obf satisfies the weak FP if and only if

$$Pr[\mathcal{O}(x) = C(x)] = 1 - \text{negl}(n)$$

for every $n \in \mathbb{N}$, $C \in \mathcal{C}_n$, and $x \in \{0, 1\}^n$.

- The algorithm Obf satisfies the strong FP if and only if

$$Pr[\mathcal{O}(x) = C(x)\ \forall x \in \{0, 1\}^n] = 1 - \text{negl}(n)$$

for every $n \in \mathbb{N}$, $C \in \mathcal{C}_n$, there exists a negligible function $\text{negl}(n)$ such that

2) Efficiency: The complexity of $\mathcal{O} = Obf(C)$ is $\text{poly}(|C|, n)$ for every $n \in \mathbb{N}$ and $C \in \mathcal{C}_n$.

3) Distributional virtual black-box: For every $n \in \mathbb{N}$, every distribution $D \in \mathcal{D}_n$, and every predicate $\mathcal{P} : \mathcal{C}_n \to \{0, 1\}$, any PPT adversary $\mathcal{A}$ wins the following game with negligible advantage:

$$\begin{aligned} | \underset{C\leftarrow\mathcal{D}_n}{Pr}&[\mathcal{A}(Obf(C, 1^n)) = \mathcal{P}(C)] - \\ &\underset{C\leftarrow\mathcal{D}_n}{Pr}[\mathcal{S}^C(1^{|C|}, 1^n) = \mathcal{P}(C)]| \end{aligned}$$

where $\mathcal{S}$ is a (non-uniform) polynomial size simulator.

Remark 1 Some early conjunction obfuscations with strong functionality preservation are under the stronger assumption [23], [24], [27], [28]. The BKM18 and BLM19-1 achieve it by broadening the generic group of sufficiently large size 2^{2n}. For each input, a false acceptance will be accepted with probability $1/p$. The union bound over each of 2^n inputs implies strong functionality preservation. Our scheme uses the same technique to improve the correctness, which is explained in Section III.2.

Considering the distributional VBB security, we use the similar proof technique of the second construction in [28]. Barak *et al.* proved that perfect-circuit hiding security is equivalent to distributional virtual black-box security [16], i.e. property 3) in Definition 1. We can prove the perfect circuit-hiding security instead of the distributional virtual black-box security depended on this property. Let us recall the definition of perfect circuit-hiding [16] as Definition 2.

Definition 2 (Perfect circuit-hiding) Let $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ be a collection of polynomial-size circuits such that every $C \in \mathcal{C}_n$ maps n input bits to a single output bit. An obfuscator Obf for a circuit collection $\mathcal{C}$ is perfect circuit-hiding if for every PPT adversary $\mathcal{A}$ there exists a negligible function $\text{negl}(n)$ such that for every balanced predicate $\mathcal{P}$, every $n \in \mathbb{N}$ and every auxiliary input $z \in \{0,1\}^{\text{poly}(n)}$ to $\mathcal{A}$:

$$\underset{C\leftarrow\mathcal{C}_n}{Pr}[\mathcal{A}(z, Obf(C)) = \mathcal{P}(C)] \leq \frac{1}{2} + \text{negl}(n)$$

where the probability is also over the randomness of Obf.

3. Linear codes

Roughly speaking, an $[n,k]_q$ linear code maps a message in finite field $\mathbb{F}_q^{k\times n}$ into a codeword $\mathbb{F}_q^n$. Here is the formal definition [30].

Definition 3 (Error-correcting code) For any length n and dimension k, let a (linear) error-correcting code $\mathcal{C}$ over a finite field $\mathbb{F}_q$ be a k-dimensional vector subspace of the n-dimensional vector space $\mathbb{F}_q^n$. A code with these properties is called an $[n,k]$ code. A binary code $\mathcal{C}$ implies that $q = 2$, otherwise it is denoted by a q-ary code.

The code $\mathcal{C}$ can be defined by a generator matrix $\boldsymbol{G}$. Here $\boldsymbol{G}$ is a full rank $k \times n$ matrix over field $\mathbb{F}_q$, and it defines the linear map from message space to the code space. Namely, the code $\mathcal{C}$ can be written as

$$\mathcal{C} = \mathcal{C}(\boldsymbol{G}) = \{\boldsymbol{x}\boldsymbol{G} \mid \boldsymbol{x} \in \mathbb{F}_q^k\}$$

If $\mathcal{C}$ is the kernel of a matrix $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times k}$ where $\boldsymbol{G}\boldsymbol{H}^\top = 0$, we call $\boldsymbol{H}$ a parity check matrix of $\mathcal{C}$. Then the code $\mathcal{C}$ can be denoted as

$$\mathcal{C} = \mathcal{C}^\perp(\boldsymbol{H}) = Ker(\boldsymbol{H}) = \{\boldsymbol{y} \in \mathbb{F}_q^n | \boldsymbol{H}\boldsymbol{y} = \boldsymbol{0}\}$$

The vectors in $\mathcal{C}$ are called codewords.

The Hamming weight $wt(\boldsymbol{c})$ of a codeword $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{F}_q^n$ is defined to be the number of non-zero coordinates, i.e. $wt(\boldsymbol{c}) = |\{i \mid c_i \neq 0, 1 \leq i \leq n\}|$. We use $d(\boldsymbol{c}_1, \boldsymbol{c}_2)$ to denote the distance of two codewords $\boldsymbol{c}_1, \boldsymbol{c}_2$. The $d(\boldsymbol{c}_1, \boldsymbol{c}_2)$ counts the number of coordinates in which they differ. If the sender sends a codeword $\boldsymbol{c}$ but the receiver gets a word $\boldsymbol{c}' = \boldsymbol{c} + \boldsymbol{e}$, then we call $\boldsymbol{e}$ the error vector. Let $\{i|e_i \neq 0\}$ be the set of error positions and $wt(\boldsymbol{e})$ be the number of errors of the received word.

Security Assumption There are many hard problems in coding theory, such as syndrome decoding problem, code equivelence problem, etc. One of the well-known problems is general decoding problem. A formal definition of general decoding problem is as follows:

Definition 4 (General decoding problem) Let ω be a norm over $\mathcal{R}$. On input $(\boldsymbol{G}, \boldsymbol{y}^\top) \in \mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{(n-k)}$ from the uniform distribution, the general decoding problem $GD(n,k,\omega)$ asks to find $\boldsymbol{m} \in \mathbb{F}^k$ such that $\boldsymbol{m}\boldsymbol{G} + \boldsymbol{e} = \boldsymbol{y}^\top$ and $wt(\boldsymbol{e}) = \omega$.

This problem for binary codes and its dual version, i.e. syndrome decoding problem, were proved to be NP-complete in 1978 [31]. Then, it is proven to be NP-complete for any finite fields in 1994 [32].

III. Obfuscating Conjunctions

Having defined what is meant by distributional VBB obfuscation and the security assumption, we will now move to show our construction of conjunction obfuscation. In addition, it is necessary to analyse its correctness, time complexity and security. The rest of this section will show our scheme satisfying the strong functionality preservation and the distributional VBB security.

1. Construction

For any pattern $\boldsymbol{pat} \in \{0,1,*\}^n$ with m wildcards, our conjunction obfuscation is based on a $[2n, n-k]$ random linear code $\mathcal{C}$ over $\mathbb{F}_q$. The parameter k is a constant which only affects the complexity of our obfuscator. We set $k = 1$ as usually. There is a detail about how to pick these parameters securely in Section IV.

Informally, our obfuscator takes security parameter λ, a pattern $\boldsymbol{pat}$ with length n as input, and it runs Algorithm 1 to output an obfuscated program $\mathcal{O}$ which consists of a generator matrix $\boldsymbol{G}$ and a codeword $\boldsymbol{c}$. The Algorithm 1 sample a generator matrix $\boldsymbol{G}$ and a message $\boldsymbol{m}$ randomly. And then, each error $e_{2i-(1-b)}$ is set by a random element in $\mathbb{F}_q$ when $\boldsymbol{pat}_i = b$ for all $i \in [n]$. Given the random generator $\boldsymbol{G}$, the random message $\boldsymbol{x}$ and the error vector $\boldsymbol{e}$, it is easy to compute a codeword $\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G} + \boldsymbol{e}$. Thus the obfuscated program is a random linear codeword which the pattern is hid in the error vector.

The evaluation takes an input string $\boldsymbol{x} \in \{0,1\}^n$ and the obfuscated program $\mathcal{O}$ as inputs, and it runs Algorithm 2 to return the output of the conjunction $f_{\boldsymbol{pat}}(\boldsymbol{x})$. It chooses the columns of matrix $\boldsymbol{G}$ and elements of codeword $\boldsymbol{c}$ corresponding to the input $\boldsymbol{x}$. For every $i \in [n]$, we picks the column $\boldsymbol{G}_{2i-x_i}$ and the element c_{2i-x_i} satisfies $c_{2i-x_i} = \boldsymbol{m}\boldsymbol{G}_{2i-x_i} + e_{2i-x_i}$. Let $G[i][j]$ denote the element located in i-th row and j-th column of matrix $\boldsymbol{G}$. There are n linear equations

$$\begin{cases} c_{2-x_1} = m_1G[1][2-x_1] + m_2G[2][2-x_1] \\ \qquad + \cdots + m_{n-1}G[n-1][2-x_1] \\ c_{4-x_2} = m_1G[1][4-x_2] + m_2G[2][4-x_2] \\ \qquad + \cdots + m_{n-1}G[n-1][4-x_2] \\ \qquad \vdots \\ c_{2n-x_n} = m_1G[1][2n-x_n] + m_2G[2][2n-x_n] \\ \qquad + \cdots + m_{n-1}G[n-1][2n-x_n] \end{cases}$$

where only the message $\boldsymbol{m} = (m_1, m_2, \ldots, m_{n-1})$ is unknown. Because the message $\boldsymbol{m}$ can be resumed by $n-1$ linear equations, we picks two subsets which consists of $n-1$ linear equations among them. The algorithm solves these two linear equations to get messages $\boldsymbol{m_1}$ and $\boldsymbol{m_2}$, and it outputs 1 if there exists $\boldsymbol{m_1} = \boldsymbol{m_2}$.

Algorithm 1 *Obf* $(1^n, \boldsymbol{pat} \in \{0,1,*\}^n)$

```
Input: An integer $n$; The pattern $\boldsymbol{pat} \in \{0,1,*\}^n$.
Output: The obfuscated program $\mathcal{O}$.
Sample a $[2n, n-k]$ random linear code $\mathcal{C}$ over $\mathbb{F}_q$ with generator $\boldsymbol{G} \leftarrow \mathbb{F}_q^{(n-1)\times 2n}$
Sample a random message $\boldsymbol{m} \leftarrow \mathbb{F}_q^{n-1}$
for each $i \in [n]$ do :
    if $\boldsymbol{pat}_i = *$ then
        Set $e_{2i-1} = e_{2i} = 0$;
    else
        Set $e_{2i-b} = 0$ and $e_{2i-(1-b)} \leftarrow \mathbb{F}_q$.
Generate a codeword $\boldsymbol{c} = (c_1, c_2, \ldots, c_{2n})$ from $\boldsymbol{c} = \boldsymbol{mG} + \boldsymbol{e}$.
Return $\mathcal{O} = (\boldsymbol{G}, \boldsymbol{c})$.
```

Algorithm 2 *Eval*$(\mathcal{O}, \boldsymbol{x} \in \{0,1\}^n)$

```
Input: A string $\boldsymbol{x} \in \{0,1\}^n$; The obfuscated program $\mathcal{O}$.
Output: A bit 0 or 1.
Split the obfuscated program to $\mathcal{O} = (\boldsymbol{G}, \boldsymbol{c})$.
Randomly pick two constants $k_1, k_2 \in [n]$ and $k_1 \neq k_2$.
Define $\boldsymbol{c}_{\mathbf{x},k}$ to be the $n-1$-length vector where it consists of elements $c_{2i-x_i}$ for all $i \in [n]$ and $i \neq k$.
Define $\boldsymbol{G}_{\boldsymbol{x},k}$ to be the $(n-1)\times(n-1)$ matrix where it consists of vectors $(\boldsymbol{G})_{2i-x_i}$ for all $i \in (n-1)$ and $i \neq k$.
Solve $\boldsymbol{m}_1\boldsymbol{G}_{\boldsymbol{x},k_1} = \boldsymbol{c}_{\boldsymbol{x},k_1}$ and $\boldsymbol{m}_2\boldsymbol{G}_{\boldsymbol{x},k_2} = \boldsymbol{c}_{\boldsymbol{x},k_2}$.
If there exists $\boldsymbol{m}_1 = \boldsymbol{m}_2$, Return 1.
Return 0 otherwise.
```

2. Functionality preservation

Before considering the correctness of our construction, let us recall the probability of the rank of a random matrix over a finite field. Cooper [33] considered a space of random $(n \times n)$ matrix over $\mathbb{F}_q$ which is denoted by $\mathbb{M}(n,p;q)$. Let $\boldsymbol{G}$ be a $n \times n$ matrix with entries in $\mathbb{F}_q$, which are independently and identically distributed as

$$Pr[G[i][j] = r] = \begin{cases} 1-p, & r = 0 \\ \dfrac{p}{q-1}, & r \in [q-1] \end{cases}$$

The uniformly random matrix implies that $p = (q-1)/q$. If there are l columns of the matrix $\boldsymbol{G}$ that are linearly independent, an extra random column is linearly related to the l columns with probability q^l/q^n. Thus

$$Pr[\boldsymbol{G} \text{ is non-singular}] = \prod_{l=1}^{n}(1-q^{-l})$$

Given $q > 2^n$, any $n-1$ columns of an $(n-1)\times 2n$ matrix $\boldsymbol{G}$ is non-singular with probability $1 - \mathrm{negl}(n)$.

As described above, the obfuscated program consists of a codeword $\boldsymbol{c} = \boldsymbol{mG} + \boldsymbol{e}$. If $\boldsymbol{x} \in \{0,1\}^n$ is a matching input of the pattern $\boldsymbol{pat} \in \{0,1,*\}^n$, then $e_{2i-x_i} = 0$ for all $i \in [n]$. What's more, we have

$$\boldsymbol{c}_{\boldsymbol{x},k} = \boldsymbol{mG}_{\boldsymbol{x},k}$$

for any $k \in [n]$. Considering any $k_1, k_2 \in [n]$, there must be $\boldsymbol{m_1} = \boldsymbol{m_2}$ where $\boldsymbol{c}_{\boldsymbol{x},k_1} = \boldsymbol{m}_1\boldsymbol{G}_{\boldsymbol{x},k}$ and $\boldsymbol{c}_{\boldsymbol{x},k_2} = \boldsymbol{m_2}\boldsymbol{G}_{\boldsymbol{x},k}$ if the matrices $\boldsymbol{G}_{\mathbf{x},k_1}$ and $\boldsymbol{G}_{\mathbf{x},k_2}$ are non-singular. Thus, the evaluation algorithm outputs 1 with probability $1 - \mathrm{negl}(n)$.

On the other hand, we move to discuss the evaluation algorithm outputs 1 when $\boldsymbol{x}$ does not match the pattern $\boldsymbol{pat}$ (It is a false acceptance.). The existence of $\boldsymbol{m}_1 = \boldsymbol{m}_2$ implies that there is a solution to the linear equations

$$\boldsymbol{m}_1\boldsymbol{G}_{\boldsymbol{x}} = \boldsymbol{c}_{\boldsymbol{x}} \tag{1}$$

where $\boldsymbol{G}_{\boldsymbol{x}}$ consists of the columns $(\boldsymbol{G})_{2i-x_i}$ and $\boldsymbol{c}_{\boldsymbol{x}}$ consists of the elements c_{2i-x_i} for all $i \in [n]$. As we know, the equation $\boldsymbol{m}_1\boldsymbol{G}_{\boldsymbol{x}} = \boldsymbol{c}_{\boldsymbol{x}}$ has a solution if and only if $\boldsymbol{c}_{\boldsymbol{x}}$ is a linear combination of the rows of $\boldsymbol{G}_{\boldsymbol{x}}$. We have that

$$\boldsymbol{c}_{\boldsymbol{x}} = \boldsymbol{mG}_{\boldsymbol{x}} + \boldsymbol{e}_{\boldsymbol{x}} \tag{2}$$

from the obfuscated program $\mathcal{O}(\boldsymbol{G}, \boldsymbol{c})$. The vector $\boldsymbol{e}_{\boldsymbol{x}}$ denotes the elements e_{2i-x_i} for $i \in [n]$. Combining (1) and (2), it implies that

$$(\boldsymbol{m}_1 - \boldsymbol{m})\boldsymbol{G}_{\boldsymbol{x}} = \boldsymbol{e}_{\boldsymbol{x}} \tag{3}$$

Now we only need to consider the situation that the vector $\boldsymbol{e}_{\boldsymbol{x}}$ is a linear combination of the rows of $\boldsymbol{G}_{\boldsymbol{x}}$. (The following analysis is based on the $(n-1)\times n$ matrix $\boldsymbol{G}_{\boldsymbol{x}}$ has rank $n-1$.) The $(n-1)\times n$ matrix $\boldsymbol{G}_{\boldsymbol{x}}$ is row reduced to $\begin{pmatrix} \boldsymbol{I}_{n-1} & \boldsymbol{g} \end{pmatrix}$ where $\boldsymbol{I}_{n-1}$ is a $(n-1)\times(n-1)$ identity matrix and $\boldsymbol{g} \in \mathbb{F}_q^n$. Now $\sum_{i=1}^{n}(\boldsymbol{e}_{\boldsymbol{x}})_i g_i = 0 \mod q$ means that $\boldsymbol{e}_{\boldsymbol{x}}$ is a linear combination of the rows of $\boldsymbol{G}_{\boldsymbol{x}}$. The generator matrix is generated randomly which implies the distribution of vector $\boldsymbol{g}$ is a uniform distribution. Thus,

$$Pr\left[\sum_{i=1}^{n}(\boldsymbol{e_x})_i g_i = 0 \mod \mathrm{q} \mid \mathrm{G_x} \text{ is non-singular}\right] = 1/q$$

In other words, the probability of a false acceptance is $1/q \cdot (1 - \mathrm{negl}(n))$.

Remark 2 The false acceptance only consider the situation that the matrix $\boldsymbol{G_x}$ has rank $n-1$. First of all, the matrix $\boldsymbol{G_x}$ has rank $n-1$ with probability $1-\mathrm{negl}(n)$. Even all the false inputs under a matrix $\boldsymbol{G_x}$ with non-full rank is a false acceptance, the probability of a false acceptance is only $1/q \cdot (1-\mathrm{negl}(n)) + \mathrm{negl}(n)$. It implies our scheme also satisfies the weak functionality preservation.

As noted in [27], we can boost this to strong functionality preservation by setting $q > 2^{2n}$. With the union bound over each of 2^n inputs, our obfuscated program is simultaneous correct with overwhelming probability on all inputs.

3. Efficiency

For the obfuscating algorithm, we need a $[2n, n-1]$ random linear code $\mathcal{C}$ with a generator matrix $\boldsymbol{G}$. These can be pre-computed and we overlook them. Next, it picks some random element from $\mathbb{F}_q$ and sets them on $e_{1-(1-b)}$ when $\boldsymbol{pat}_i = b$. Finally, the algorithm generates a codeword $\boldsymbol{c}$ by $\boldsymbol{c} = \boldsymbol{mG} + \boldsymbol{e}$. Therefore the time complexity of the obfuscation is $\mathcal{O}(n^2)$ in the number of operations in $\mathbb{F}_q$.

For the time complexity of the evaluation algorithm, it involves two $(n-1)\times(n-1)$ matrices $\boldsymbol{G}_{\boldsymbol{x},k_1}$,$\boldsymbol{G}_{\boldsymbol{x},k_2}$ and two $(n-1)$-length vectors $\boldsymbol{c}_{\boldsymbol{x},k_1}$, $\boldsymbol{c}_{\boldsymbol{x},k_2}$. And then, it needs to solve linear equations in time $\mathcal{O}(n^3)$ and determine $\boldsymbol{m}_1 = \boldsymbol{m}_2$ or not. Thus, The time complexity of the evaluation is $\mathcal{O}(n^3)$ in the number of operations in $\mathbb{F}_q$.

4. Security analysis

The rest of this section will discuss its security. We will show the obfuscator constructed in Section III.1 satisfies distributional virtual black-box security. Recall from [16] that perfect circuit-hiding security is equivalent to distributional virtual black-box security. Hence, we are going to prove that our construction satisfies perfect circuit-hiding security.

Theorem 1 Assuming the hardness of the general decoding problem

$$Pr\left[\mathcal{A}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}) = \boldsymbol{m}\right] \le \mathrm{negl}(\lambda)$$

with some negligible function $\mathrm{negl}(\lambda)$, the above obfuscation satisfies the perfect circuit-hiding for Boolean functions $f_{\boldsymbol{pat}}$ where $\boldsymbol{pat} \leftarrow \mathcal{U}_{n,m}$.

Proof For any predicate $\mathcal{P}: \mathbb{F}_q^{2n} \to \{0,1\}$ and any polynomial-time algorithm $\mathcal{B}$, we will prove that

$$\mid Pr[\mathcal{B}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e})) = \boldsymbol{m}] \le \mathrm{negl}(\lambda)$$

relying on the hardness of the general decoding problem.

Suppose an adversary can recover the message $\boldsymbol{m}$ with non-negligible possibility, we can built an adversary $\mathcal{A}$ who break the hardness assumption of the general decoding problem. The adversary receives the challenge $(\boldsymbol{G}, \boldsymbol{y})$ where $\boldsymbol{y} = \boldsymbol{mG}+\boldsymbol{e}$, then it simulates the view $\mathcal{B}$ as follows.

- Randomly choose one bit $s \in \{0,1\}$;
- Send the challenge $(\boldsymbol{G}, \boldsymbol{y}, s)$ to $\mathcal{B}$ and receive $\boldsymbol{m}_1$;
- Send the challenge $(\boldsymbol{G}, \boldsymbol{y}, 1-s)$ to $\mathcal{B}$ and receive $\boldsymbol{m}_2$;
- Choose one message $\boldsymbol{m}$ from $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ where $wt(\boldsymbol{y}-\boldsymbol{mG}) = n-m$;

This implies that

$$\begin{aligned}
&Pr\left[\mathcal{A}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}) = \boldsymbol{m}\right] \\
&\quad= Pr\left[\mathcal{B}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, s) = \boldsymbol{m}\right] \\
&\qquad+ Pr\left[\mathcal{B}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, 1-s) = \boldsymbol{m}\right] \\
&\quad= Pr\left[\mathcal{B}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e})) = \boldsymbol{m}\right] \\
&\qquad+ Pr\left[\mathcal{B}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, 1-\mathcal{P}(\boldsymbol{e})) = \boldsymbol{m}\right] \\
&\quad= 2Pr\left[\mathcal{B}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e})) = \boldsymbol{m}\right]
\end{aligned}$$

This is a contraction.

The next step is a search-to-decision reduction. We have that

$$\mid Pr[\mathcal{B}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e})) = \boldsymbol{m}] \le \mathrm{negl}(\lambda)$$

By the Goldreich-Levin hardcore bit theorem, given $(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}))$ and a random $(n-1)$-bit vector $\boldsymbol{r}$, an efficient adversary cannot compute $\langle \boldsymbol{r}, \boldsymbol{m}\rangle \bmod 2$ with probability greater than $\frac{1}{2} + \mathrm{negl}(\lambda)$

Suppose an adversary $\mathcal{C}$ can distinguish $(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}))$ and $(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e}))$ with non-negligible advantage p such that

$$\mid Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e})) = 1] - Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e})) = 1] \mid$$

is equal to $p(\lambda)$. We will build an adversary $\mathcal{N}$ can break the Goldreich-Levin hardcore bit theorem. The adversary $\mathcal{N}$ receives the challenge $(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r})$, then it similates the view $\mathcal{M}$ as follows.

- compute a new generation matrix $\bar{\boldsymbol{G}} = \boldsymbol{G} - \boldsymbol{r}^{\top} \cdot \boldsymbol{s}$ where $\boldsymbol{s}$ is random $(n-1)$-bit vector;
- Send the challenge $(\bar{\boldsymbol{G}}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e}))$ to $\mathcal{N}$.

When $\langle \boldsymbol{r}, \boldsymbol{s}\rangle = 0 \mod 2$ it holds that $(\bar{\boldsymbol{G}}, \boldsymbol{y}) = (\bar{\boldsymbol{G}}, \boldsymbol{m}\bar{\boldsymbol{G}}+\boldsymbol{e})$ and when $\langle \boldsymbol{r}, \boldsymbol{s}\rangle = 1 \mod 2$ we have that $(\bar{\boldsymbol{G}}, \boldsymbol{y}) \equiv (\bar{\boldsymbol{G}}, U_{\boldsymbol{m}})$ where $U_{\boldsymbol{m}}$ is independent of $\bar{\boldsymbol{G}}$. This implies that

$$
\begin{aligned}
&Pr[\mathcal{N}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = \langle \boldsymbol{r}, \boldsymbol{m}\rangle] \\
&= Pr[\mathcal{N}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 0 \mid \langle \boldsymbol{r}, \boldsymbol{m}\rangle = 0 \bmod 2] \\
&\quad \cdot Pr[\langle \boldsymbol{r}, \boldsymbol{m}\rangle = 0 \bmod 2] \\
&\quad + Pr[\mathcal{N}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 1 \mid \langle \boldsymbol{r}, \boldsymbol{m}\rangle = 1 \bmod 2] \\
&\quad \cdot Pr[\langle \boldsymbol{r}, \boldsymbol{m}\rangle = 1 \bmod 2] \\
&= \frac{1}{2} Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 0] \\
&\quad + 1 - \frac{1}{2} Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 0] \\
&= 1 - \frac{1}{2}(Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 0] \\
&\quad - Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 0])
\end{aligned}
$$

Whenever $Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{e}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 0] \geq Pr[\mathcal{M}(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e}), \boldsymbol{r}) = 0]$ or not, it implies that the adversary $\mathcal{N}$ can guess $\langle \boldsymbol{r}, \boldsymbol{m}\rangle$ with non-negligible advantage. This is a contraction.

Finally, we will show that for any probabilistic polynomial-time adversary $\mathcal{B}'$ and any balanced predicate $\mathcal{P} : \{0, 1, *\}^n \to \{0, 1\}$ (that is, $\mathcal{P}$ takes the values 0 and 1 with probability 1/2 over the randomness of $\boldsymbol{pat} \leftarrow \mathcal{U}_{n,n-m}$),

$$\underset{C \leftarrow \mathcal{C}_n}{Pr}[\mathcal{B}'(Obf(C)) = \mathcal{P}(C)] \leq \frac{1}{2} + \mathrm{negl}(n)$$

The above proof completes

$$
\begin{aligned}
&\mid Pr[\mathcal{B}(\mathcal{O}(\boldsymbol{pat}), \mathcal{P}(\boldsymbol{pat})) = 1] \\
&\quad - Pr[\mathcal{B}(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e})) = 1] \mid \leq \mathrm{negl}(\lambda)
\end{aligned}
$$

Assumed that the algorithm $\mathcal{B}'$ can guess the output of $\mathcal{P}(\boldsymbol{pat})$ with significant advantages, the adversary $\mathcal{B}$ invokes $\mathcal{B}'$ on $(\boldsymbol{G}, \boldsymbol{y})$ and outputs 1 if $\mathcal{B}'(\boldsymbol{G}, \boldsymbol{y}) = \mathcal{P}(\boldsymbol{pat})$ and 0 otherwise. If $(\boldsymbol{G}, \boldsymbol{y}) = \mathcal{O}(\boldsymbol{pat})$, then $\mathcal{B}$ outputs 1 with the same advantages as $\mathcal{B}'$'s. If $(\boldsymbol{G}, \boldsymbol{y})$ is uniformly random, $\mathcal{B}'(\boldsymbol{G}, \boldsymbol{y})$ is independent of $\mathcal{P}(\boldsymbol{pat})$. The balanced predicate $\mathcal{P}$ implies $\mathcal{B}$ outputs 1 with probability exactly 1/2. Thus, the adversary $\mathcal{B}$ can distinguish $(\mathcal{O}(\boldsymbol{pat}), \mathcal{P}(\boldsymbol{pat}))$ from $(\boldsymbol{G}, \boldsymbol{y}, \mathcal{P}(\boldsymbol{e}))$ with non-negligible probability. This a contradiction, which implies our scheme satisfies the perfect input-hiding security.

IV. Attack Resistance and Parameter Analysis

The following is a brief description of the related attacks and parameter analysis. We consider the information set decoding attack in Section IV.1 and the structured error attack in Section IV.2, which uses the parameters in Table 2 . In Section IV.3, we conclude these parameter constraints and provide the suggested parameters under 80-bit and 128-bit security.

Table 2. The Parameters

Parameter	Definition
λ	Security parameter
n	The length of a pattern $\boldsymbol{pat} \in \{0, 1, *\}^n$
m	The number of wildcards $\mid \{i \mid \boldsymbol{pat}_i = *\} \mid$
$\mathbb{F}_q$	A finite field
k	The length of a message $\boldsymbol{m} \in \mathbb{F}_q^{n-k}$
$[n_c, k_c]$	A linear $[n_c, k_c]$ code where $n_c = 2n$ and $k_c = n - k$
t_c	The weight of a error $\boldsymbol{e} \in \mathbb{F}_q^{2n}$ where $t_c = n - m$

1. Information set decoding

Information set decoding is one of the most famous decoding attacks towards code-based cryptography.

ISD algorithm was introduced by Prange [34] at first. Its idea is to find a set of coordinates of a garbled vector which are error-free. The restriction of the code's generator matrix to these positions should be invertible. Then, the message vector can be computed by multiplying the received vector by the inverse of the submatrix. The Prange's algorithm focused on the finite field $\mathbb{F}_2$. Soon, Peters [35] generalised Prange's ISD algorithm over $\mathbb{F}_2$ to $\mathbb{F}_q$. What's more, a lower bound for ISD algorithm was showed by Niebuhr *et al.* [36].

Let n_c be the length of the code $\mathcal{C}$ over $\mathbb{F}_q$, k_c be the dimension and $r_c = n_c - k_c$ be the co-dimension. To correct t_c errors, the lower bound for the expected cost in the binary operation of the algorithm is

$$
\begin{aligned}
&WF_{qISD}(n_c, k_c, t_c, q) \\
&= \min_{p} \left\{ \frac{1}{\sqrt{q-1}} \cdot \frac{2l \min\left(\binom{n_c}{t_c}(q-1)^{t_c}, q^{r_c}\right)}{\lambda_q \binom{r-l}{t_c-p} \binom{k_c+l}{p} (q-1)^{t_c}} \right. \\
&\quad \left. \cdot \sqrt{\binom{k_c+l}{p}(q-1)^p} \right\}
\end{aligned}
$$

where $l = \log_q(K_q \lambda_q \sqrt{\binom{k_c}{p}(q-1)^{p-1}} \cdot \ln(q)/2)$ and $\lambda_q = 1 - \exp(-1) \approx 0.63.$

Noticed that the function above is associated with n_c, k_c, and t_c very tightly, but does not affect by the size of q when q is large enough. There are also some other works on ISD over an arbitrary finite field $\mathbb{F}_q$ inspired by [37], such as [38] and [39]. However, the time complexity of all these works grows fast with the growing of q. Thus the work in [36] still remains the most efficient ISD attack against our work.

Example 1 To reach 2^{80} security level and the pattern length $n = 50$ with $m = 2$ wildcards, we choose parameters $q=1125899906842679$ and $[n_c, k_c]=[100, 49]$, then $r_c = n_c - k_c = 51$. Set $t_c = 48$. Take them into the equation above, we found that when $p = 1$ the right side of the equation gets the minimum value.

$$\begin{aligned}
&WF_{qISD}(100, 49, 48, 1125899906842679)\\
&= \frac{1}{\sqrt{q-1}} \cdot \frac{2l\binom{n_c}{t_c}}{\lambda_q\binom{r_c-l}{t_c-1}\binom{k_c+l}{1}} \cdot \sqrt{\binom{k_c+l}{1}(q-1)}\\
&= \frac{1}{1125899906842678} \cdot \frac{2l\binom{100}{48}}{\lambda_q\binom{51-l}{48}}(49+l)\\
&\quad \cdot\sqrt{\binom{49+l}{l}1125899906842678}\\
&\approx 2^{80.8205}
\end{aligned}$$

here $l = \log_q(K_q\lambda_q\sqrt{\binom{k}{p}(q-1)^{p-1}} \cdot \ln(q)/2) \approx 0.2568$.

2. Structured error attack

In 2011, Arora and Ge proposed a structed error attack on LPN assumption and LWE assumption [40]. An structured error is associated with a non-zero degree-d polynomial $P(e_1, e_2, \dots, e_n)$ over $\mathbb{F}_2$. For any error $\boldsymbol{e} \in \mathbb{F}_2^n$, the polynomial $P(e_1, e_2, \dots, e_n)$ is 0. As we known, an $[n_c, k_c]$ − codeword $\boldsymbol{c}$ is encoded as $\boldsymbol{c} = \boldsymbol{mG}$ $+\boldsymbol{e}$ where $\boldsymbol{G} = [\boldsymbol{g}_1\ \boldsymbol{g}_2\ \dots\ \boldsymbol{g}_{n_c}]$ and $\boldsymbol{g}_i \in \mathbb{F}_2^k$. It implies $P(c_1 - \boldsymbol{mg}_1, \dots, c_{n_c} - \boldsymbol{mg}_{n_c}) = 0$. After linearization by replacing the monomial $\prod_{i\in S}\boldsymbol{m}_i$ by $y_S = \prod_{i\in S}\boldsymbol{m}_i$ for all $S \subseteq [n_c]$, it can be viewed as a linear equation. If there are enough codewords, we can build N linear equations under $N = \sum_{i=0}^{d}\binom{n}{i} - 1$ variables.

Putting it to our scheme, our error vector also has structured. For any $i \in [n]$, there must be a zero in e_{2i-1} or e_{2i}. Thus, the polynomial is built as

$$P(e_{2i-1}, e_{2i}) = e_{2i-1} \cdot e_{2i} = 0$$

which is equal to

$$\begin{aligned}
&P(c_{2i-1} - \boldsymbol{mg}_{2i-1}, c_{2i} - \boldsymbol{mg}_{2i})\\
&= (c_{2i-1} - \boldsymbol{mg}_{2i-1}) \cdot (c_{2i} - \boldsymbol{mg}_{2i})\\
&= 0
\end{aligned}$$

It is a polynomial of degree 2 on $m_1, m_2, \dots, m_{n-k}$ given the generator matrix $\boldsymbol{G}$ and the codeword $\boldsymbol{c}$. The linearization adds $\binom{n-k}{2}$ variables $y_{i,j} = m_i \cdot m_j$ where $i, j \in [n]$. If there are $N = \binom{n-k}{2} + n - k$ equations, the message will be computed by solving linear equations. Luckily, our obfuscated program only outputs the generator $\boldsymbol{G}$ and a codeword $\boldsymbol{c} = (c_1, c_2, \dots, c_{2n-1}, c_{2n})$ which implies n linear equations. Thus, to resist this attack, our parameters need to satisfy $n - k + \binom{n-k}{2} > n$.

3. Parameter analysis

As discussed above, our parameters need to satisfy the following requirements.

- $m = cn$ where $0 < c < 1$.
- $q > 2^n$ for weak functionality preservation and $q > 2^{2n}$ for strong functionality preservation.
- For Information Set Decoding over $\mathbb{F}_q$,

$$\begin{aligned}
&WF_{qISD}(n_c, k_c, t_c, q)\\
&= \min_p\left\{\frac{1}{\sqrt{q-1}} \cdot \frac{2l\min\left(\binom{n_c}{t_c}(q-1)^{t_c}, q^r\right)}{\lambda_q\binom{r-l}{t_c-p}\binom{k_c+l}{p}(q-1)^{t_c}}\right.\\
&\qquad\left.\cdot\sqrt{\binom{k_c+l}{p}(q-1)^p}\right\}\\
&\le \mathrm{negl}(\lambda)
\end{aligned}$$

where $l = \log_q\left(K_q\lambda_q\sqrt{\binom{k_c}{p}(q-1)^{p-1}} \cdot \ln(q)/2\right)$ and $\lambda_q = 1 - \exp(-1) \approx 0.63$.

- For structed error attack, $(n - k) + \binom{n-k}{2} > n$.

We give our suggested parameters in Table 3 . To reach the weak or strong functionality preservation, the parameter q is chosen great than 2^n. It is so big that the weight $n - m$ of error shows little weak influence on the size of q (it is $q > 2^n$ or $1 > 2^{2n}$).

Table 3. The suggested parameters of our obfuscation

Parameters	Weak FP				Strong FP			
λ	80		128		80		128	
n	50	100	100	200	50	100	100	200
$\mid q \mid$	50	100	100	200	100	200	200	400
k	1	1	1	1	1	1	1	1
m	2	35	15	95	2	35	15	95

Besides, Fig.1 shows the increase of the maximum parameter m under $k = 1$ and the increase of the maximum parameter k under $m = 0.1n$. Fig.1(a) indicates that a significant increase in the m is recorded following the addition of n. The parameter m can reach $0.708n$ when $n = 250$, which shows the flexibility of the number of wildcards. Fig.1(b) shows the increase in the k following the addition of n. It is easy to learn that the bigger k means the smaller scale of the linear equations used to resume the message $\boldsymbol{m}$. It only need $n - k$

linear equations to compute an $(n-k)$-length message. It means the faster computation in evaluation algorithm. These parameters m and k can be set flexibility according to the actual requirements.

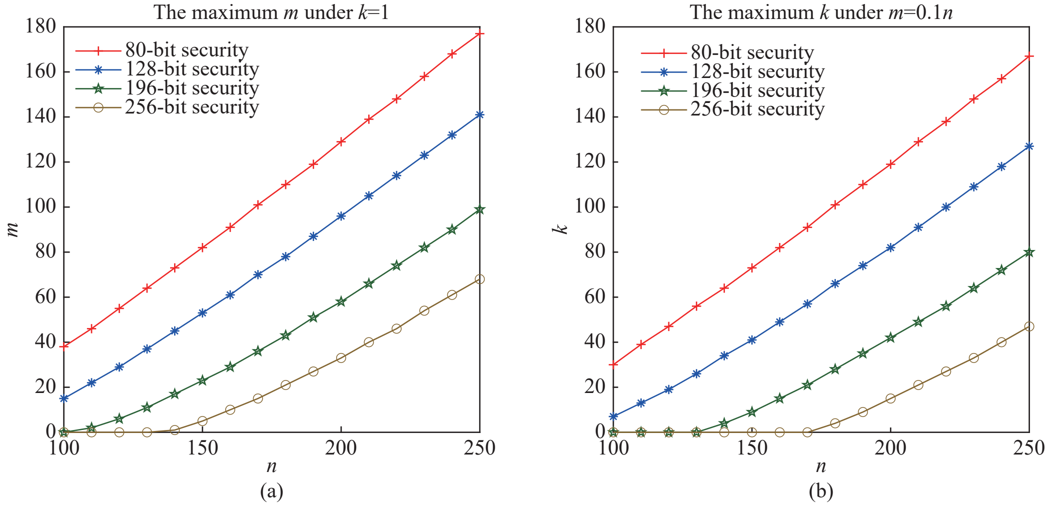

Fig. 1. The growth of parameters m and k.

V. Optimization

Some optimizations are given in this section. We will consider the size of the obfuscated program, the complexity of each evaluation and the functionality such as conjunctions with multi-bit output.

We generally hold that every obfuscated program $\mathcal{O}$, obfuscated by Obf, is stored and calculated many times. The smaller size of the obfuscated program means the smaller storage space and communication size. The lower complexity means smaller computing devices and faster response. Thus the size of the obfuscated program and the complexity of each evaluation are important characteristics of an obfuscator.

Let us first consider the size of obfuscated program. As described on the previous chapter, the obfuscation algorithm outputs a generator matrix $\boldsymbol{G} \in \mathbb{F}_q^{(n-k)\times 2n}$ and a codeword $\boldsymbol{c} \in \mathbb{F}_q^{2n}$. The generator matrix is generated randomly and it can be replaced by hash functions. The obfuscation sample a random seed $\boldsymbol{s} \in \{0,1\}^{\ell_1}$ as an input of a secure hash function $H : \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2}$ as Definition 4. After this, the obfuscated program only include the random seed $\boldsymbol{s}$ instead of a matrix over $\mathbb{F}_q$. The size of the obfuscated program is $| \mathcal{O} |= \ell_1 + 2n \log q$, which is $2n(n-k+1)\log q$ before.

Further, the size of the obfuscated program can be reduced by using pseudorandom generation, pseudorandom function, or random oracle. Different cryptography primitives have different effects on the obfuscated program's size. The actual effect will depend on the specific construction scheme of cryptography primitives.

Definition 5 Given a secure hash function $H : \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2}$ with a random vector $\boldsymbol{s} \leftarrow \{0,1\}^{\ell_1}$ and a function $\sigma : \{0,1\}^{\ell_2} \to \mathbb{F}_q^n$, let $\boldsymbol{G}_{n,k,q,\boldsymbol{s}} \in \mathbb{F}_q^{k\times n}$ be the matrix as follows:

$$\boldsymbol{G}_{n,k,q,\boldsymbol{s}} = \begin{pmatrix} \sigma(H(\boldsymbol{s})) \\ \sigma(H^2(\boldsymbol{s})) \\ \vdots \\ \sigma(H^k(\boldsymbol{s})) \end{pmatrix}$$

Then moving on to consider the complexity of evaluation algorithm. The most time-consuming part is solving linear equations. Our based evaluation needs calculate two vectors $\boldsymbol{m_1}$ and $\boldsymbol{m_2}$ from two linear equations. In fact, this step is used to determine whether the selected codeword $\boldsymbol{c_x} = \boldsymbol{mG_x}$ or not. Without giving the message $\boldsymbol{m}$, we choose two $(n-k)\times(n-k)$ matrices $\boldsymbol{G}_{\boldsymbol{x},k_1}$, $\boldsymbol{G}_{\boldsymbol{x},k_2}$ and $(n-k)$-length vectors $\boldsymbol{c}_{\boldsymbol{x},k_1}$, $\boldsymbol{c}_{\boldsymbol{x},k_1}$ to resume $\boldsymbol{m}$. In order to simplify, we give up two linear equations and only solve the first one $\boldsymbol{c}_{\boldsymbol{x},k_1} = \boldsymbol{mG}_{\boldsymbol{x},k_1}$. And then, it checks $\boldsymbol{c_x} = \boldsymbol{mG_x}$ or not. Furthermore, it also can add an hash value $H(\boldsymbol{m})$ to the obfuscated program and the evaluation outputs 1 if and only if $H(\boldsymbol{m}_1) = H(\boldsymbol{m})$.

In addition, our obfuscation can be extended for the multi-bit conjunctions $f_{\boldsymbol{pat},\boldsymbol{msg}}(\boldsymbol{x})$ which outputs a message $\boldsymbol{msg} \in \{0,1\}^r$ when the input string $\boldsymbol{x} \in \{0,1\}^n$ matches the pattern $\boldsymbol{pat} \in \{0,1,*\}^n$. The length of the message is a polynomial function of n, where $r = \mathrm{poly}(n)$. Given a secure pseudo-random generator $PRG : \{0,1\}^{n-k} \to \{0,1\}^r$, the message can be encrypted by $\widetilde{\boldsymbol{msg}} = PRG(\boldsymbol{m}) \oplus \boldsymbol{msg}$. The evaluation will decrypt it by $\boldsymbol{msg} = PRG(\boldsymbol{m}) \oplus \widetilde{\boldsymbol{msg}}$ after it gets $\boldsymbol{mG_x} = \boldsymbol{c_x}$.

VI. Conclusions

This work have argued that how to realize the obfuscation for conjunctions under random linear code while satisfying the strong functionality preservation. We use a linear code with structured errors instead of a syndrome to construct an obfuscator which decodes the codeword by solving linear equations. The new evaluation outputs 1 on a false input with probability at most $1/q_{\rm negl}(n)$. Thus, it can be boosted to strong functionality preservation by larger field $\mathbb{F}_q$. We give a concrete analysis that our obfuscation can resist the information set decoding attack and the structured error attack. And then some suggested parameters are proposed. In addition, we also discuss how to reduce the size of the obfuscated program and the complexity of the evaluation algorithm. Our obfuscation also can be expanded for the conjunctions with multi-bit outputs.

Our work is a small step on the construction of obfuscation for special functions under the standard model. It provides ideas on how to modify a code-based obfuscation to achieve strong functionality preservation. Code-based cryptography is an important member of post-quantum cryptography and code-based obfuscation also can resist the attacks and threats brought by quantum computers.

References

[1] Y. Zhang, D. He, Y. Li, *et al.*, "Efficient obfuscation for encrypted identity-based Signatures in wireless body area networks," *IEEE System Journal*, vol.14, no.4, pp.5320–5328, 2020.

[2] M. Zhang, Y. Jiang, H. Shen, *et al*, "Cloud-based data-sharing scheme using verifiable and CCA-secure re-encryption from indistinguishability obfuscation," in *Information Security and Cryptology – INSCRYPT 2018, Lecture Notes in Computer Science (vol.11449)*, Springer, Cham, pp.240–259, 2019.

[3] M. Zhang, Y. Zhang, Y. Jiang, *et al.*, "Obfuscating eves algorithm and its application in fair electronic transactions in public cloud systems," *IEEE System Journal*, vol.13, no.2, pp.1478–1486, 2019.

[4] S. Goldwasser, S. D. Gordon, V. Goyal, *et al*, "Multi-input functional encryption," in *Advances in Cryptology – EUROCRYPT 2014, Lecture Notes in Computer Science (vol.8441)*, Springer, Berlin, Heidelberg, pp.578–602, 2014.

[5] D. Boneh and M. Zhandry, "Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation," *Algirithmica*, vol.79, pp.1233–1285, 2017.

[6] A. Sahai and B. Waters, "How to use indistinguishability obfuscation deniable encryption and more," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC'14)*, New York, NY, USA, pp.475–484, 2014.

[7] S. Garg, C. Gentry, S. Halevi, *et al*, "Candidate indistinguishability obfuscation and functional encryption for all circuits," in *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS' 13)*, Berkeley, CA, USA, pp.40–49, 2013.

[8] N. Bitansky and V. Vaikuntanathan, "Indistinguishability obfuscation from functional encryption," in *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS'15)*, Berkeley, CA, USA, pp.171–190, 2015.

[9] P. Ananth and A. Jain, "Indistinguishability obfuscation from compact functional encryption," in *Advances in Cryptology – CRYPTO 2015, Lecture Notes in Computer Science (vol.9215)*, Springer, Berlin, Heidelberg, pp.308–326, 2015.

[10] S. Garg, C. Gentry and S. Halevi, "Candidate multilinear maps from ideal lattices," *Advances in Cryptology – EUROCRYPT 2013, Lecture Notes in Computer Science, (vol.7881)*, Springer, Berlin, Heidelberg, pp.1–17, 2013.

[11] J. S. Coron, T. Lepoint and M. Tibouchi. "Practical multilinear maps over the integers," in *Advances in Cryptology – CRYPTO 2013, Lecture Notes in Computer Science (vol.8042)*, Springer, Berlin, Heidelberg, pp.476-493, 2013.

[12] C. Gentry, S. Gorbunov, and S. Halevi, "Graph-induced multilinear maps from lattices," in *Theory of Cryptography – TCC 2015, Lecture Notes in Computer Science (vol.9015)*, Springer, Berlin, Heidelberg, pp.498–527, 2015.

[13] J. S. Coron, C. Gentry, and S. Halevi, "Zeroizing without low-Level zeroes: New MMAP attacks and their limitations," in *Advances in Cryptology – CRYPTO 2015, Lecture Notes in Computer Science (vol.9215)*, Springer, Berlin, Heidelberg, pp.247–266, 2015.

[14] E. Miles, A. Sahai, and M. Zhandry, "Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13," in *Advances in Cryptology – CRYPTO 2016, Lecture Notes in Computer Science (vol.9815)*, Springer, Berlin, Heidelberg, pp.629–658, 2016.

[15] Y. Chen, V. Vaikuntanathan, and H. Wee, "GGH15 beyond permutation branching programs: Proofs, attacks, and candidates," in *Advances in Cryptology – CRYPTO 2018, Lecture Notes in Computer Science (vol.10992)*, Springer, Berlin, Heidelberg, pp.577–607, 2018.

[16] B. Barak, N. Bitansky, R. Canetti, *et al.*, "Obfuscation for evasive functions," in *Theory of Cryptography – TCC 2014, Lecture Notes in Computer Science (vol.8349)*, Springer, Berlin, Heidelberg , pp.26–51, 2014.

[17] H. Wee, "On obfuscating point functions," in *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC'05)*, Baltimore, MD, USA, pp.523–532, 2005.

[18] C. Brzuska and A. Mittelbach, "Indistinguishability obfuscation versus multi-bit point obfuscation with auxiliary input," in *Advances in Cryptology – ASIACRYPT 2014, Lecture Notes in Computer Science (vol.8874)*, Springer, Berlin, Heidelberg, pp.142–161, 2014.

[19] S. Goldwasser and Y. T. Kalai, "On the impossibility of obfuscation with auxiliary input," in *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, Pittsburgh, PA, USA, pp.553–562, 2005.

[20] R. Canetti and R.R. Dakdouk, "Obfuscating point functions with multibit output," in *Advances in Cryptology – EUROCRYPT 2008, Lecture Notes in Computer Science, (vol.4965)*, Springer, Berlin, Heidelberg, pp.489–508, 2008

[21] M. Bellare and I. Stepanovs, "Point-function obfuscation: A framework and generic constructions," in *Theory of Cryptography – TCC 2016, Lecture Notes in Computer Science (vol.9563)*, Springer, Berlin, Heidelberg, pp.565–594, 2016.

[22] R. Canetti, G. N. Rothblum, and M. Varia, "Obfuscation of

hyperplane membership," in *Theory of Cryptography – TCC 2010, Lecture Notes in Computer Science (vol.5978)*, Springer, Berlin, Heidelberg, pp.72–89, 2010.

[23] R. Goyal, V. Koppula, and B. Waters, " Lockable obfuscation," in *Proceedings of the 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS'17)*, Berkeley, CA, USA, pp.612–621, 2017.

[24] D. Wichs and G. Zirdelis, " Obfuscating compute-and-compare programs under LWE," in *Proceedings of the 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS'18)*, Berkeley, CA, USA, pp.600–611, 2017.

[25] Z. Brakerski and G.N. Rothblum, " Obfuscating conjunctions," *Advances in Cryptology – CRYPTO 2013, Lecture Notes in Computer Science (vol.8043)*, Springer, Berlin, Heidelberg, pp.416–434, 2013.

[26] Z. Brakerski, V. Vaikuntanathan, H. Wee, *et al*, "Obfuscating conjunctions under entropic ring LWE," in *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science (ITCS 2016)*, Cambridge, MA, USA, pp.147–156, 2016.

[27] A. Bishop, L. Kowalczyk, T. Malkin, *et al*, "A simple obfuscation scheme for pattern-matching with wildcards," in *Advances in Cryptology–CRYPTO 2018, Lecture Notes in Computer Science (vol.10993)*, Springer, Berlin, Heidelberg, vol.10993, pp.731–752, 2018.

[28] J. Bartusek, T. Lepoint, F. Ma, *et al*, " New techniques for obfuscating conjunctions," in *Advances in Cryptology – EUROCRYPT 2019, Lecture Notes in Computer Science (vol.11478)*, Springer, Berlin, Heidelberg, pp.636–666, 2019.

[29] S. D. Galbraith and L. Zobernig, " Obfuscated fuzzy hamming distance and conjunctions from subset product problems," in *Theory of Cryptography – TCC 2019, Lecture Notes in Computer Science (vol.11891)*, Springer, Berlin, Heidelberg, pp.81–110, 2019.

[30] R. Niebuhr, "Attacking and defending code-based cryptosystems: towards secure efficient cryptographic applications based on error-correcting codes," *Ph.D Thesis*, Technische Universität Darmstadt, Germany, 2012.

[31] E. R. Berlekamp, R. J. Maceliece, and H. Van Tilborg, "On the inherent intractability of certain coding problems (Corresp.)," *IEEE Transactions on Information Theory* , vol.24, no.3, pp.384–386, 1978.

[32] A. Barg, " Complexity issues in coding theory," *Electronic Colloquium on Computaional Complexity (ECCC)*, vol.4, no.46, pp.1–115, 1997.

[33] C. Cooper, " On the distribution of rank of a random matrix over a finite field," *Random Structures and Algorithms*, vol.17, no.3-4, pp.197–212, 2000.

[34] E. Prange, " The use of information sets in decoding cyclic codes," *IRE Transactions on Information Theory* , vol.8, no.5, pp.5–9, 1962.

[35] C. Peters, " Information-set decoding for linear codes over $\mathbf{F}_q$," in *Post-Quantum Cryptography – PQCrypto 2010, Lecture Notes in Computer Science (vol.6061)*, Springer, Berlin, Heidelberg, pp.81–94, 2010.

[36] R. Niebuhr, P. L. Cayrel, S. Bulygin, *et al*, " On lower bounds for information set decoding over $\mathbf{F}_q$," in *Proceeding of the 2nd International Conference on Symbolic Computation and Cryptography (SCC 2010)*, Egham, UK, pp.143–157, 2010.

[37] A. May and I. Ozerov, " On computing nearest neighbors with applications to decoding of binary linear codes," in *Advances in Cryptology – EUROCRYPT 2015, Lecture Notes in Computer Science (vol.9056)*, Springer, Berlin, Heidelberg, pp.203–228, 2015.

[38] S. Hirose, "May-ozerov algorithm for nearest-neighbor problem over $\mathbf{F}_q$ and its application to information set decoding," in *Innovative Security Solutions for Information Technology and Communications – SECITC 2016, Lecture Notes in Computer Science (vol.10006)*, Springer, Cham., pp.115–126

[39] C. T. Gueye, J. B. Klamti, and S. Hirose, "Generalization of BJMM-ISD using may-ozerov nearest neighbor algorithm over an arbitrary finite field $\mathbf{F}_q$," in *Codes, Cryptology and Information Security – C2SI 2017, Lecture Notes in Computer Science(10194)*, Springer, Cham., pp.96–109, 2017.

[40] S. Arora and R. Ge, " New algorithms for learning in presence of errors," in *Automata, Languages and Programming – ICALP 2011, Lecture Notes in Computer Science (vol.6755)*, Springer, Berlin, Heidelberg, pp.403–415, 2011.

ZHANG Zheng was born in 1994. She received the Ph.D. degree in computer science and engineering from Sun Yat-sen University, Guangzhou, China. Her research interests include secure obfuscation, garbled circuits, and functional encryption. (Email: zhangzh65@mail2.sysu.edu.cn)

ZHANG Zhuoran was born in 1995. She received the Ph.D. degree in computer science and engineering from Sun Yat-sen University. Her research interests include code-based cryptography. (Email: zhangzhr26@mail2.sysu.edu.cn)

ZHANG Fangguo (corresponding author) was born in 1972. He received the Ph.D. degree from the School of Communication Engineering, Xidian University, Xi'an, China, in 2001. He is currently a Professor at the School of Computer Science and Engineering of Sun Yat-sen University, China. He is the Co-director of Guangdong Key Laboratory of Information Security Technology. His research mainly focuses on cryptography and its applications. Specific interests are elliptic curve cryptography, secure obfuscation, blockchain, anonymity and privacy, etc. (Email: isszhfg@mail.sysu.edu.cn)