Constructing Secure Sketches from Groups Generically

Axel Durbet[1], Koray Karabina[2], and Kevin Thiry-Atighehchi[3]

[1] University of Calgary, Canada
axel.durbet@ucalagary.ca
[2] National Research Council of Canada, University of Waterloo, Canada
koray.karabina@nrc-cnrc.gc.ca
[3] Clermont Auvergne University, France
kevin.atighehchi@uca.fr

Abstract. Secure sketches are designed to facilitate the recovery of originally enroled data from inputs that may vary slightly over time. This capability is important in applications where data consistency cannot be guaranteed due to natural variations, such as in biometric systems and hardware security. Traditionally, secure sketches are constructed using error-correcting codes to handle these variations effectively. Additionally, principles of information theory ensure the security of these sketches by managing the trade-off between data recoverability and confidentiality. In this paper, we show how to construct a new family of secure sketches generically from groups. The notion of groups with unique factorization property is first introduced, which is of independent interest and serves as a building block for our secure sketch construction. Next, an in-depth study of the underlying mathematical structures is provided, and some computational and decisional hardness assumptions are defined. As a result, it is argued that our secure sketches are efficient; can handle a linear fraction of errors with respect to the $\mathtt{L}_1$ distance; and that they are reusable and irreversible. To our knowledge, such generic group-based secure sketch construction is the first of its kind, and it offers a viable alternative to the currently known secure sketches.

Keywords: Reusable secure sketch, Irreversible secure sketch, Group-based secure sketch, Finite fields, Lattices

1 Introduction

A typical online biometric authentication protocol runs between a client and a server in two phases: *enrolment* and *verification*. During enrolment, biometric samples are captured from the user, from which a biometric template is derived using a feature extraction algorithm. The server stores the biometric template (or some information derived from the template, *e.g.*, a cryptographic key) together with the user's identifier (ID). In the verification phase, the user regenerates their

biometric template and uses it in the protocol to prove the authentic ownership of their biometric data (enroled under their ID) against the server.

Online biometric applications, such as authentication and identification, require processing, transmitting, and storing information derived from users' biometric data, also known as the biometric template. Biometric templates are the main reference data in recognizing individuals uniquely in applications, and they are part of personally identifiable information. As a result, protecting individuals' biometric information and their privacy is crucial in biometric systems and applications. Some regulations such as the General Data Protection Regulation (GDPR) in the EU and the California Consumer Privacy Act (CCPA) for California residents of the US have been put in place for data protection and privacy. Due to the noisy nature and the personally identifying characteristics of biometrics, and that the biometric characteristics of an individual are not easy to renew, designing secure biometric-based authentication protocols is considered to be more challenging than designing token or password-based protocols. Research and standardization efforts [27,2,1] have identified several requirements for securing biometric information and templates, including *renewability*, *irreversibility*, *unlinkability*, *indistinguisahbility*, and *reusability*. Informally, *renewability* is the ability to create (randomized) biometric templates from the same biometric data. *Irreversibility* implies that it is infeasible to recover the plain biometric data from its protected template. *Unlinkability*, *indistinguishability*, and *reusability* are closely related, and they require that an adversary, who observes a user's protected biometric templates enroled at different servers, cannot yield a significant advantage towards degrading that user's security or privacy, such as cross-matching the individual's records or recovering their biometric data by reversing their biometric templates.

Secure sketch schemes are one of the main cryptographic primitives that protect biometric templates with formal security guarantees. Informally, given a biometric input $\mathtt{x}$, a sketch s is derived through a randomized process. Here, the sketch s should be irreversible, but it should allow recovering $\mathtt{x}$ in the presence of another biometric input $\mathtt{y} \approx \mathtt{x}$. In this paper, we show how to *generically* construct a new family of secure sketches from a certain family of groups. We call these secure sketches as *Subset Product Sketch* (SPS) because the construction relies on multiplying group elements from a particular subset. In a nutshell, our main contribution can be summarized as follows.

Contributions. A sketch scheme SPS can be instantiated from a triple $(\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$, where $\mathbb{G}$ is a (multiplicative) group with unique t-factorization property with respect to $\mathbb{B} \subseteq \mathbb{G}$, and $\mathcal{B}_n$ is an ordered sequence of pairwise distinct elements from $\mathbb{B}$ (see Definition 1). Furthermore, SPS satisfies the following properties.

1. Given the sketch of a vector $\mathtt{x} = [x_i]_{i=1}^n \in \mathbb{Z}^n$, and another vector $\mathtt{y} \in \mathbb{Z}^n$ with $\mathtt{L}_1(\mathtt{x}, \mathtt{y}) \leq t$, SPS can recover $\mathtt{x} \in \mathbb{Z}^n$ (Theorem 3).
2. SPS provides an abstraction of the previously known schemes `VBB` and `NTT`, and realizes them as secure sketches by adding a randomized parameter generation (Section 6 and Theorem 3). More importantly, SPS provides a unified

understanding of the security of these primitives and extends the scope of their analysis via a formal discussion on their reusability. SPS is a reusable (Theorem 5) and irreversible (Theorem 2, Theorem 6, and Theorem 3) sketch scheme under certain plausible assumptions (Figure 1).

3. SPS can *securely* handle a linear fraction of errors in the sense that if $0 \leq \mathtt{x}_i < b$, for $b \geq 2$, then one can set $t = \alpha(b-1)n$ for some positive constant α, and under that choice of parameters the complexity of the guessing attack remains exponential (Theorem 9). To this end, we also provide tight bounds on the volume of balls for the $\mathtt{L}_1$ distance (Appendix A, Theorem 11, and Theorem 12), which are of independent interest.

We should emphasize that our SPS can create sketches from integer vectors and can tolerate a linear fraction of errors with respect to the $\mathtt{L}_1$ distance, which offers a more powerful functionality than some previously known major and practical constructions whose input space is binary vectors with Hamming distance [5,35,33,8] and extends on the set of other metric spaces such as set difference and edit distance [14,3].

Outline. The rest of our paper is organized as follows. We discuss related work in Section 2. In Section 3, we introduce the notion of groups with unique factorization properties and show how to construct sketch schemes from certain families of groups generically. In Section 4, we present a theoretical security analysis for our construction, prove that it is reusable and irreversible under the hardness of certain decisional and computational problems, and provide relations between these problems. Section 5 complements our theoretical analysis by studying a list of attack strategies and providing concrete security estimates. In Section 6, we observe two concrete and practical instantiations for SPS using the virtual black box (`VBB`) [16] and noise tolerant template (`NTT`) [23] primitives, and shed light on their performance and security. The paper concludes with Section 7, which presents our final remarks.

2 Related Work

Traditionally, security has been built into the definition of sketch schemes and their security has been defined via information-theoretic notions [14]. More specifically, the information revealed via publishing the sketch value of a secret input is required to be bounded for the *(average) min-entropy* notion [29]. A relaxation has been made in [15] by switching from min-entropy to *Hill-entropy* [29], which mainly says that the distribution of secret inputs given their sketch values have entropy at least k if that distribution is computationally indistinguishable from a distribution (conditioned on their sketch values) with entropy at least k with respect to the min-entropy. As explained in detail in Section 3, we avoid associating entropy-based security notions to sketch schemes and base our security on the hardness of decisional and computational problems.

Two closely related constructions are the virtual black box (`VBB`) [16] and the noise tolerant template (`NTT`) [23] schemes. They consist of deterministic

functions and do not yield a sketch scheme as defined. Also, their security has not been previously analyzed with respect to reusability. As we discuss in Section 6, they can be used to realize concrete instantiation of SPS, and that our generic construction provides a unified understanding of these primitives and their security as secure sketches.

Early examples of secure sketches mostly rely on error-correcting codes, where the noise tolerance is measured with respect to Hamming distance or set difference metric, and their error tolerance is bounded by the error-correcting capacity of the underlying code. Fuzzy commitment [22] and fuzzy vault [21] schemes are two well-known constructions, and for a more extensive treatment of sketches and extractors based on error-correcting codes, we refer to [14].

In secure sketches and their extension to fuzzy extractor schemes, adversaries can exploit (distinct but correlated) sketches of the same client over different servers and gain significant information about their secret input. This is also known as the reusability attack [7,31,5]. Apon *et al.* [5] show that reusable fuzzy extractors can be constructed based on learning with errors problem (LWE). However, [5] can tolerate a sublinear fraction of errors (as opposed to linear). Furthermore, [5] requires that some universal public domain parameters be used across all service providers which may not be practical for implementing the scheme in real-life applications. Another reusable fuzzy extractor is constructed in [8], where the idea is to sample a sufficiently large number of sufficiently small subsets from noisy data so that samples from a relatively close data pair contain at least one identical pair that can be verified using *digital lockers*. Due to the communication and memory cost, this scheme and its variants are not yet considered to be practical [11,26]. Another disadvantage of [8] is its low error tolerance rate $k/(n \log n)$, where k is the length of the subsequences. Other reusable fuzzy extractors have been proposed based on LWE and discrete logarithm problems [33,35,34]. As discussed in this paper, our secure sketch construction can handle a linear fraction of errors with respect to the Hamming and $\mathtt{L}_1$ distances, and satisfy reusability and irreversibility, where best-known attacks seem to have exponential complexity in the length of input vectors.

3 Sketch Schemes from Groups

In this section, we show how to construct sketch schemes from certain families of groups generically, whose security will be discussed in Section 4. First, a specific family of groups $\mathbb{G}$ with *(unique) t-factorization property* with respect to a (factor) basis $\mathbb{B}$ in Section 3.1 is defined. Section 3.2 shows that $(\mathbb{G}, \mathbb{B})$ (generically) yields a sketch scheme, which we call *subset product sketch* (SPS).

3.1 Groups with Unique t-Factorization

Let $\mathbb{G}$ be a finite multiplicative group and let $\mathbb{B}$ be a finite subset of $\mathbb{G}$, where all pairwise distinct elements $u_i, u_j \in \mathbb{B}$ are ordered using the lexicographic ordering

on the binary representation of group elements. More precisely, we have $u_i < u_j$ for $i < j$. We also define

$$\mathbb{G}_{\mathbb{B},t} = \{\prod_{i=1}^{|\mathbb{B}|} u_i^{\delta_i} : \ u_i \in \mathbb{B}, \ \delta_i \in \mathbb{Z}, \ \sum_{i=1}^{|\mathbb{B}|} |\delta_i| \leq t\}. \tag{1}$$

Definition 1 ((unique) t-factorization property) *$\mathbb{G}$ has a (unique) t-factorization property with respect to a (factor) basis $\mathbb{B}$, if for all $g \in \mathbb{G}_{\mathbb{B},t}$, there is a (unique) ordered integer sequence $[\delta_i]_{i=1}^{|\mathbb{B}|}$ such that*

$$g = \prod_{i=1}^{|\mathbb{B}|} u_i^{\delta_i} \ \text{ and } \ \sum_{i=1}^{|\mathbb{B}|} |\delta_i| \leq t. \tag{2}$$

For $\mathbb{G}$ with a (unique) t-factorization property, we associate an algorithm $Factor$ that takes as input $g \in \mathbb{G}_{\mathbb{B},t}$ and outputs $[\delta_i]_{i=1}^{|\mathbb{B}|}$ satisfying (2).

Remark 1 *Even though we do not require $Factor$ to be a polynomial time algorithm in this section, its efficiency becomes important when we define our subset product sketch scheme* SPS *(see Definition 5 and Theorem 2). Section 6 provides two instantiations of* SPS *based on previous work [16,23], where the complexity of $Factor$ is polynomial in* $\log|\mathbb{G}|$ *(also see Appendix B).*

Example 1 *$\mathbb{G} = \mathbb{Z}_{31}^*$ has a unique 2-factorization property with respect to the (factor) basis $\mathbb{B} = \{2, 3\}$ because*

$$\mathbb{G}_{\mathbb{B},2} = \{1, 2, 3, 4, 6, 7, 8, 9, 11, 16, 17, 21, 26\} \tag{3}$$

has 13 elements and that there are exactly 13 distinct ordered integer sequences $[\delta_i]_{i=1}^2$ with $\sum_{i=1}^2 |\delta_i| \leq 2$. However, $\mathbb{G} = \mathbb{Z}_{31}^*$ does not have a unique 2-factorization property with respect to the (factor) basis $\mathbb{B} = \{2, 3, 11\}$ because there are two distinct ordered integer sequences $[\delta_i]_{i=1}^3$ such that $\sum_{i=1}^3 |\delta_i| \leq 2$, and that yield the same element. Namely, for $[1, 0, 0]$ and $[0, 1, 1]$, we have $2 = 2^1 3^0 11^0 = 2^0 3^1 11^1$ in $\mathbb{G}$.

Example 1 shows that groups with unique t-factorization property exist. Next, we show in Theorem 1 and Corollary 1 that for sufficiently large prime order groups with t-factorization property, the uniqueness property follows under the Gaussian heuristic [17].

Heuristic 1 (Gaussian Heuristic) *The length $\lambda_1(L)$ of the shortest vector in an n-dimensional random lattice L satisfy*

$$\lambda_1(L) \approx \sqrt{\frac{n}{2\pi e}} (\det(L))^{1/n}. \tag{4}$$

In particular, assume that there is a positive number C_n, depending on n, such that

$$\lambda_1(L) \geq C_n \sqrt{\frac{n}{2\pi e}} (\det(L))^{1/n}. \tag{5}$$

Remark 2 *Even though it seems to be a difficult problem to precisely estimate C_n in Heuristic 1 for all random lattices, Yuanmi and Nguyen state in Section 4.3 in [10], that the ratio between the final norm and the Gaussian heuristic prediction is mostly within 0.95 and 1.05. Therefore, one may replace C_n in Heuristic 1 by 0.95. It is worth noting other research papers providing estimates for C_n [9,4], which can be used to replace C_n in Heuristic 1.*

Theorem 1 *Suppose that a prime order group $\mathbb{G}$ has a t-factorization property with respect to*

$$\mathbb{B} = \{u_i \in \mathbb{G} : \ i = 1, ..., c\}, \tag{6}$$

where $|\mathbb{B}| = c$ and u_i are chosen uniformly and independently in $\mathbb{G}$. Moreover, suppose that

$$|\mathbb{G}| > \frac{1}{C_{N+1}^{N+1}}(2(\sqrt{2\pi e})(b-1))^{2t+1}, \tag{7}$$

for some integer $b \geq 2$, for all $1 \leq N \leq 2t$, and that Heuristic 1 holds. Then for all $g \in \mathbb{G}_{\mathbb{B},t}$, there is a unique ordered integer sequence $[\delta_i]_{i=1}^{|\mathbb{B}|}$ with $\sum_{i=1}^{|\mathbb{B}|} |\delta_i| \leq t$ and $|\delta_i| \leq (b-1)$ such that $g = \prod_{i=1}^{|\mathbb{B}|} u_i^{\delta_i}$.

Proof. Suppose for contradiction that there exist two distinct ordered integer sequences $[\delta_i]_{i=1}^{|\mathbb{B}|}$ and $[\tau_i]_{i=1}^{|\mathbb{B}|}$, with $\sum_{i=1}^{|\mathbb{B}|} |\delta_i| \leq t$, $|\delta_i| \leq (b-1)$, $\sum_{i=1}^{|\mathbb{B}|} |\tau_i| \leq t$, $|\tau_i| \leq (b-1)$, such that

$$\prod_{i=1}^{|\mathbb{B}|} u_i^{\delta_i} = \prod_{i=1}^{|\mathbb{B}|} u_i^{\tau_i}. \tag{8}$$

Moreover, suppose that $I = \{i_1, ..., i_N\} \subseteq \{1, ..., |\mathbb{B}|\}$ is the set of indices for which $\gamma_j = \delta_{i_j} - \tau_{i_j} \neq 0$, and that $u_{i_j} = g^{r_j}$ for some integer $r_j \in [1, |\mathbb{B}|]$, where g is a generator of $\mathbb{G} = \langle g \rangle$. Note that $1 \leq N \leq 2t$. The equation (8) is equivalent to $\sum_{j=1}^{N} r_j \gamma_j - k|\mathbb{G}| = 0$, for some integer $k \in [1, |\mathbb{G}|]$. In other words, the vector $\gamma = [\gamma_1, ..., \gamma_N, 0]$ belongs to the integer lattice L generated by the rows of the $(N+1) \times (N+1)$ matrix

$$M = \begin{bmatrix} 1 & 0 & \dots & \dots & \dots & 0 & r_1 \\ 0 & 1 & 0 & \dots & \dots & \vdots & r_2 \\ \vdots & 0 & 1 & 0 & \dots & \vdots & \vdots \\ \vdots & \vdots & 0 & \ddots & 0 & \vdots & \vdots \\ \vdots & \vdots & \vdots & 0 & 1 & 0 & r_{N-1} \\ \vdots & \vdots & \vdots & \vdots & 0 & 1 & r_N \\ 0 & \dots & \dots & \dots & \dots & 0 & |\mathbb{G}| \end{bmatrix}$$

because $\gamma = [\gamma_1, ..., \gamma_N, -k] \times M$. The length of $\gamma \in L$ can be estimated as

$$\|\gamma\| = \sqrt{\sum_{j=1}^{N} \gamma_j^2} \leq 2\sqrt{N}(b-1), \tag{9}$$

because $|\gamma_j| = |\delta_{i_j} - \tau_{i_j}| \leq 2(b-1)$ for all $j = 1, ..., N$. By assumption, u_i are uniformly and independently drawn from $\mathbb{G}$. Then, r_i are uniformly and independently drawn from $\{1, ..., |\mathbb{G}| - 1\}$ where $|\mathbb{G}|$ is prime. Hence, the lattice L can be assumed random [24] and the Gaussian heuristic 1 implies that the length of the shortest vector in L is

$$\lambda_1(L) \geq C_{N+1}\sqrt{\frac{N+1}{2\pi e}}|\mathbb{G}|^{1/(N+1)}. \tag{10}$$

Finally, using the inequalities (9), (7), the approximation (10), and our previous observation $N \leq 2t$, we derive

$$\|v\| \leq 2\sqrt{N}(b-1) \leq 2\sqrt{N+1}(b-1) \tag{11}$$

$$= \sqrt{\frac{N+1}{2\pi e}} \left(\left(2\sqrt{2\pi e}(b-1) \right)^{N+1} \right)^{1/(N+1)} \tag{12}$$

$$\leq \sqrt{\frac{N+1}{2\pi e}} \left(\left(2\sqrt{2\pi e}(b-1) \right)^{2t+1} \right)^{1/(N+1)} \tag{13}$$

$$< \sqrt{\frac{N+1}{2\pi e}} \left(|\mathbb{G}| \cdot C_{N+1}^{N+1} \right)^{1/(N+1)} \leq \lambda_1(L). \tag{14}$$

This is a contradiction because the norm of a non-zero lattice vector in L cannot be smaller than $\lambda_1(L)$. ■

Corollary 1 *Suppose that a prime order group $\mathbb{G}$ has a t-factorization property with respect to*

$$\mathbb{B} = \{u_i \in \mathbb{G} : \ i = 1, ..., c\}, \tag{15}$$

where $|\mathbb{B}| = c$ and u_i are chosen uniformly and independently in $\mathbb{G}$. Moreover, suppose that

$$|\mathbb{G}| > \frac{1}{C_{N+1}^{N+1}} (2(\sqrt{2\pi e})t)^{2t+1}, \tag{16}$$

for all $1 \leq N \leq 2t$, and that Heuristic 1 holds. Then $\mathbb{G}$ has a unique t-factorization property with respect to $\mathbb{B}$.

Proof. The proof follows by replacing b in Theorem 1 by $(t+1)$. ■

Remark 3 *If one uses the Chen and Nguyen's estimate $C_n = 0.95$ (see Remark 2), then the inequality (16) in Corollary 1 can be rewritten as*

$$|\mathbb{G}| > \frac{1}{0.95^{N+1}} (2(\sqrt{2\pi e})t)^{2t+1}.$$

3.2 Subset Product Sketch (SPS)

The main result of this section is Theorem 3, which shows that sketch schemes with respect to the $\mathtt{L}_1$ distance can be constructed using groups with unique factorization.

We start by defining a *sketch scheme*. Even though our definition is closely related to the previous definitions of (secure) sketches, there are two significant differences to point out. First, our Definition 2 avoids associating entropy-based security notions to sketch schemes and allows us to be more flexible in constructing sketch schemes and to discuss their security based on decisional and computational problems, rather independent of the entropy of the input space conditioned on the sketch values. Second, our sketch function outputs a pair of values, where the first value explicitly enforces randomization, whereas in a traditional sketch scheme, the sketch function outputs a single (sketch) value and the randomization is built into the process. For more details on related work, please see Section 2.

Definition 2 (Sketch Scheme) *Let λ be a security parameter and t a positive real number. A sketch scheme with threshold t is a tuple of randomized procedures* $\mathrm{SS} = (\mathrm{ParamGen}, \mathrm{Sketch}, \mathrm{Rec})$ *such that*

$$\mathbb{M},\ \mathcal{R} \leftarrow \mathrm{ParamGen}(\lambda), \tag{17}$$

$$\mathrm{Sketch}:\ \mathbb{M} \to \mathcal{R} \times S \tag{18}$$

$$\mathtt{x} \mapsto (R, s),\ \textit{where } R \leftarrow\!\$\ \mathcal{R}, \tag{19}$$

$$\mathrm{Rec}:\ \mathcal{R} \times S \times \mathbb{M} \to \mathbb{M} \tag{20}$$

$$(R, s, \mathtt{y}) \mapsto \mathtt{x} \tag{21}$$

and that for all $\mathtt{x}, \mathtt{y} \in \mathbb{M}$ with $d(\mathtt{x}, \mathtt{y}) \leq t$, $\mathrm{Rec}(\mathrm{Sketch}(\mathtt{x}), y) = \mathtt{x}$, except with probability negligible in λ. Here, $d:\ \mathbb{M} \times \mathbb{M} \to \mathbb{R}$ is a distance function, and $\mathcal{R}$ and S specify the output space of the randomized process Sketch.[4]

Definition 3 *We define $\mathcal{G}_{n,t} = \{(\mathbb{G}, \mathbb{B}, \mathcal{B}_n)\}$ as a family of triples, where $\mathbb{G}$ is a multiplicative group with unique t-factorization property with respect to $\mathbb{B} = [u_i]_{i=1}^{|\mathbb{B}|}$, and $\mathcal{B}_n = [g_i]_{i=1}^{n}$ is an ordered sequence of pairwise distinct elements $g_i \in \mathbb{B}$.*

Remark 4 *Note that for a fixed $\mathbb{G}$, there can be multiple choices for $\mathbb{B}$ in $\mathcal{G}_{n,t} = \{(\mathbb{G}, \mathbb{B}, \mathcal{B}_n)\}$; and for fixed $\mathbb{G}$ and $\mathbb{B}$, there can be multiple choices for $\mathcal{B}_n$.*

Definition 4 (Sketch**)** *Let n and t be positive integers and $A \subseteq \mathbb{Z}$ a finite subset of $\mathbb{Z}$. We define a sketch algorithm as follows:*

$$\mathrm{Sketch}_{n,t} : A^n \to \mathcal{G}_{n,t} \times \mathbb{G} \tag{22}$$

$$\mathtt{x} = (\mathtt{x}_1, \ldots, \mathtt{x}_n) \mapsto \left(R = (\mathbb{G}, \mathbb{B},\ \mathcal{B}_n = [g_1, ..., g_n]),\ s = \prod_{i=1}^{n} g_i^{\mathtt{x}_i} \right), \tag{23}$$

[4] Intuitively, $\mathcal{R}$ specifies the family of domain parameters and S specifies the domain for sketch values. See Definition 4 for a concrete instantiation.

where $(\mathbb{G}, \mathbb{B}, \mathcal{B}_n = [g_i]_{i=1}^n) \leftarrow\!\!\$\ \mathcal{G}_{n,t}$.

Next, Theorem 2 shows that $\text{Sketch}_{n,t}$ can be associated with a recovery algorithm Rec.

Theorem 2 *Let* $\mathtt{x} \in A^n$ *and* $\text{Sketch}_{n,t}(\mathtt{x}) = (R, s)$. *Let* $\mathtt{y} \in A^n$ *such that* $\mathtt{L}_1(\mathtt{x}, \mathtt{y}) \leq t$. *Then, there exists an algorithm* Rec *that takes as input* R, s, *and* $\mathtt{y}$; *calls Factor as a subroutine; and outputs (i.e., recovers)* $\mathtt{x}$.

Proof. Observe that

$$\Delta = \frac{s}{\prod_{i=1}^n g_i^{\mathtt{y}_i}} = \prod_{i=1}^n g_i^{\mathtt{x}_i - \mathtt{y}_i} \in \mathbb{G}_{\mathbb{B},t} \tag{24}$$

because $g_i \in \mathbb{B}$ and $\mathtt{x}_i - \mathtt{y}_i \in \mathbb{Z}$ with $\sum_{i=1}^n |\mathtt{x}_i - \mathtt{y}_i| \leq t$. Therefore, Rec can compute Δ and, using a subroutine call to *Factor* with input Δ, it can obtain an ordered sequence of integers $[\delta_k]_{k=1}^{|\mathbb{B}|}$ such that $\Delta = \prod_{k=1}^{|\mathbb{B}|} u_k^{\delta_k}$, where $\mathbb{B} = \{u_k \in \mathbb{G} :\ k = 1, ..., |\mathbb{B}|\}$ and $\sum_{k=1}^{|\mathbb{B}|} |\delta_k| \leq t$. Notice that, for each g_i, there exists a unique $k_i \in \{1, ..., |\mathbb{B}|\}$ such that $g_i = u_{k_i}$ and $\mathtt{x}_i - \mathtt{y}_i = \delta_{k_i}$ because $\mathbb{G}$ has unique t-factorization property. Moreover, the (u_{k_i}, δ_{k_i}) pair can be efficiently identified from the factorization of Δ and the knowledge of $\mathcal{B}_n$, and hence Rec can recover $\mathtt{x}$ by setting $\mathtt{x}_i = \mathtt{y}_i + \delta_{k_i}$ for $i = 1, ..., n$. ■

Finally, we define our subset product sketch scheme in Definition 5 and show in Theorem 3 that SPS is a sketch scheme.

Definition 5 (Subset Product Sketch (SPS)**)** *A subset product sketch* (SPS) *is a triple of randomized procedures* (ParamGen, $\text{Sketch}_{n,t}$, Rec), *where* ParamGen *and* $\text{Sketch}_{n,t}$ *are defined as in Definition 4 with* $\mathbb{M} = A^n$, $\mathcal{R} = \mathcal{G}_{n,t}$, $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$, $S = \mathbb{G}$; *and* Rec *is defined as in Theorem 2.*

Theorem 3 $\text{SPS} = (\text{ParamGen}, \text{Sketch}_{n,t}, \text{Rec})$ *satisfies Definition 2 with* $\mathbb{M} = A^n$, $\mathcal{R} = \mathcal{G}_{n,t}$, $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$, $S = \mathbb{G}$, $d = \mathtt{L}_1$, *and a threshold* t. *That is,* SPS *is a sketch scheme with threshold* t *with respect to the* $\mathtt{L}_1$ *distance.*

Proof. The proof follows from Definition 2, Definition 4, and Theorem 2. ■

4 On the Security of SPS

In this section, we show that SPS (see Definition 5) satisfies the reusability and irreversibility security properties under certain plausible assumptions.

4.1 Security Definitions and Games

Let SS = (ParamGen, Sketch, Rec) be a sketch scheme with threshold t as in Definition 2, $\mathcal{A}$ a probabilistic polynomial time algorithm with access to SS, and $\mathcal{D}$ a distribution over $\mathbb{M}$. We say that a problem parameterized by λ is hard if the success probability of all probabilistic polynomial time algorithms to solve that problem is negligible in λ. We first adapt reusability [34] and irreversibility [32] notions for our purposes. The concept of reusability was initially defined for fuzzy extractors [7] but to the best of our knowledge, we are the first to adapt this property to secure sketches.

The *reusability* property describes a scenario in which an adversary has access to multiple sketches of an individual, each subject to adaptively chosen perturbations. The adversary is challenged to determine whether a new sketch belongs to the same individual.

Definition 6 (The reusability experiment $\mathrm{Exp}^{\mathrm{REU}}_{\mathrm{SS},\mathcal{D},\mathcal{A}}(\lambda)$) *Let K be a positive integer polynomial in λ. The reusability experiment $\mathrm{Exp}^{\mathrm{REU}}_{\mathrm{SS},\mathcal{D},\mathcal{A}}(\lambda)$ is defined as follows.*

1. $\mathbb{M}, \mathcal{R} \leftarrow \mathrm{ParamGen}(\lambda)$
2. $b \leftarrow\$ \{0,1\}$; $\mathtt{x} \leftarrow\$ \mathcal{D}$; $(R_0, s_0) \leftarrow \mathrm{Sketch}(\mathtt{x})$
3. $\psi_0 \leftarrow \perp$; $\mathtt{s}_0 \leftarrow \perp$
4. *$\mathcal{A}$ makes K adaptive queries for $i = 1, .., K$. More precisely, at step i, $\mathcal{A}$ uses $(\mathbb{M}, \mathcal{R}; R_0, ..., R_{i-1}; s_0, ..., s_{i-1}; \psi_0, ..., \psi_{i-1})$ as input; generates ψ_i; and receives (R_i, s_i), which is computed on input $\psi_i + \mathtt{x}$ by the challenger using* Sketch.
   (a) $\mathbb{M} \ni \psi_i \leftarrow \mathcal{A}(\mathbb{M}, \mathcal{R}; R_0, ..., R_{i-1}; s_0, ..., s_{i-1}; \psi_0, ..., \psi_{i-1})$
   (b) $(R_i, s_i) \leftarrow \mathrm{Sketch}(\psi_i + \mathtt{x})$
5. *If $b = 1$, then $\mathtt{y} \leftarrow\$ \mathcal{D}$ with $d(\mathtt{x}, \mathtt{y}) \leq t$*
6. *If $b = 0$, then $\mathtt{y} \leftarrow\$ \mathcal{D}$ with $d(\mathtt{x}, \mathtt{y}) > t$*
7. $(R', s') \leftarrow \mathrm{Sketch}(\mathtt{y})$
8. $b' \leftarrow \mathcal{A}(\mathbb{M}, \mathcal{R}; R_0, ..., R_K; s_0, ..., s_K; \psi_0, ..., \psi_K; R', s')$
9. *If $b = b'$, then return 1; otherwise return 0*

Definition 7 (Reusable Sketch) *A sketch scheme* SS = (ParamGen, Sketch, Rec) *is said to be reusable with respect to the distribution $\mathcal{D}$ if*

$$\mathcal{A}dv_{\mathcal{A},\mathrm{REU}}(\lambda) = \left| \mathbb{P}(\mathrm{Exp}^{\mathrm{REU}}_{\mathrm{SS},\mathcal{D},\mathcal{A}}(\lambda) = 1) - \frac{1}{2} \right| \tag{25}$$

is negligible in λ for all $\mathcal{A}$.

The *irreversibility* property models the scenario where an attacker is challenged to recover the secret input from which the sketch is derived. An adversary can simply guess by sampling a vector at random and under this naive strategy, the probability of success can be measured relative to the size of the closed ball of radius t in the n-dimensional space A^n, centered at the input vector. A successful adversary should capture better strategies, hence motivating the following definition.

Definition 8 (The Irreversibility Experiment $\mathbf{Exp}^{\mathbf{IRR}}_{\mathrm{SS},\mathcal{D},\mathcal{A}}(\lambda)$) *The irreversibility experiment* $\mathrm{Exp}^{\mathrm{IRR}}_{\mathrm{SS},\mathcal{D},\mathcal{A}}(\lambda)$ *is defined as follows.*

1. $\mathbb{M}, \mathcal{R} \leftarrow \mathrm{ParamGen}(\lambda)$
2. $\mathtt{x} \leftarrow \mathcal{D}$; $(R, s) \leftarrow \mathrm{Sketch}(\mathtt{x})$
3. $\mathtt{y} \leftarrow \mathcal{A}(\mathbb{M}, \mathcal{R}; R, s)$
4. *If* $d(\mathtt{x}, \mathtt{y}) \leq t$, *then return* 1; *otherwise, return* 0

Definition 9 (Irreversible Sketch) *A sketch scheme* $\mathrm{SS} = (\mathrm{ParamGen}, \mathrm{Sketch}, \mathrm{Rec})$ *is said to be irreversible with respect to the distribution* $\mathcal{D}$ *if*

$$\mathcal{A}dv_{\mathcal{A},\mathrm{IRR}}(\lambda) = \left|\mathbb{P}(\mathrm{Exp}^{\mathrm{IRR}}_{\mathrm{SS},\mathcal{D},\mathcal{A}}(\lambda) = 1) - V_t\right|, \tag{26}$$

is negligible for all $\mathcal{A}$. *Here,*

$$V_t = \max_{\mathtt{x} \in \mathbb{M}} \frac{|\{\mathtt{y} \in \mathbb{M} : \ d(x, y) \leq t\}|}{|\mathbb{M}|} \tag{27}$$

estimates the success probability of the naive $\mathcal{A}$ *returning* $\mathtt{y} \leftarrow\!\!\$\ \mathbb{M}$ *at step (3) in* $\mathrm{Exp}^{\mathrm{IRR}}_{\mathrm{SS},\mathcal{D},\mathcal{A}}$.

Theorem 4 (Reusable implies irreversible) *Let* $\mathrm{SS} = (\mathrm{ParamGen}, \mathrm{Sketch}, \mathrm{Rec})$ *be a polynomial time sketch scheme. If* SS *is reusable with respect to* $\mathcal{D}$, *then* SS *is irreversible with respect to* $\mathcal{D}$.

Proof. Suppose that SS is reversible. Then there exists an adversary $\mathcal{A}$ such that $\mathcal{A}dv_{\mathcal{A},\mathrm{IRR}}(\lambda)$ is non-negligible. In the following, we show that SS is not reusable by constructing an adversary $\mathcal{A}'$ such that $\mathcal{A}dv_{\mathcal{A}',\mathrm{REU}}(\lambda)$ is non-negligible. $\mathcal{A}'$ uses $\mathcal{A}$ as a subroutine with the following strategy:

1. $\mathcal{A}'$ skips the adaptive queries and receives $(\mathbb{M}, \mathcal{R}; R, s; R', s')$ as in $\mathrm{Exp}^{\mathrm{REU}}_{\mathrm{SS},\mathcal{D},\mathcal{A}'}(\lambda)$
2. $\mathcal{A}'$ runs $\mathcal{A}$, and gets $\mathtt{y} \leftarrow \mathcal{A}(\mathbb{M}, \mathcal{R}; R, s)$, $\mathtt{y}' \leftarrow \mathcal{A}(\mathbb{M}, \mathcal{R}; R', s')$
3. $\mathcal{A}'$ runs Rec, and obtains $\mathtt{x} \leftarrow \mathrm{Rec}(R, s, \mathtt{y})$, $\mathtt{x}' \leftarrow \mathrm{Rec}(R, s', \mathtt{y}')$
4. $\mathcal{A}'$ outputs $b' = 1$ if $d(\mathtt{x}, \mathtt{x}') \leq t$; and outputs $b' = 0$, otherwise
5. $\mathcal{A}'$ outputs $b' \leftarrow\!\!\$\ \{0, 1\}$ if $\mathcal{A}$ or Rec fails

We conclude that the advantage of the adversary for reusability $\mathcal{A}dv_{\mathcal{A}',\mathrm{REU}}(\lambda)$ is non-negligible because $\mathcal{A}dv_{\mathcal{A},\mathrm{IRR}}(\lambda)$ is non-negligible and that, for all $\mathtt{x}, \mathtt{y} \in \mathbb{M}$ with $d(\mathtt{x}, \mathtt{y}) \leq t$,$\mathrm{Rec}(\mathrm{Sketch}(\mathtt{x}), \mathtt{y}) = \mathtt{x}$ by definition, except with negligible probability. ■

4.2 Reusability and Irreversibility

In the following, we show that SPS is reusable and irreversible under the hardness assumption of the *decisional subset product problem* (DSPP) and *computational subset product problem* (CSPP), respectively. We should note that DSPP and CSPP generalize the decisional distributional modular subset product and distributional modular subset product problems as defined in [16], which consider only binary vectors and Hamming distance.

Problem 1 (Decisional Subset Product Problem (DSPP)). Let $A \subseteq \mathbb{Z}$ and $\mathcal{B}_n = [g_i]_{i=1}^n$, $g_i \in \mathbb{G}$. Let $\mathcal{D}$ be a distribution over A^n. Define the distribution $\mathcal{D}_0 = (\mathcal{B}_n, X)$ where $X = \prod_{i=1}^n g_i^{\mathtt{x}_i} \in \mathbb{G}$ with $\mathtt{x} = (\mathtt{x}_1, ..., \mathtt{x}_n) \leftarrow\!\!\$\ \mathcal{D}$. Define the distribution $\mathcal{D}_1 = (\mathcal{B}_n, X')$ where $X' \leftarrow\!\!\$\ \mathbb{G}$. The decisional subset product problem (DSPP) in $\mathbb{G}$ with respect to $\mathcal{B}_n$ and $\mathcal{D}$ is to distinguish $\mathcal{D}_0$ from $\mathcal{D}_1$. When $\mathcal{D}$ is the uniform distribution over A^n, we call this problem $\mathcal{U}$-DSPP in $\mathbb{G}$ with respect to $\mathcal{B}_n$.

Theorem 5 (DSPP implies reusable) *Let* SPS = (ParamGen, $\text{Sketch}_{n,t}$, Rec) *be a sketch scheme with* $\mathbb{M} = A^n$ *and* $\mathcal{R} = \mathcal{G}_{n,t}$. *If DSPP in* $\mathbb{G}$ *with respect to* $\mathcal{B}_n$ *and* $\mathcal{D}$ *is hard for all* $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n) \in \mathcal{R}$, *then* SPS *is reusable with respect to* $\mathcal{D}$.

Proof. Consider the reusability experiment $\mathtt{Exp}^{\text{REU}}_{\text{SPS},\mathcal{A}}(\lambda)$ and let S_0 denote the event that $\mathtt{Exp}^{\text{REU}}_{\text{SPS},\mathcal{A}}(\lambda)$ outputs 1. Moreover, assume that in step (4b), we have $(R_i, s_i) \leftarrow \text{Sketch}_{n,t}(\psi_i + \mathtt{x})$, where $s_i = \prod_{j=1}^n g_j^{\mathtt{x}_j + \psi_{i,j}} = \left(\prod_{j=1}^n g_j^{\mathtt{x}_j}\right)\left(\prod_{j=1}^n g_j^{\psi_{i,j}}\right)$, which is indistinguishable from a random element in $\mathbb{G}$, because $\prod_{j=1}^n g_j^{\mathtt{x}_j}$ is indistinguishable from a random element in $\mathbb{G}$ if the DSPP is hard. Therefore, a hybrid $\mathtt{Exp}^{\text{REU-1}}_{\text{SPS},\mathcal{A}}(\lambda)$ can be defined by replacing s_1 by a random element of $\mathbb{G}$ in step (4b) in $\mathtt{Exp}^{\text{REU}}_{\text{SPS},\mathcal{A}}(\lambda)$. Similarly, $\mathtt{Exp}^{\text{REU-}i}_{\text{SPS},\mathcal{A}}(\lambda)$ can be defined by replacing s_i by a random element of $\mathbb{G}$ in step (4b) in $\mathtt{Exp}^{\text{REU-}(i-1)}_{\text{SPS},\mathcal{A}}(\lambda)$ and S_i denotes the event that $\mathtt{Exp}^{\text{REU-}i}_{\text{SPS},\mathcal{A}}(\lambda)$ outputs 1, for $i = 2, ..., K$. Observe that the probability of the event S_K that $\mathtt{Exp}^{\text{REU-}K}_{\text{SPS},\mathcal{A}}(\lambda)$ outputs 1 is 1/2. Using a sequence of hybrid arguments and the triangle inequality, we obtain:

$$|\mathbb{P}(S_0) - 1/2| \leq \underbrace{|\mathbb{P}(S_0) - \mathbb{P}(S_1)|}_{\mathcal{A}dv_{\mathcal{A},\text{DSPP}}(\lambda)} + \underbrace{|\mathbb{P}(S_1) - \mathbb{P}(S_2)|}_{\mathcal{A}dv_{\mathcal{A},\text{DSPP}}(\lambda)} + \ldots$$
$$\underbrace{+ |\mathbb{P}(S_{K-1}) - \underbrace{\mathbb{P}(S_K)}_{1/2}|}_{\mathcal{A}dv_{\mathcal{A},\text{DSPP}}(\lambda)} + \underbrace{|\underbrace{\mathbb{P}(S_K)}_{1/2} - 1/2|}_{0} \leq K \times \mathcal{A}dv_{\mathcal{A},\text{DSPP}}(\lambda),$$

and the result follows. ■

Corollary 2 (DSPP implies irreversible) *Let* SPS = (ParamGen, $\text{Sketch}_{n,t}$, Rec) *be a polynomial time sketch scheme with* $\mathbb{M} = A^n$, $\mathcal{R} = \mathcal{G}_{n,t}$, $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$, $S = \mathbb{G}$, $d = \mathtt{L}_1$, *and a threshold* t. *If DSPP in* $\mathbb{G}$ *with respect to* $\mathcal{B}_n$ *and* $\mathcal{D}$ *is hard for all* $(\mathbb{G}, \mathbb{B}, \mathcal{B}_n) \in \mathcal{G}_{n,t}$, *then* SPS *is irreversible with respect to* $\mathcal{D}$.

Proof. The proof follows from Theorem 5 and Theorem 4. ■

Theorem 2 assures the irreversibility of SPS if the DSPP is hard. Next, we provide an alternative argument for the irreversibility of SPS under the hardness assumption of the discrete logarithm problem (DLP). We first recall the definition of the DLP and define the *computational subset product problem* (CSPP).

Problem 2 (Discrete logarithm problem). The discrete logarithm problem (DLP) in $\mathbb{G}$ with respect to g is the following: Given g and $h = g^x \in \mathbb{G}$ for some (unknown) $x \leftarrow\$ \mathbb{Z}^*_{|\mathbb{G}|}$, compute x.

Problem 3 (Computational Subset Product Problem (CSPP)). Let $A \subseteq \mathbb{Z}$ and $\mathcal{B}_n = [g_i]_{i=1}^n$, $g_i \in \mathbb{G}$. Let $\mathcal{D}$ be a distribution over A^n. The computational subset product problem (CSPP) in $\mathbb{G}$ with respect to $\mathcal{B}_n$ and $\mathcal{D}$ is the following: Given $\mathcal{B}_n = [g_i]_{i=1}^n$ and $s = \prod_{i=1}^n g_i^{\mathtt{x}_i}$ for $\mathtt{x} = (\mathtt{x}_1, ..., \mathtt{x}_n) \leftarrow\$ \mathcal{D}$, compute $\mathtt{y} = (\mathtt{y}_1, ..., \mathtt{y}_n) \in \mathbb{Z}^n$ such that $s = \prod_{i=1}^n g_i^{\mathtt{y}_i}$. When $\mathcal{D}$ is the uniform distribution over A^n, we call this problem $\mathcal{U}$-CSPP in $\mathbb{G}$ with respect to $\mathcal{B}_n$.

Theorem 6 (CSPP implies irreversible) *Let* SPS = (ParamGen, $\text{Sketch}_{n,t}$, Rec) *be a polynomial time sketch scheme with* $\mathbb{M} = A^n$, $\mathcal{R} = \mathcal{G}_{n,t}$, $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$, $S = \mathbb{G}$, $d = \mathtt{L}_1$, *and a threshold* t. *If CSPP in* $\mathbb{G}$ *with respect to* $\mathcal{B}_n$ *and* $\mathcal{D}$ *is hard for all* $(\mathbb{G}, \mathbb{B}, \mathcal{B}_n) \in \mathcal{G}_{n,t}$, *then* SPS *is irreversible with respect to* $\mathcal{D}$.

Proof. Let $s = \prod_{i=1}^n g_i^{\mathtt{x}_i}$ be a given instance of the CSPP with $\mathcal{B}_n = [g_i]_{i=1}^n$ and $\mathtt{x} = (\mathtt{x}_1, ..., \mathtt{x}_n) \leftarrow\$ \mathcal{D}$ for some distribution $\mathcal{D}$ over A^n. Suppose that SPS is not irreversible. Then there exists an adversary $\mathcal{A}$ such that $\mathcal{A}dv_{\mathcal{A},IRR}(\lambda)$ is non-negligible. In other words, $\mathcal{A}$ can output $\mathtt{y} \in A^n$ such that $d(\mathtt{x}, \mathtt{y}) \leq t$. Now, given $\mathtt{y}$ and s, the recovery algorithm Rec can output $\mathtt{x}$ in polynomial time with a non-negligible probability. Hence, CSPP can be solved in polynomial time with a non-negligible probability and that finishes the proof. ■

4.3 On the Hardness of and Relationships between DSPP and CSPP

As discussed previously, reusability and irreversibility of SPS rely on the hardness of DSPP and CSPP, respectively. In this section, we study the hardness of DSPP, CSPP, and study some relationships between these problems and the DLP. We first show that the hardness of DLP and $\mathcal{U}$-DSPP implies the hardness of $\mathcal{U}$-DSPP (Theorem 7). This can be seen as evidence that CSPP may be harder than DSPP. We also show that the hardness of DLP implies the hardness of CSPP if the underlying Sketch in SPS is surjective (Theorem 8). Here, our motivation to introduce surjectivity for Sketch is to relate the subset product problems and the security of SPS to other well-known problems in cryptography. Based on Theorem 7 and Theorem 8, it is natural to ask if there is a strong relationship between the hardness of DSPP and surjectivity of Sketch. Figure 1 provides a summary of our results.

Theorem 7 ($\mathcal{U}$-DSPP and DLP implies $\mathcal{U}$-CSPP) *Let* SPS = (ParamGen, $\text{Sketch}_{n,t}$, Rec) *be a sketch scheme with* $\mathbb{M} = A^n$, $\mathcal{R} = \mathcal{G}_{n,t}$, $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$, $S = \mathbb{G}$, $d = \mathtt{L}_1$, *and a threshold* t. *Suppose that* $\mathbb{G} = \langle g \rangle$ *is a cyclic group generated by* g *and that* $\mathcal{U}$*-DSPP with respect to* $\mathcal{B}_n$ *is hard. If there is an algorithm that solves* $\mathcal{U}$*-CSPP in* $\mathbb{G}$ *with respect to* $\mathcal{B}_n$ *in time* $T_{\mathcal{U}\text{-}CSPP}$ *for all* $(\mathbb{G}, \mathbb{B}, \mathcal{B}_n) \in \mathcal{G}_{n,t}$, *then DLP in* $\mathbb{G}$ *with respect to* g *can be solved in time* $\mathcal{O}(nT_{\mathcal{U}\text{-}CSPP})$.

Proof. Let $h \in \mathbb{G}$ be given as an instance of the DLP, and let $\mathcal{A}_{\mathcal{U}\text{-}CSPP}$ be an algorithm that solves $\mathcal{U}$-CSPP in time $T_{\mathcal{U}\text{-}CSPP}$. We describe an algorithm $\mathcal{A}$ that computes $a \in \mathbb{Z}$ such that $h = g^a$. First, $\mathcal{A}$ computes $\mathtt{s}_k = g^{a_k}$ for $a_k \leftarrow\!\!\$\ \mathbb{Z}_{|\mathbb{G}|}$, and calls $\mathcal{A}_{\mathcal{U}\text{-}CSPP}$ with input $\mathtt{s}_k$ and $\mathcal{B}_n = [g_i]_{i=1}^n$. Since $\mathcal{U}$-DSPP is hard,

$$\mathtt{s}_k = \prod_{i=1}^{n} g_i^{\mathtt{x}_{k,i}},\ \mathtt{x}_{k,i} \in \mathbb{Z}. \tag{28}$$

for some $(\mathtt{x}_{k,1}, ..., \mathtt{x}_{k,n}) \leftarrow\!\!\$\ A^n$, with non-negligible probability. Hence, $\mathcal{A}$ will receive $\mathtt{x}_k = (\mathtt{x}_{k,1}, ..., \mathtt{x}_{k,n})$ as output of $\mathcal{A}_{\mathcal{U}\text{-}CSPP}$. As a result, $\mathcal{A}$ obtains a modular linear relation

$$a_k \equiv \sum_{i=1}^{n} \mathtt{x}_{k,i} d_i \pmod{|\mathbb{G}|}, \tag{29}$$

where $g_i = g^{d_i}$ for some integers d_i and $i = 1, ..., n$. $\mathcal{A}$ repeats this process until it obtains n linearly independent relations, where the total number of repetitions is expected to be polynomial in n. After collecting n linearly independent relations, $\mathcal{A}$ can recover d_i for all $i = 1, ..., n$ by solving a linear system of equations in time polynomial in n. Finally, $\mathcal{A}$ computes $s = h^b = g^{ab}$ for some random integer b relatively prime with $|\mathbb{G}|$; calls $\mathcal{A}_{\mathcal{U}\text{-}CSPP}$ with input s and $\mathcal{B}_n$, until receiving $\mathtt{x} = (\mathtt{x}_1, ..., \mathtt{x}_n)$, $\mathtt{x}_i \in \mathbb{Z}$, such that

$$s = \prod_{i=1}^{n} g_i^{\mathtt{x}_i}, \tag{30}$$

and recovers the discrete logarithm a of h with respect to g via the modular equation

$$a \equiv b^{-1} \left(\sum_{i=1}^{n} \mathtt{x}_i d_i \right) \pmod{|\mathbb{G}|}. \tag{31}$$

■

Definition 10 (Surjective Sketch) *We say that a* Sketch : $\mathbb{M} \to \mathcal{R} \times S$ *is surjective if for any* $R \in \mathcal{R}$ *and* $s \in S$*, there exists* $\mathtt{x} \in \mathbb{M}$ *such that* Sketch$(\mathtt{x}) = (R, s)$.

Example 2 *Let* $\mathbb{G} = \mathbb{Z}_{13}^*$, $\mathbb{B} = \{2, 3, 5\}$, *and* $t = 1$. *Notice that* $\mathbb{G}$ *has a unique* t*-factorization property with respect to* $\mathbb{B}$. *Let* $n = 2$, $A = \{0, 1, 2, 3\}$, *and*

$$\mathcal{G}_{n,t} = \{(\mathbb{G}, \mathbb{B}, \mathcal{B}_n = [2, 3]),\ (\mathbb{G}, \mathbb{B}, \mathcal{B}_n = [2, 5]),\ (\mathbb{G}, \mathbb{B}, \mathcal{B}_n = [3, 5])\}.$$

One can easily verify that Sketch : $A^n \to \mathcal{G}_{n,t} \times \mathbb{G}$ *is surjective.*

Theorem 8 *(Surjective* Sketch *and DLP implies CSPP) Let* SPS $=$ $(\text{ParamGen}, \text{Sketch}_{n,t}, \text{Rec})$ *be a sketch scheme with* $\mathbb{M} = A^n$*,* $\mathcal{R} = \mathcal{G}_{n,t}$*,* $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$*,* $S = \mathbb{G}$*,* $d = \mathtt{L}_1$*, and a threshold* t*. Suppose that* $\text{Sketch}_{n,t}$ *is surjective, and* $\mathbb{G} = \langle g \rangle$ *is a cyclic group generated by* g*. If there is an algorithm that solves CSPP in* $\mathbb{G}$ *with respect to* $\mathcal{B}_n$ *and* $\mathcal{D}$ *in time* T_{CSPP} *for all* $(\mathbb{G}, \mathbb{B}, \mathcal{B}_n) \in \mathcal{G}_{n,t}$*, then DLP in* $\mathbb{G}$ *with respect to* g *can be solved in time* $\mathcal{O}(nT_{CSPP})$*.*

Proof. Let $h \in \mathbb{G}$ be given as an instance of the DLP, and let $\mathcal{A}_{CSPP}$ be an algorithm that solves CSPP in time T_{CSPP}. We describe an algorithm $\mathcal{A}$ that computes $a \in \mathbb{Z}$ such that $h = g^a$. First, $\mathcal{A}$ computes $\mathtt{s}_k = g^{a_k}$ for randomly chosen integers $a_k \in \mathbb{Z}_{|\mathbb{G}|}$ and calls $\mathcal{A}_{CSPP}$ with input $\mathtt{s}_k$ and $\mathcal{B}_n = [g_i]_{i=1}^n$. Since $\text{Sketch}_{n,t}$ is surjective, $\mathcal{A}$ will receive $\mathtt{x}_k = (\mathtt{x}_{k,1}, ..., \mathtt{x}_{k,n})$ as output of $\mathcal{A}_{CSPP}$, where $\mathtt{s}_k = \prod_{i=1}^n g_i^{\mathtt{x}_{k,i}}$, $\mathtt{x}_{k,i} \in \mathbb{Z}$. The rest of the proof follows similarly the proof of Theorem 7. ■

Corollary 3 *(Surjective* Sketch *and DLP implies irreversible) Let* SPS $=$ $(\text{ParamGen}, \text{Sketch}_{n,t}, \text{Rec})$ *be a sketch scheme with* $\mathbb{M} = A^n$*,* $\mathcal{R} = \mathcal{G}_{n,t}$*,* $R = (\mathbb{G}, \mathbb{B}, \mathcal{B}_n)$*,* $S = \mathbb{G}$*,* $d = \mathtt{L}_1$*, and a threshold* t*. Suppose that* $\mathbb{G} = \langle g \rangle$ *is cyclic and* $\text{Sketch}_{n,t}$ *is surjective for all* $(\mathbb{G}, \mathbb{B}, \mathcal{B}_n) \in \mathcal{G}_{n,t}$*. If DLP in* $\mathbb{G}$ *with respect to* g *is hard, then* SPS *is irreversible.*

Proof. The proof follows from Theorem 6 and Theorem 8. ■

5 On the Concrete Security of SPS

As discussed in Section 4 and summarized in Figure 1, DSPP and CSPP are the main hardness problems to claim reusability and irreversibility of SPS. In addition, reusability implies irreversibility, and that irreversibility follows mainly from the hardness of DLP. As much as these reductionist arguments provide

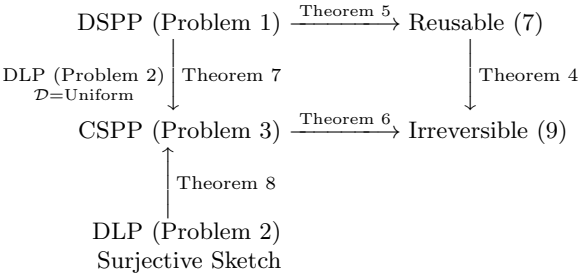

Fig. 1: Relations between the hardness of problems DSPP, CSPP, DLP, and the security properties reusability and irreversibility of SPS.

some security assurance for SPS, one should carefully study all of the underlying assumptions and try to estimate the concrete security of SPS. More precisely, DSPP and CSPP assumptions depend on the choice of the basis $\mathcal{B}_n$ and the distribution $\mathcal{D}$ over the input space. The sketch function is assumed to be surjective when reducing DLP to the irreversibility of SPS in Theorem 3. As a worst-case scenario, the hardness of DSPP, CSPP, and DLP may fail and the sketch function may not be surjective due to the choice of $\mathcal{B}_n$, $\mathcal{D}$, and other parameters such as n, t, and $\mathbb{G}$. Even though some of these failures may not imply an immediate threat for the security of SPS, our reductionist arguments would be inconclusive. Therefore, in this section, we follow a common practice in cryptography and try to estimate the security of SPS based on the best-known attack strategies.

We first investigate attacks on the irreversibility of SPS with threshold t with respect to the $\mathtt{L}_1$ distance. Suppose that an adversary $\mathcal{A}$ knows the parameters of SPS, namely $\mathbb{M}$, $\mathbb{G}$, $\mathbb{B}$, t, and $\mathcal{B}_n = [g_i]_{i=1}^n$. For concreteness, we furthermore assume $\mathbb{M} = A_b^n$, where $A_b = \{0, 1, ..., b-1\}$ for some integer $b \geq 2$, and that $\mathcal{D}$ is some distribution over A_b^n. Now, suppose that $\mathcal{A}$ captures $X = \prod_{i=1}^n g_i^{x_i} \in \mathbb{G} = \langle g \rangle$ for some unknown $\mathtt{x} = (x_1, ..., x_n) \leftarrow\!\!\$\ \mathcal{D}$, and aims to output $\mathtt{y} \in A_b^n$ such that $\mathtt{L}_1(\mathtt{x}, \mathtt{y}) \leq t$. $\mathcal{A}$ may follow the strategies as described below.

Exploit $\mathcal{D}$. In practice, we may not have control over the choice of $\mathcal{D}$. For example, $\mathtt{x}$ may be the encoding of a biometric input and could induce a low entropy on the input space for several reasons such as a high correlation on the components of $\mathtt{x}$. Therefore, we assume that $\mathcal{A}$ can fully exploit $\mathcal{D}$ and succeed in her attack with complexity $C_{\mathcal{D}} \approx 2^{\mu_{\mathcal{D}}}$ for some $\mu_{\mathcal{D}} > 0$. Note that $\mu_{\mathcal{D}} = (\log_2 b)n$ would correspond to the uniform distribution $\mathcal{D}$ over A_b^n. In practice, $\mu_{\mathcal{D}}$ is expected to be lower than $(\log_2 b)n$ and estimating $\mu_{\mathcal{D}}$ is an active area of research. In the context of biometrics, Daugman's study demonstrates that 2048-bit iriscodes exhibit entropies of 249 bits [12].

Guess $\mathtt{y}$. In this strategy, $\mathcal{A}$ chooses $\mathtt{y}$ uniformly at random from A_b^n and hopes that $\mathtt{L}_1(\mathtt{x}, \mathtt{y}) \leq t$. The success probability of this attack can be estimated as

$$\frac{|\mathtt{Ball}_{A_b^n, \mathtt{L}_1}(\mathtt{x}, t) = \{\mathtt{y} \in A_b^n :\ \mathtt{L}_1(\mathtt{x}, \mathtt{y}) \leq t\}|}{|A_b^n|}, \tag{32}$$

and so the complexity of the attack can be estimated as

$$C_{\text{Guess}} \approx \frac{b^n}{|\mathtt{Ball}_{A_b^n, \mathtt{L}_1}(\mathtt{x}, t)|} \tag{33}$$

There are two cases to consider: $b = 2$ and $b \geq 3$. Estimating $|\mathtt{Ball}_{A_b^n, \mathtt{L}_1}(\mathtt{x}, t)|$ in the binary case for $b = 2$, when $\mathtt{L}_1$ is the same as Hamming distance, is a well-studied problem in the literature, and we have

$$|\mathtt{Ball}_{A_b^n, \mathtt{L}_1}(\mathtt{x}, t)| = \sum_{k=0}^{t} \binom{n}{k} \tag{34}$$

Estimating $|\mathtt{Ball}_{A_b^n,\mathrm{L}_1}(\mathbf{x},t)|$ for $b \geq 3$ seems to be a harder problem mainly because the size of the balls are not independent of the choice of their centers. We are not aware of any previous work on this topic and we study this problem in Appendix A, which could be of independent interest. In particular, in Theorem 11, we prove that

$$\left|\mathtt{Ball}_{A_b^n,\mathrm{L}_1}(\mathbf{x},t)\right| \leq 1 + \sum_{k=1}^{t}\sum_{m=1}^{k} 2^m \binom{n}{m} \omega_{m,b-1}(k-m), \tag{35}$$

where $\omega_{m,b-1}(k-m)$ is the number of ordered partitions of the integer $(k-m)$ into m parts of size in $[0, b-2]$, which can be explicitly computed using Lemma 1. As a result, we can estimate

$$C_{\mathrm{Guess}} \gtrapprox 2^{\mu_G n}, \tag{36}$$

where $\mu_G = (\log_2 b)(1-c_t)$ for some $c_t \geq 0$ such that

$$c_t \approx \begin{cases} \log_2\left(\sum_{k=0}^{t}\binom{n}{k}\right)/n, & \text{for } b = 2. \\ \log_b\left(1 + \sum_{k=1}^{t}\sum_{m=1}^{k} 2^m\binom{n}{m}\omega_{m,b-1}(k-m)\right)/n, & \text{for } b \geq 3. \end{cases} \tag{37}$$

In Table 1, we present some concrete estimates for the cases $n = 128, 256$, $b = 2, 4, 8$, and $t = (b-1)n/8$, which corresponds to a capability of correcting a linear fraction of errors.

	$b=2$			$b=4$			$b=8$			$b \in \{2,4,8\}$
n	t	μ_G	C_{Guess}	t	μ_G	C_{Guess}	t	μ_G	C_{Guess}	C_{S}
128	16	0.48	2^{61}	48	0.58	2^{74}	112	0.66	2^{84}	2^{25}
256	32	0.47	2^{120}	96	0.57	2^{145}	224	0.64	2^{165}	2^{51}
640	80	0.46	2^{296}	240	0.55	2^{356}	560	0.63	2^{405}	2^{128}
1024	128	0.46	2^{471}	384	0.55	2^{568}	896	0.63	2^{645}	2^{204}

Table 1: Concrete security estimates for the cases $n = 128, 256$, $b = 2, 4, 8$, and $t = (b-1)n/8$. In this table, C_{Guess} estimates the complexity of the guessing attack as $2^{\mu_G n}$ while C_{S} estimates the complexity of the attack based on solving DLP, Knapsack, and SVP as $2^{0.2n}$.

More generally, we prove in Theorem 9 that C_{Guess} is exponential while a linear fraction of errors can be recovered.

Theorem 9 *Let n and $b \geq 2$ be positive integers. Let $t = \alpha(b-1)n$, where $\alpha \in (0, 1/2)$ if $b = 2$, and $\alpha \in (0, (\log_2 b - 1)/(2(b-1)))$ if $b \geq 3$. Then, $C_{Guess} \gtrapprox 2^{\mu_G n}$ for some $\mu_G > 0$. In other words, there exist parameters for* SPS *that allow recovering a linear fraction of errors (namely, up to $n/2$ errors when $b = 2$, and up to $(\log_2 b - 1)n/2$ errors when $b \geq 3$) while the complexity of the guessing attack is exponential.*

Proof. First assume $b = 2$, $t = \alpha n$, and $\alpha \in (0, 1/2)$. We can observe, using Theorem 10, that

$$C_{\text{Guess}} \gtrapprox 2^{(1-H_2(\alpha))n} \tag{38}$$

and finish the proof by setting $\mu_G = (1-H_2(\alpha))$ because $H_2(\alpha) < 1$ for $\alpha < 1/2$. Now, assume $b \geq 3$, $t = \alpha(b-1)n$, and $\alpha \in (0, (\log_2 b - 1)/(2(b-1))$. We can observe, using Theorem 12, that

$$C_{\text{Guess}} \gtrapprox 2^{(\log_2 b-(1+(2t/n)-(1/n)))n} \tag{39}$$

and finish the proof by setting $\mu_G = (\log_2 b - (1 + (2t/n) - (1/n)))$ because one can show after some algebra that $\alpha < (\log_2 b - 1)/(2(b-1))$ implies $\mu_G > 0$. ■

Solve DLP/Knapsack/SVP. In this strategy, $\mathcal{A}$ mounts a more sophisticated attack and first solves discrete logarithms of X and g_i for $i = 1, ..., n$, namely $r, d_i \in \mathbb{Z}$ such that $X = g^r$ and $g_i = g^{d_i}$. This yields a modular equation

$$\sum_{i=1}^{n} x_i d_i \equiv r \pmod{|\mathbb{G}|}.$$

$\mathcal{A}$ can try to solve for x_i from this equation via solving the (modular) knapsack problem (KP) or via solving the shortest vector problem (SVP) (see the proof of Theorem 1). The complexity of solving DLP is subexponential in $\log_2 |\mathbb{G}|$ and can be estimated based on the best-known attacks with respect to the characteristic of the finite field (small/medium/large characteristic) [13]. The complexity of solving SVP and KP can be estimated as $2^{0.2n}$ [20] and $2^{0.241n}$ [6], respectively. Therefore, we estimate the complexity of this attack strategy as

$$C_S \gtrapprox 2^{0.2n}, \tag{40}$$

and present some estimates for a certain set of parameters in Table 1.

Remark 5 *(Complexity of attacking* SPS *is exponential in n) We are not aware of better strategies to attack the irreversibility of* SPS *other than the ones we discussed above. Similarly, the best approach to compromise the reusability of* SPS *appears to be attacking irreversibility. Hence, our analysis indicates that the complexity of attacking* SPS *is* $2^{\mu \cdot n}$*, where* $\mu = \min(\mu_{\mathcal{D}}/n, \mu_G, 0.2)$*. Note that this complexity is exponential in n if the input space has sufficient entropy and t is chosen carefully as explicitly described in Theorem 9.*

6 Concrete Instantiations of SPS

In this section, we observe that SPS can be realized in practice using the virtual black box (`VBB`) [16] and noise tolerant template (`NTT`) [23] primitives.

Instantiation of $\text{Sketch}_{n,t}$. Both VBB and NTT parameter generations take as input n and t, and output a group $\mathbb{G}$ as well as a basis $\mathcal{B}_n = [g_1, ..., g_n]$. In the case of NTT, $\mathbb{G}$ is a subgroup of the multiplicative group of a finite field $\mathbb{F}_{q^2}$, and $\mathcal{B}_n$ consists of elements $(\alpha + \sigma)/(\alpha - \sigma)$, where $\alpha \in \mathbb{F}_p$, $\mathbb{F}_{q^2} = \mathbb{F}_q(\sigma)$, and $q = p^t$. Notice that the basis elements strictly lie in $\mathbb{F}_{q^2}$ but are represented by the base field $\mathbb{F}_p$. In the case of VBB, $\mathbb{G}$ is a subgroup of the multiplicative group of integers modulo a prime q, and $\mathcal{B}_n$ consists of small prime numbers. In both cases, $\mathbb{G}$ and $\mathcal{B}_n$ are used to map a binary vector $\mathtt{x} = (\mathtt{x}_1, ..., \mathtt{x}_n)$ to a value $X = \prod_{i=1}^{n} g_i^{\mathtt{x}_i} \in \mathbb{G}$. The transformation is referred to as *project* in NTT, and as *encode* in VBB. It is straightforward to generalize this transformation from binary vectors to $\mathtt{x}$ with $0 \leq \mathtt{x}_i \leq b-1$, which we use in our instantiation (see Appendix B.1).

Instantiation of Rec. Both VBB and NTT propose algorithms to reconstruct the vector $\mathtt{x}$ given X and another vector $\mathtt{y}$, where $\mathtt{x}$ and $\mathtt{y}$ are binary vectors with $\mathtt{HD}(\mathtt{x}, \mathtt{y}) \leq t$. The reconstruction can be generalized from binary vectors to $\mathtt{x}$, $\mathtt{y}$ with $0 \leq \mathtt{x}_i, \mathtt{y}_i \leq b-1$ when $\mathtt{L}_1(\mathtt{x}, \mathtt{y}) \leq t$. Reconstruction algorithms are referred to as *Decomp* in NTT, and as *Decoding* in VBB. Hence, by Theorem 1, Corollary 1, and Theorem 3, SPS can be realized using VBB and NTT under Heuristic 1. The reader is referred to Appendix B.1 and Appendix B.2 for further details.

Performance and Security Evaluations. We provide running time evaluations for our constructions based on single-thread C and C++ programs using the GMP [18] and NTL [30] libraries. The aforementioned processes are executed on a computer running Debian 11, which is equipped with an 11th-generation Intel Core i7-1185G7 processor operating at 3.00 GHz and 16 GB of RAM. Table 2 summarizes the performance tests over 100 iterations for $n = 640$, $b \in \{2, 4, 8\}$, and $t = (b-1)n/8$. Note that the parameter set provides 128-bit security level according to Table 1. Our implementation demonstrates that both realizations of the SPS (SPS_{NTT} and SPS_{VBB}) are efficient and suitable for applications in practice.

Algorithm	Space	Threshold	Sketch time (ms) Average	Rec time (ms) Average	Template size (in bits)
SPS_{NTT}	$(\mathbb{Z}_2)^{640}$	80	16.505	33.05	880
SPS_{VBB}			0.09	10.28	971
SPS_{NTT}	$(\mathbb{Z}_4)^{640}$	240	124.24	379.21	2,640
SPS_{VBB}			2.87	11.92	2,970
SPS_{NTT}	$(\mathbb{Z}_8)^{640}$	560	651.26	3,389.00	6,160
SPS_{VBB}			23.36	45.07	6,781

Table 2: Experimental results for Sketch and Rec implementations over the parameters $n = 640$, $b \in \{2, 4, 8\}$, and $t = (b-1)n/8$. Timings have been averaged over 100 iterations.

7 Concluding Remarks

We have proposed a generic way of constructing secure sketches from groups, called SPS. Our novel construction operates on integer vectors for $\mathtt{L}_1$ distance, thereby extending previously known sketch constructions on binary vectors with the Hamming distance. We observed that our SPS can be instantiated using known primitives such as `NTT` and `VBB` based on finite field groups and subgroups.

In contrast to most other secure sketches or fuzzy extractors of the literature, SPS tolerates a linear fraction of errors. As a result, we present a flexible, practical, and secure construction, as the sketch is proven to be irreversible and reusable under computational and decisional assumptions. A concrete security analysis of a list of attack strategies on SPS indicates that the complexity of such attacks is exponential in n. In addition, our running time analysis shows that even with a high-dimensional vector and a large number of errors, SPS remains fast and practical with an average time under 50 ms.

It would be interesting to investigate whether SPS can be realized using other cryptographically interesting groups. It would also be interesting to challenge the security of SPS and find new attack strategies with improved complexities.

ACKNOWLEDGMENTS

We thank the reviewers for their feedback which improved the quality of this work. The authors acknowledge the support of the French Agence Nationale de la Recherche (ANR), under grant ANR-20-CE39-0005 (project PRIVABIO).

References

1. Joint Technical Committee ISO/IEC JTC 1. ISO/IEC30136:2018(E): Information technology – Performance testing of biometric template protection scheme. Standard, International Organization for Standardization, 2018.
2. Joint Technical Committee ISO/IEC JTC 1. ISO/IEC 24745:2022 Information Security, Cybersecurity and Privacy Protection - Biometric Information Protection. Standard, International Organization for Standardization, 2022.
3. Quentin Alamélou, Paul-Edmond Berthier, Chloé Cachet, Stéphane Cauchie, Benjamin Fuller, Philippe Gaborit, and Sailesh Simhadri. Pseudoentropic isometries: A new framework for fuzzy extractor reusability. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pages 673–684, 2018.
4. Yoshinori Aono, Thomas Espitau, and Phong Q Nguyen. Random lattices: theory and practice. *Preprint. https://espitau. github. io/bin/random_lattice.pdf*, 2019.
5. Daniel Apon, Chongwon Cho, Karim Eldefrawy, and Jonathan Katz. Efficient, reusable fuzzy extractors from LWE. In *Cyber Security Cryptography and Machine Learning: First International Conference, CSCML 2017, Beer-Sheva, Israel, June 29-30, 2017, Proceedings 1*, pages 1–18. Springer, 2017.
6. Daniel J Bernstein, Stacey Jeffery, Tanja Lange, and Alexander Meurer. Quantum algorithms for the subset-sum problem. In *Post-Quantum Cryptography: 5th International Workshop, PQCrypto 2013, Limoges, France, June 4-7, 2013. Proceedings 5*, pages 16–33. Springer, 2013.

7. Xavier Boyen. Reusable cryptographic fuzzy extractors. In *Proceedings of the 11th ACM conference on Computer and Communications Security*, pages 82–91, 2004.
8. Ran Canetti, Benjamin Fuller, Omer Paneth, Leonid Reyzin, and Adam Smith. Reusable fuzzy extractors for low-entropy distributions. *Journal of Cryptology*, 34:1–33, 2021.
9. Hao Chen. A measure version of gaussian heuristic. Cryptology ePrint Archive, Paper 2016/439, 2016.
10. Yuanmi Chen and Phong Q Nguyen. Bkz 2.0: Better lattice security estimates. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 1–20. Springer, 2011.
11. Jung Hee Cheon, Jinhyuck Jeong, Dongwoo Kim, and Jongchan Lee. A Reusable Fuzzy Extractor with Practical Storage Size: Modifying Canetti et al.'s Construction. In *Information Security and Privacy*, pages 28–44, 2018.
12. John Daugman. How iris recognition works. In *The essential guide to image processing*, pages 715–739. Elsevier, 2009.
13. Gabrielle De Micheli, Pierrick Gaudry, and Cécile Pierrot. Asymptotic complexities of discrete logarithm algorithms in pairing-relevant finite fields. In *Advances in Cryptology–CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part II 40*, pages 32–61. Springer, 2020.
14. Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM journal on computing*, 38(1):97–139, 2008.
15. Benjamin Fuller, Xianrui Meng, and Leonid Reyzin. Computational fuzzy extractors. *Information and Computation*, 275:1–20, 2020.
16. Steven D. Galbraith and Lukas Zobernig. Obfuscated fuzzy hamming distance and conjunctions from subset product problems. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography*, pages 81–110, Cham, 2019. Springer International Publishing.
17. Nicolas Gama and Phong Q Nguyen. Predicting Lattice Reduction. In *Advances in Cryptology – EUROCRYPT 2008*, pages 31–51, 2008.
18. Torbjörn Granlund, Gunnar Sjödin, Hans Riesel, Richard Stallman, Brian Beuning, Doug Lea, John Amanatides, Paul Zimmermann, Ken Weber, Per Bothner, Joachim Hollman, Bennet Yee, Andreas Schwab, Robert Harley, David Seal, Torsten Ekedahl, Linus Nordberg, Kevin Ryde, Kent Boortz, Steve Root, Gerardo Ballabio, Jason Moxham, Pedro Gimeno, Niels Möller, Alberto Zanoni, Marco Bodrato, Marco Bodrato, David Harvey, Martin Boij, Marc Glisse, David S Miller, Mark Sofroniou, and Ulrich Weigand. The gnu multiple precision arithmetic library, 2024. `https://gmplib.org/`.
19. Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. Essential coding theory, 2023.
20. Alexander Helm and Alexander May. The power of few qubits and collisions–subset sum below grover's bound. In *Post-Quantum Cryptography: 11th International Conference, PQCrypto 2020, Paris, France, April 15–17, 2020, Proceedings*, pages 445–460. Springer, 2020.
21. Ari Juels and Madhu Sudan. A fuzzy vault scheme. *Designs, Codes and Cryptography*, 38(2):237–257, 2006.
22. Ari Juels and Martin Wattenberg. A fuzzy commitment scheme. In *Proceedings of the 6th ACM Conference on Computer and Communications Security*, pages 28–36. Association for Computing Machinery, 1999.

23. Koray Karabina and Onur Canpolat. A new cryptographic primitive for noise tolerant template security. *Pattern Recognition Letters*, 80:70–75, 2016.
24. Phong Q Nguyen and Damien Stehlé. LLL on the Average. In *Proceedings of the 7th International Conference on Algorithmic Number Theory*, pages 238–256, 2006.
25. Cornelia Ott, Sven Puchinger, and Martin Bossert. Bounds and genericity of sum-rank-metric codes. In *2021 XVII International Symposium" Problems of Redundancy in Information and Control Systems"(REDUNDANCY)*, pages 119–124. IEEE, 2021.
26. Somnath Panja, Nikita Tripathi, Shaoquan Jiang, and Reihaneh Safavi-Naini. Robust and reusable fuzzy extractors and their application to authentication from iris data. Cryptology ePrint Archive, Paper 2023/284, 2023. `https://eprint.iacr.org/2023/284`.
27. Christian Rathgeb and Andreas Uhl. A Survey on Biometric Cryptosystems and Cancelable Biometrics . *EURASIP Journal on Information Security*, 3:1–25, 2011.
28. Joel Ratsaby. Estimate of the number of restricted integer-partitions. *Applicable Analysis and Discrete Mathematics*, pages 222–233, 2008.
29. Leonid Reyzin. Some notions of entropy for cryptography: (invited talk). In *International Conference on Information Theoretic Security*, pages 138–142. Springer, 2011.
30. Victor Shoup. Ntl: A library for doing number theory, 2024. `https://libntl.org/`.
31. Koen Simoens, Pim Tuyls, and Bart Preneel. Privacy weaknesses in biometric sketches. In *2009 30th IEEE Symposium on Security and Privacy*, pages 188–203. IEEE, 2009.
32. Koen Simoens, Pim Tuyls, and Bart Preneel. Privacy Weaknesses in Biometric Sketches. In *2009 30th IEEE Symposium on Security and Privacy*, pages 188–203, 2009.
33. Yunhua Wen and Shengli Liu. Reusable Fuzzy Extractor from LWE. In *Information Security and Privacy*, pages 13–27, 2018.
34. Yunhua Wen and Shengli Liu. Robustly reusable fuzzy extractor from standard assumptions. In *Advances in Cryptology – ASIACRYPT 2018*, pages 459–489, 2018.
35. Yunhua Wen, Shengli Liu, and Shuai Han. Reusable fuzzy extractor from the decisional diffie–hellman assumption. In *Designs, Codes, and Cryptography*, volume 86, pages 2495–2512, 2018.

A Estimating the Size of Balls

We denote the set of integers and real numbers by $\mathbb{Z}$ and $\mathbb{R}$, respectively. For positive integers $b \geq 2$ and n, we define

$$A_b = \{0, 1, ..., b-1\}, \tag{41}$$

as the set of integers from 0 to $b-1$; and

$$A_b^n = \{\mathtt{x} = (\mathtt{x}_1, ..., \mathtt{x}_n) : \ \mathtt{x}_i \in A_b\}, \tag{42}$$

as the set of length-n vectors over A_b.

In this paper, we will be interested in two different types of distance functions

$$d : \ A_b^n \times A_b^n \to \mathbb{R}$$

on A_b^n, namely the generalized Hamming distance $\mathtt{HD}$, and the Manhattan distance $\mathtt{L}_1$, which are defined as follows:

$$d(\mathtt{x},\mathtt{y}) = \mathtt{HD}(\mathtt{x},\mathtt{y}) = \#\{i : \ \mathtt{x}_i \neq \mathtt{y}_i\} \tag{43}$$

$$d(\mathtt{x},\mathtt{y}) = \mathtt{L}_1(\mathtt{x},\mathtt{y}) = \sum_{i=1}^{n} |\mathtt{x}_i - \mathtt{y}_i|, \tag{44}$$

where $\mathtt{x} = (\mathtt{x}_1, ..., \mathtt{x}_n)$ and $\mathtt{y} = (\mathtt{y}_1, ..., \mathtt{y}_n)$ are in A_b^n.

Notice that, when $b = 2$, both $\mathtt{HD}$ and $\mathtt{L}_1$ yield the regular Hamming distance. For given $\mathtt{x} \in A_b^n$, $t \in \mathbb{R}$, and a distance function d on A_b^n, the ball of radius t about its center $\mathtt{x}$ with respect to d, denoted $\mathtt{Ball}_{A_b^n,d}(\mathtt{x},t)$, is defined as

$$\mathtt{Ball}_{A_b^n,d}(\mathtt{x},t) = \{\mathtt{y} \in A_b^n : \ d(\mathtt{x},\mathtt{y}) \leq t\}. \tag{45}$$

When $d = \mathtt{HD}$, the size of the ball $\mathtt{Ball}_{A_b^n,\mathtt{HD}}(\mathtt{x},t)$ is independent of its center $\mathtt{x}$, and which leads to the definition of the *volume of a Hamming ball of radius* t [19]:

$$\mathtt{Vol}_{A_b^n,\mathtt{HD}}(t) = \left|\mathtt{Ball}_{A_b^n,\mathtt{HD}}(\mathtt{x},t)\right| = \sum_{k=0}^{t} \binom{n}{k} (b-1)^k \tag{46}$$

Part of our security discussions rely on estimating $\left|\mathtt{Ball}_{A_b^n,d}(\mathtt{x},t)\right|$. When $d = \mathtt{HD}$, this boils down to estimating $\mathtt{Vol}_{A_b^n,\mathtt{HD}}(t)$ (see (46)). for which we refer to the definition of the b-ary entropy function $H_b(\alpha)$ (Definition 11) and to the well-known result Theorem 10 (a proof can be found in Section 3.3.1 in [19]).

Definition 11 (b-ary entropy function) *For an integer $b \geq 2$ and $0 \leq \alpha \leq 1$, the b-ary entropy function is defined as*

$$H_b(\alpha) = \alpha \log_b(b-1) - \alpha \log_b(\alpha) - (1-\alpha)\log_b(1-\alpha) \tag{47}$$

Theorem 10 (Estimating $\mathtt{Vol}_{A_b^n,\mathtt{HD}}(t) = \left|\mathtt{Ball}_{A_b^n,\mathtt{HD}}(\mathtt{x},t)\right|$) *Let $b \geq 2$ be an integer and $0 \leq \alpha \leq (b-1)/b$. Then, for all $\mathtt{x} \in A_b^n$ and sufficiently large n, we have*

$$b^{H_b(\alpha)n - o(n)} \leq \left|\mathtt{Ball}_{A_b^n,\mathtt{HD}}(\mathtt{x}, t = \alpha \cdot n)\right| \leq b^{H_b(\alpha)n}. \tag{48}$$

Note that for a fixed $b \geq 2$ and sufficiently large n, Theorem 10 yields the estimate

$$\left|\mathtt{Ball}_{A_b^n,\mathtt{HD}}(\mathtt{x}, t = \alpha \cdot n)\right| \approx b^{H_b(\alpha)n}, \tag{49}$$

where $0 \leq \alpha \leq (b-1)/b$.

We should emphasize that, for general d, $\left|\mathtt{Ball}_{A_b^n,d}(\mathtt{x},t)\right|$ depends on the center $\mathtt{x}$ of the ball, and so $\mathtt{Vol}_{A_b^n,\mathtt{HD}}(t)$ may not be generalized for other distance functions. In particular, we are not aware of analogous estimates for $\left|\mathtt{Ball}_{A_b^n,d}(\mathtt{x},t)\right|$ for a general distance function d. However, as we show in Theorem 11, we can derive explicitly computable upper and lower bounds for $\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right|$. Theorem 11 builds on Lemma 1 from [28].

Lemma 1 *[Lemma 1.1 in [28]] Let m, k, b be integers such that $1 \leq m \leq k$, $b \geq 2$. Let $\omega_{m,b}(k)$ be the number of ordered partitions of the integer k into m parts of size between 0 and $(b-1)$. Then,*

$$\omega_{m,b}(k) = \sum_{i=0,b,2b,\ldots}^{k} (-1)^{i/b} \binom{m}{i/b} \binom{k-i+m-1}{k-i}. \tag{50}$$

For a better presentation of the proof of Theorem 11, we define the following sets and prove some results.

For $\mathtt{x} \in A_b^n$, and positive integers k and m with $m \leq k$, define

$$S_{\mathtt{x},b,k} = \{\mathtt{y} \in A_b^n : \sum_{i=1}^{n} |\mathtt{y}_i - \mathtt{x}_i| = k\}, \tag{51}$$

$$U_{b,k} = \{\mathtt{z} \in \mathbb{Z}^n : 1-b \leq \mathtt{z}_i \leq b-1, \sum_{i=1}^{n} |\mathtt{z}_i| = k\}, \tag{52}$$

$$U_{b,k,m} = \{\mathtt{z} \in U_{b,k} : |\operatorname{Supp}(\mathtt{z})| = m\}, \tag{53}$$

$$U_{b,k,m}^{\geq 0} = \{\mathtt{z} \in U_{b,k,m} : \mathtt{z}_i \geq 0\}, \tag{54}$$

$$V_{b,k,m} = \{\mathtt{z} \in \mathbb{Z}^m : 1 \leq \mathtt{z}_i \leq b-1, \sum_{i=1}^{m} \mathtt{z}_i = k\}, \tag{55}$$

$$W_{b,k,m} = \{\mathtt{z} \in \mathbb{Z}^m : 0 \leq \mathtt{z}_i \leq b-2, \sum_{i=1}^{m} \mathtt{z}_i = k-m\}, \tag{56}$$

$$L_{b',k} = \{\mathtt{z} \in A_{b'}^n : \sum_{i=1}^{n} \mathtt{z}_i = k\} \tag{57}$$

Here, the *support* of a vector $\mathtt{z}$, $\operatorname{Supp}(\mathtt{z})$, is defined as the set of indices i, where the components $\mathtt{z}_i$ of $\mathtt{z}$ are non-zero. That is,

$$\operatorname{Supp}(\mathtt{z}) = \{i : \mathtt{z}_i \neq 0\}. \tag{58}$$

Lemma 2 *Let $b \geq 2$ and t be positive integers. Then*

$$\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right| - 1 = \sum_{k=1}^{t} |S_{\mathtt{x},b,k}| \leq \sum_{k=1}^{t} |U_{b,k}| = \sum_{k=1}^{t} \sum_{m=1}^{k} |U_{b,k,m}|, \tag{59}$$

for all $\mathtt{x} \in A_b^n$.

Proof. The proof easily follows by the definitions of the underlying sets, and by observing that the function

$$\begin{aligned} \phi : \; & S_{\mathtt{s},b,k} \to U_{b,k} \\ & (\mathtt{y}_1, ..., \mathtt{y}_n) \mapsto (\mathtt{z}_1, ..., \mathtt{z}_n), \end{aligned} \tag{60}$$

where $\mathtt{z}_i = \mathtt{y}_i - \mathtt{x}_i$, is well-defined and injective. ■

Lemma 3 *Let $b \geq 2$, k, m be positive integers with $m \leq k$. Then*

$$|U_{b,k,m}| = 2^m \left| U^{\geq 0}_{b,k,m} \right| \tag{61}$$

Proof. Let ϕ be the function defined as

$$\begin{aligned} &\phi : \ U_{b,k,m} \to U^{\geq 0}_{b,k,m} \\ &(\mathtt{z}_1, ..., \mathtt{z}_n) \mapsto (|\mathtt{z}_1|, ..., |\mathtt{z}_n|). \end{aligned} \tag{62}$$

The proof follows because ϕ is an onto function and each element in $U^{\geq 0}_{b,k,m}$ has exactly m elements in its support and hence has exactly 2^m preimages under ϕ. ■

Lemma 4 *Let $b \geq 2$, k, m, n be positive integers with $m \leq k$ and $m \leq n$. Then*

$$\left| U^{\geq 0}_{b,k,m} \right| \leq \binom{n}{m} |V_{b,k,m}| \tag{63}$$

Proof. Let ϕ be the function defined as

$$\begin{aligned} &\phi : \ U^{\geq 0}_{b,k,m} \to V_{b,k,m} \\ &(\mathtt{z}_1, ..., \mathtt{z}_n) \mapsto (\mathtt{z}_{i_1}, ..., \mathtt{z}_{i_m}), \end{aligned} \tag{64}$$

where $\{i_j : \ j = 1, .., m\} = \mathrm{Supp}(\mathtt{z})$. The proof follows because ϕ is an onto function and each element in $V_{b,k,m}$ has at most $\binom{n}{m}$ preimages under ϕ. ■

Lemma 5 *Let $b \geq 2$, k, m be positive integers with $m \leq k$. Then*

$$|V_{b,k,m}| = |W_{b,k,m}| = \omega_{m,b-1}(k-m) \tag{65}$$

Proof. The function

$$\begin{aligned} &\phi : \ V_{b,k,m} \to W_{b,k,m} \\ &(\mathtt{z}_1, ..., \mathtt{z}_m) \mapsto (\mathtt{z}_1 - 1, ..., \mathtt{z}_m - 1), \end{aligned} \tag{66}$$

is a bijection and so $|V_{b,k,m}| = |W_{b,k,m}|$. $|W_{b,k,m}| = \omega_{m,b-1}(k-m)$ follows because, by definition, $\omega_{m,b-1}(k-m)$ is the number of ordered partitions of the integer $(k-m)$ into m parts of size between 0 and $(b-2)$, which is precisely $|W_{b,k,m}|$. ■

Lemma 6 *Let $b \geq 2$, k, n be positive integers, and $b' = \lfloor (b-1)/2 \rfloor + 1$. Then $|S_{\mathtt{x},b,k}| \geq |L_{b',k}| = \omega_{n,b'}(k)$, for all $x \in A_b^n$.*

Proof. Consider the function

$$\begin{aligned} \phi: \ & L_{b',k} \to S_{\mathtt{x},b,k} \\ & (\mathtt{z}_1, ..., \mathtt{z}_n) \mapsto (\mathtt{y}_1, ..., \mathtt{y}_n), \end{aligned} \tag{67}$$

where $\mathtt{y}_i = \mathtt{x}_i - \mathtt{z}_i$ if $\mathtt{x}_i > b' - 1$; and $\mathtt{y}_i = \mathtt{x}_i + \mathtt{z}_i$ if $\mathtt{x}_i \leq b' - 1$. Note that this is a well-defined function because $0 \leq \mathtt{y}_i \leq b - 1$ and

$$\sum_{i=1}^{n} |\mathtt{y}_i - \mathtt{x}_i| = \sum_{i=1}^{n} |\mathtt{z}_i| = k. \tag{68}$$

Also note that ϕ is injective, and so $|S_{\mathtt{x},b,k}| \geq |L_{b',k}|$. Finally, $|L_{b',k}| = \omega_{n,b'}(k)$ follows because, by definition, $\omega_{n,b'}(k)$ is the number of ordered partitions of the integer k into n parts of size between 0 and $b' - 1$, which is precisely $|L_{b',k}|$. ■

Theorem 11 (A lower bound for $\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right|$) *Let $b \geq 2$ be an integer, $b' = \lfloor (b-1)/2 \rfloor + 1$, $\mathtt{x} \in A_b^n$, and $\omega_{m,b}(k)$ defined as in Lemma 1. We have*

$$\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right| \leq 1 + \sum_{k=1}^{t} \sum_{m=1}^{k} 2^m \binom{n}{m} \omega_{m,b-1}(k-m) \tag{69}$$

and

$$\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right| \geq 1 + \sum_{k=1}^{t} \omega_{n,b'}(k) \tag{70}$$

Proof. The upper bound follows from Lemmas 2-5. The lower bound follows from Lemma 2 and Lemma 6. ■

Theorem 12 (An upper bound for $\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right|$) *Let $\mathtt{x} \in A_b^n$ and $b \geq 3$. Then,*

$$\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right| \leq 2^{n+2t-1} \tag{71}$$

Proof. Using Theorem 11, Vandermonde's identity, and the bound on $\omega_{m,b-1}(k-m)$ given by Ott *et al.* [25], we have

$$\left|\mathtt{Ball}_{A_b^n,\mathtt{L}_1}(\mathtt{x},t)\right| \leq 1 + \sum_{k=1}^{t} \sum_{m=1}^{k} 2^m \binom{n}{m} \omega_{m,b-1}(k-m) \tag{72}$$

$$\leq \sum_{k=1}^{t} 2^k \sum_{m=1}^{k} \binom{n}{m} \binom{k-1}{m-1} \leq 2^t \sum_{k=1}^{t} \binom{n+k-1}{k} \tag{73}$$

$$\leq 2^t \sum_{k=1}^{t} \binom{n+t-1}{k} \leq 2^{n+2t-1} \tag{74}$$

and the result follows. ■

B Details on VBB and NTT

In this section, we provide details on our implementation of SPS using NTT [16] and VBB [23].

B.1 Instantiation of SPS using VBB

VBB [16] is an obfuscator for Hamming distance-based fuzzy matching where the security relies on the difficulty of the Modular Subset Product Problem (MSP), which can be realized as a special case of CSPP for $A = \{0, 1\}$ and $\mathbb{G} = \mathbb{Z}_q$; see Problem 3. In this section, we build a sketch using the building blocks of VBB. As before, λ is a security parameter, and the integers n and t are system parameters. $\mathrm{SPS}_{\mathtt{VBB}}$ can be instantiated via a tuple of probabilistic polynomial time (PPT) algorithms $(\mathrm{ParamGen}, \mathrm{Sketch}, \mathrm{Rec})$ as described in Algorithm 1, Algorithm 2 and Algorithm 4, respectively.

The ParamGen algorithm (Algorithm 1) takes as input a security parameter λ, a positive integer n, the threshold parameter t, and a positive integer b. It outputs a family $\mathcal{G}_{n,t} = \{(\mathbb{G}, \mathbb{B}, \mathcal{B}_n)\}$. In this family, $\mathbb{G} = \mathbb{Z}_q^*$ is parametrized by prime numbers q such that $\mathbb{Z}_q^*$ has λ-bit security with respect to DLP. Once $\mathbb{G} = \mathbb{Z}_q^*$ is fixed, $\mathbb{B}$ is chosen as the set of all prime numbers g_i satisfying

$$\max_I \prod_{i \in I} g_i^b < \frac{q}{2} < (1 + o(1)) \max_i (g_i)^t \text{ with } |I| = \lfloor t/b \rfloor, \quad \text{and } I \subset \{0, \ldots, n\}. \tag{75}$$

For a fixed $\mathbb{G}$ and $\mathbb{B}$, $\mathcal{B}_n = [g_i]_{i=1}^n$ can be chosen as any ordered sequence of n pairwise distinct elements g_i in $\mathbb{B}$.

Algorithm 1: ParamGen

Input: λ, n, t, b

Return $\{(\mathbb{G} = \mathbb{Z}_q^*,$
$\mathbb{B} =$ The set of all primes g_i satisfying (75),
$\mathcal{B}_n = \{[g_i]_{i=1}^n \mid g_i \in \mathbb{B} \text{ and } g_i \neq g_j \ \forall i \neq j\})$

The Sketch algorithm (Algorithm 2) takes as input a security parameter λ, a vector size n, a threshold t, an integer b, and a vector $\mathtt{x} = (\mathtt{x}_1, \ldots, \mathtt{x}_n) \in A_b^n$, where $A_b = \{0, 1, ..., b-1\}$. It outputs a pair (R, s) with R a parameter set and s the encoded version of $\mathtt{x}$ (*i.e.*, the sketch of $\mathtt{x}$).

The CFactor algorithm (Algorithm 3) takes as inputs a number x and a fixed list of primes $[g_1, \ldots, g_n]$. It outputs $\perp$ if x is composite with factors that are not in the list of primes and the factorization of x otherwise.

The Rec algorithm (Algorithm 4) receives as input R, s, and $\mathtt{y} = (\mathtt{y}_1, \ldots, \mathtt{y}_n) \in A_b^n$. It outputs $\mathtt{x}$ if $d(\mathtt{x}, \mathtt{y}) \leq t$ and fails ($\perp$) otherwise.

Algorithm 2: Sketch

Input: $\lambda, n, t, b, \mathtt{x} \in A_b^n$
Output: $R = (\mathbb{G} = \mathbb{Z}_q^*,\ \mathbb{B},\ \mathcal{B}_n = [g_i]_{i=1}^n),\ s$
$(\mathbb{G}, \mathbb{B}, \mathcal{B}_n) \leftarrow\!\!\$\ \text{ParamGen}(\lambda, n, t, b)$
$s \leftarrow \prod_{i=1}^n g_i^{\mathtt{x}_i} \bmod q$
Return $R,\ s$

Algorithm 3: $CFactor$

Input: $n, [g_1, \ldots, g_n], x$
Output: S
$S = [\,]$;
for *i=1,. . . ,n* **do**
  while $g_i | x$ **do**
    Append g_i to S;
    $x \leftarrow x/g_i$;
  end
end
if $x = 1$ **then**
  Return S;
else
  Return $\perp$;
end

Algorithm 4: Rec

Input: $R = (\mathbb{G} = \mathbb{Z}_q^*,\ \mathbb{B},\ \mathcal{B}_n = [g_i]_{i=1}^n), s, \mathtt{y}$
Output: $\mathtt{x}$ if $d(\mathtt{x}, \mathtt{y}) \leq t$ and $\perp$ otherwise.
Compute $s' = \prod_{i=1}^n g_i^{-\mathtt{y}_i} \bmod q$;
Compute $\Delta = s \cdot s' \bmod q$;
Compute the continued fraction representation of Δ/q,
with convergent C;
forall $h/k \in C$ **do**
  $F \leftarrow CFactor(n, [g_1, \ldots, g_n], k)$;
  $G \leftarrow CFactor(n, [g_1, \ldots, g_n], k \cdot \Delta \bmod q)$;
  if $F \neq \perp$ *and* $G \neq \perp$ **then**
    Let $m = \{0\}^n$ be the zero vector;
    for $i = 1, \ldots, n$ **do**
      if $g_i \in F$ **then**
        Set m_i to minus the number of repetitions of g_i;
      else if $g_i \in G$ **then**
        Set m_i to the number of repetitions of g_i;
      end
    end
    Return $\mathtt{y} + m$;
end
Return $\perp$;

B.2 Instantiation of SPS using NTT

NTT [23] is a cryptographic primitive to create noise-tolerant templates, where the security relies on the hardness of DLP and the Knapsack Problem (KP). SPS_{NTT} is a tuple of polynomial time algorithms (ParamGen, Sketch, Rec). The parameter generation algorithm ParamGen (Algorithm 5) takes as input a security parameter λ, a positive integer n, the threshold parameter t, and a positive integer b. It outputs a family $\mathcal{G}_{n,t} = \{(\mathbb{G}, \mathbb{B}, \mathcal{B}_n)\}$. In this family, $\mathbb{G} = \mathbb{F}_{q^2} = \mathbb{F}_q[\sigma]/\langle\sigma^2 - c\rangle$ is a finite field of size $q^2 = p^{2t}$ for some irreducible polynomial $(\sigma^2 - c)$ over $\mathbb{F}_q$ and that $\mathbb{F}_q$ is a finite field of size p^t parametrized by a prime number p. $\mathbb{G}$ must have λ-bit security with respect to DLP. Once $\mathbb{G} = \mathbb{F}_q$ is fixed, $\mathbb{B}$ is chosen as the set of all elements of the form $g = (\alpha + \sigma)/(\alpha - \sigma)$ for $\alpha \in \mathbb{F}_p$. For a fixed $\mathbb{G}$ and $\mathbb{B}$, $\mathcal{B}_n = [g_i]_{i=1}^n$ can be chosen as any ordered sequence of n pairwise distinct elements g_i in $\mathbb{B}$.

Algorithm 5: ParamGen

Input: λ, n, t, b

Return $\{(\mathbb{G} = \mathbb{F}_{q^2},$
$\mathbb{B} = \{\frac{\alpha+\sigma}{\alpha-\sigma} \in \mathbb{G} \mid \alpha \in \mathbb{F}_p\}$
$\mathcal{B}_n = \{[g_i]_{i=1}^n \mid g_i \in \mathbb{B} \text{ and } g_i \neq g_j \ \forall i \neq j\})$

The Sketch algorithm (Algorithm 6) takes as inputs a security parameter λ, a vector size n, a threshold t, a positive integer b and a vector $\mathtt{x} = (\mathtt{x}_1, \ldots, \mathtt{x}_n) \in A_b^n$. It outputs (R, s) with $R = (\mathbb{G} = \mathbb{F}_{q^2},\ \mathbb{B},\ \mathcal{B}_n = [g_i]_{i=1}^n)$ a parameter set and $s = \prod_{i=1}^n g_i^{\mathtt{x}_i}$.

Algorithm 6: Sketch

Input: $\lambda, n, t, \mathtt{x}, b$
Output: $(R = (\mathbb{G} = \mathbb{F}_{q^2},\ \mathbb{B},\ \mathcal{B}_n = [g_i]_{i=1}^n), s)$
$R = (\mathbb{G},\ \mathbb{B},\ \mathcal{B}_n = [g_i]_{i=1}^n) \leftarrow\!\!\$\ \text{ParamGen}(\lambda, n, t, b)$;
$s \leftarrow \prod_{i=1}^n g_i^{\mathtt{x}_i} \in \mathbb{F}_{q^2}$;
Return (R, s);

The Rec algorithm (Algorithm 7) takes as input $R = (\mathbb{G} = \mathbb{F}_{q^2}, \mathbb{B}, \mathcal{B}_n = [g_i]_{i=1}^n), s,\ b$ a positive integer and $\mathtt{y} = (\mathtt{y}_1, \ldots, \mathtt{y}_n) \in A_b^n$ a vector. It then outputs $\mathtt{x}$ if $d(\mathtt{x}, \mathtt{y}) \leq t$, and $\perp$ otherwise.

Remark 6 *In Algorithm 7, $f_0 - f_1\alpha = 0$ is a single equation over $\mathbb{F}_q$ including t $\mathbb{F}_p$-variables e_i. Using the fact that $\mathbb{F}_q$ is a t-dimensional vector space over $\mathbb{F}_p$, this equation can be rewritten as t linear equations over $\mathbb{F}_p$ with the same $\mathbb{F}_p$-variables e_i. Hence, we can use linear algebra to solve for e_i.*

Algorithm 7: Rec

Input: $R = (\mathbb{G} = \mathbb{F}_q[\sigma]/\langle \sigma^2 - c \rangle,\ \mathbb{B},\ \mathcal{B}_n = [g_i = (\alpha_i + \sigma)/(\alpha_i - \sigma)]_{i=1}^n), s, b, \mathtt{y}$
Output: $\mathtt{x}$ if $d(\mathtt{x}, \mathtt{y}) \leq t$ and $\perp$ otherwise.
Compute $s' = \prod_{i=1}^n g_i^{\mathtt{y}_i} \in \mathbb{G}$;
$T \leftarrow s/s' \in \mathbb{G}$;
Find $\alpha \in \mathbb{F}_q$ such that $T = ((\alpha + \sigma)/(\alpha - \sigma))$;
For $\mathbb{F}_p$-variables e_i, $i = 1, ..., t$, assign:
$f_0 \leftarrow \sum_{i=0}^{\lfloor t/2 \rfloor} e_{t-2i} c^j$ and $f_1 \leftarrow \sum_{i=0}^{\lfloor (t-1)/2 \rfloor} e_{t-2i-1} c^j$;
Use $f_0 - f_1 \alpha = 0$ and solve for $e_i \in \mathbb{F}_p$;
Construct $P(X) = X^t + \sum_{i=1}^t (-1)^i e_i X^{t-i}$;
Find $\mathbb{F}_p$-roots r_i of $P(X)$ and their multiplicity m_i. $i = 1, ..., k$.
$\mathtt{w} \leftarrow (0, \ldots, 0) \in \mathbb{Z}^n$;
for $i \in \{1, \ldots, n\}$ **do**
  if $\alpha_i \in \{r_1, \ldots r_k\}$ **then**
    $\mathtt{w}_i \leftarrow m_j$, where $\alpha_i = r_j$;
  else if $-\alpha_i \in \{r_1, \ldots r_k\}$ **then**
    $\mathtt{w}_i \leftarrow -m_j$, where $-\alpha_i = r_j$;
  else
    Return $\perp$;
  end
end
$\mathtt{x} \leftarrow \mathtt{w} + \mathtt{y}$;
Return $\mathtt{x}$;