Generalizing Cryptosystems Based on the Subset Sum Problem*

Aniket Kate Ian Goldberg

David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, Canada N2L 3G1
{akate,iang}@cs.uwaterloo.ca

Abstract

We identify a generic construction of cryptosystems based on the subset sum problem and characterize the required homomorphic map. Using the homomorphism from the Damgård-Jurik cryptosystem, we then eliminate the need for a discrete logarithm oracle in the key generation step of the Okamoto, Tanaka and Uchiyama scheme to provide a practical cryptosystem based on the subset sum problem. We also analyze the security of our cryptosystem and show that with proper parameter choices, it is computationally secure against lattice-based attacks. Finally, we present a practical application of this system for RFID security and privacy.

1 Introduction

Knapsack Cryptosystems

Knapsack cryptosystems, or more accurately, subset sum problem (SSP)-based cryptosystems, are the most well-studied cryptosystems based on NP-complete problems. We refer readers to a survey by Lai [17] for a detailed discussion and cryptanalysis of many of them. In 2000, Okamoto, Tanaka and Uchiyama [24] presented an SSP-based cryptosystem (the OTU scheme), which combines good features of the multiplicative Merkle-Hellman [20] and the Chor-Rivest [5] cryptosystems to overcome known attacks on SSP-based cryptosystems. A motivation behind this system is to define a quantum public key cryptosystem, as many classical public key cryptosystems (PKCs) are breakable using Shor's algorithm [30] on a quantum computer. Further, its use of low-weight subsets reduces its public-key size from $\Omega(n^2 \log n)$ to $o(n^2)$. Recent lattice-based attacks [13, 22, 25] raise serious theoretical questions about the security of the OTU scheme; however, given the very large polynomial complexity of the LLL algorithm, for moderate values of n, lattice-based attacks can be thwarted in practice while retaining a reasonable public key size (see [22, Sec. 7] for practical examples).

RFID Security and Privacy

RFID (Radio-Frequency IDentification) is a technology for automated identification of objects and people. When compared with its predecessor—barcodes—RFID tags offer significant improvements in efficiency through unique object identification rather than categorical recognition, and through fully automated reading. On the other hand, ubiquitous use of RFID tags raises a significant privacy concern of clandestine tracking and inventorying [15]. Although data and communication privacy using public key encryption are

*Revised June 2009

solved problems in the cryptographic literature with schemes such as RSA and Diffie-Hellman, all of those protocols are beyond the computational capabilities of the majority of RFID tags.

A variety of security and privacy protocols have been suggested for RFID tags (see Avoine's web resource [1] for a detailed list). However, most of them have some known weakness such as a shared symmetric key between a RFID reader and many RFID tags, or the requirement of exhaustive database search for ID management. All of these weaknesses in the existing RFID security and privacy schemes can easily be overcome using a public key encryption system. In such a system, a RFID tag encrypts the identification information using a pre-installed public key of the valid RFID readers, which decrypts the transferred message to securely obtain identification information of the tag. However, the computational requirements of encryption in cryptosystems based on the discrete logarithm and integer factorization problems are beyond the capacity of these resource-constrained devices.

Along with its significance to post-quantum cryptography, the OTU scheme is also suitable for resource-constrained RFID devices; its public-key size is reasonable for active RFID tags and encryption in this scheme only involves a very small number of integer additions. Still, practical use of the OTU scheme is not possible in the near future, as it requires a (quantum) oracle to solve arbitrary discrete logarithm problems (DLP) in the key generation step. This paper eliminates the requirement of a DLP oracle in the OTU scheme and makes the resulting scheme feasible for practical use in identification of resource-constrained devices.

Our Contributions

As our major contribution, we present the generic construction of SSP-based cryptosystems, where the finite field DLP computation is replaced by an additive homomorphism with certain properties. We then show that a specialization of the homomorphic map of the Damgård-Jurik generalization [9] of the Paillier cryptosystem [26] provides the required homomorphism and define a concrete realization of our scheme with polynomial-time key generation on classical computers. Finally, we show the computational inapplicability of lattice-based attacks for our chosen key sizes and define a computation-constrained identification protocol for RFID chips.

In the next section, we summarize the important SSP-based cryptosystems and discuss the *densities* of the corresponding SSP instances in relation to lattice-based attacks. In Section 3, we present our generic SSP-based cryptosystem and use the Damgård-Jurik homomorphism to provide a realistic SSP-based scheme in Section 4. The security of our scheme is discussed in Section 5 and Section 6 presents a practical application of our scheme for RFID security and privacy. We conclude our discussion in Section 7.

2 Preliminaries

2.1 SSP in Cryptosystems

Definition 2.1 (Subset Sum Problem)**.** *Let $\mathscr{A} = \{a_1, \ldots, a_n\}$ be a set of positive integers. Given the set $\mathscr{A}$ and an integer c, find $\mathscr{A}' \subseteq \mathscr{A}$ (if such a subset exists) such that $c = \sum_{a_i \in \mathscr{A}'} a_i$.*

In the key generation step of an SSP-based PKC, a set $\mathscr{A}$ of size n is computed using a private key and published as a public key along with some other scheme-specific parameters. A sender chooses a subset $\mathscr{A}' \subseteq \mathscr{A}$ uniquely associated to a plaintext M via a binary encoding, computes the subset sum $c = \sum_{a_i \in \mathscr{A}'} a_i$ and sends c as the ciphertext. The intended receiver, knowing the private key, converts this subset sum instance $(\mathscr{A}, c)$ to an efficiently solvable problem and recovers the plaintext M by solving it.

To be useful in cryptography, any subset sum (or ciphertext c) should not have two different subsets associated with it, as in that case, a unique decryption would not be possible. The associated specialization of the SSP is known as the *unique subset sum* problem.

Definition 2.2 (Unique Subset Sum Problem)**.** *Let $\mathscr{A} = \{a_1, \ldots, a_n\}$ be a set of positive integers such that sum of every subset is unique. That is, for any $\mathscr{A}_1, \mathscr{A}_2 \subseteq \mathscr{A}$, if $\sum_{a_i \in \mathscr{A}_1} a_i = \sum_{a_j \in \mathscr{A}_2} a_j$, then $\mathscr{A}_1 = \mathscr{A}_2$. Given the set $\mathscr{A}$ and an integer c, find $\mathscr{A}' \subseteq \mathscr{A}$ (if such a subset exists) such that $c = \sum_{a_i \in \mathscr{A}'} a_i$.*

As unique SSPs are a proper subset of (general) SSPs, given an oracle to solve the SSP, one can solve a unique subset-sum instance; thus, `Unique Subset Sum` $\leq_p$ `Subset Sum`. A reduction in other direction seems to be unlikely, as deciding if a subset sum instance is a unique subset sum instance is itself P^{NP}-complete [27]. Separation between these two problems is further supported by the difference in their densities and subsequent susceptibility of many instances of the unique SSP to lattice-based attacks; we discuss these issues next.

2.2 SSP Density and Lattice-Based Attacks

Although the subset-sum problem is NP-hard in the worst case, not all instances of the SSP are equally difficult. The applicability of known attacks on SSP-based cryptosystems is determined by a metric called the subset sum *density* [16].

Definition 2.3 (Density)**.** *The density D of a subset sum instance with set $\mathscr{A} = \{a_1, \ldots, a_n\}$ is the ratio of the size of the set $\mathscr{A}$ to the size of the largest element in it. $D = n/\log_2(\max_{\mathscr{A}})$, where $\max_{\mathscr{A}} = \max_i(a_i)$.*

The density D for a general subset sum instance can take any non-negative value; however, for a unique subset sum instance, D is bounded above by $1 + \mathrm{O}\left(\frac{\log\log\max_{\mathscr{A}}}{\log\max_{\mathscr{A}}}\right)$; we show this fact next. We find that a result of Erdős [11] (with improvements by Elkies [10]) is applicable to the density of the unique subset sum instances. [12] For $\ell > 1$, the largest number of integers $a_1, \ldots, a_n \leq 2^\ell$ having the unique sum property is limited by

$$\ell + 1 \leq n < \ell + \frac{1}{2}\log_2 \ell + \frac{1}{2}\log_2 \pi \,.$$

Using this, the density D of any set of ℓ-bit values $\{a_1, \ldots, a_n\}$ with the unique sum property is

$$\begin{aligned} D &< \frac{\ell + 0.5\log_2 \ell + 0.5\log_2 \pi}{\ell} \\ &< 1 + \frac{\log_2 \ell}{2\ell} + \frac{0.826}{\ell}. \end{aligned} \tag{1}$$

Therefore, for large values of ℓ, $D < 1 + \mathrm{O}\left(\frac{\log\log\max_{\mathscr{A}}}{\log\max_{\mathscr{A}}}\right)$. In practice, most unique subset sum instances have $D \ll 1$.

Low values of D in the unique SSP instances are a major concern, as Lagarias-Odlyzko [16] and Brickell [3] showed that it is possible to solve almost all SSPs when D is sufficiently small ($D \ll 1$). They accomplished this by reducing the SSP to the problem of finding the Euclidean norm of the shortest non-zero vector in a lattice. Coster et al. [6] and Joux-Stern [14] (the CJLOSS attack) independently improved the bound by demonstrating that it is possible to solve almost all SSPs of density $D < 0.9408$ (asymptotically) if a lattice oracle is present, which with high probability, given a lattice basis of dimension $n+1$, finds the Euclidean-norm shortest nonzero vector of the lattice in polynomial time. While no polynomial-time algorithm is known to compute the Euclidean-norm shortest nonzero vector of a lattice precisely, the polynomial-time algorithm by Lenstra, Lenstra and Lovász (LLL algorithm) [18] solves it with good approximation for $D < 1$ in practice. As for almost all unique subset sum instances, $D < 1$, the unique SSP is significantly vulnerable to lattice-based attacks and these attacks are the major concern to the security of SSP-based cryptosystems.

2.3 The Merkle-Hellman Multiplicative Knapsack-Based Cryptosystem

As the pioneering work in the area, Merkle and Hellman [20] proposed an SSP-based cryptosystem known as the *Merkle-Hellman multiplicative knapsack-based cryptosystem*. This PKC is different from the more famous Merkle-Hellman knapsack-based cryptosystem using a super-increasing sequence, which was broken by Shamir [29]. It uses the DLP computation in a prime field $\mathbb{F}_p$ for a generator g to obtain a unique subset sum instance $\mathscr{B} = (b_1, \ldots, b_n)$ from a set of n pairwise coprime integers $\mathscr{P} = (p_1, \ldots, p_n)$ such that the product $p_1 p_2 \cdots p_n < p$. Further, the DLP in the field $\mathbb{F}_p$ must be tractable, which asks for $p-1$ without any large divisors. The unique subset sum instance $(b_1, \ldots, b_n)$ forms a public key, while $(p, g, p_1, \ldots, p_n)$ forms a private key. Given an n-bit plaintext $(m_1, \ldots, m_n)$, a sender computes a ciphertext c as $c = \sum_{i=1}^{n} m_i b_i$. While decrypting, knowing the private key, a receiver computes $u \equiv g^c \pmod{p} = \prod g^{m_i b_i}$. The value u will be a product of integers chosen from $\{p_1, \ldots, p_n\}$, so the receiver can factor u by trial division of these values to recover the plaintext (see [20] for a detailed discussion).

As $b_i = \log_g p_i \in \mathbb{Z}_{p-1}$, with high probability, the size of the set of elements $\log(\max_{\mathscr{B}}) = \log(p)$, where $\max_{\mathscr{B}} = \max(b_1, \ldots, b_n)$. As density $D = \frac{n}{\log(b_{\max})}$, we obtain $\log p = n/D$. In the Merkle-Hellman cryptosystem, as $\prod_{p_i \in \mathscr{P}} p_i < p$, on average, $p_i < \sqrt[n]{p}$. Therefore, $\log p_i < \frac{1}{n} \log p$, $\log p_i < \frac{n}{nd} = \frac{1}{d}$ and for any reasonable p_i, the density $d \ll 1$. Therefore, the density of a unique subset sum instance generated by the Merkle-Hellman multiplicative knapsack-based cryptosystem is significantly less than 1 and is not secure against LLL-based lattice attacks. Due to the high $O(n^6 \log^3(\max_{\mathscr{B}}))$ complexity of the LLL algorithm, choosing a large $n \geq 500$ is an option to avoid lattice attacks, as $\max_{\mathscr{B}} > (n \log n)^n$. However, this results in an infeasibly large public key, often on the order of megabytes.

2.4 The OTU Quantum Public Key Cryptosystem

Okamoto, Tanaka and Uchiyama propose a new SSP-based scheme (the OTU scheme) [24], which combines the best features from the multiplicative Merkle-Hellman and Chor-Rivest [5] cryptosystems and overcomes the low density problem and the known underlying field problems in the respective systems. We summarize the basic OTU scheme in Appendix A and refer the reader to [24] for a detailed discussion.

In the basic OTU scheme, as in the Merkle-Hellman cryptosystem, knowledge of a finite field $\mathbb{F}_p$, a particular generator g, and an extra "shifting parameter" d allows decryption by converting a subset sum instance into a factorization problem with a known small set of factors to choose from. The key distinction is that in the Merkle-Hellman scheme, p must be chosen so that $p > p_1 p_2 \cdots p_n$, whereas in the OTU scheme, it suffices that p be greater than products of any k of the p_i, for a suitable parameter $k = \mathrm{o(n)}$. This eliminates the low density problem as the resulting b_i will be smaller, increasing the density D by a factor of n/k. The derived public key is no longer a unique subset sum instance; it is rather an instance of a variant we call the *k-unique subset sum* problem.

Definition 2.4 (k-Unique Subset Sum Problem)**.** *Let $\mathscr{A} = \{a_1, \ldots, a_n\}$ be a set of positive integers such that the sum of every subset of any fixed size $\leq k$ is unique (i.e., for $\mathscr{A}_1, \mathscr{A}_2 \subseteq \mathscr{A}$ of the same size $\leq k$, if $\sum_{a_i \in \mathscr{A}_1} a_i = \sum_{a_j \in \mathscr{A}_2} a_j$, then $\mathscr{A}_1 = \mathscr{A}_2$). Given the set $\mathscr{A}$ and an integer c, find $\mathscr{A}' \subseteq \mathscr{A}$ (if such a subset exists) such that*

$$c = \sum_{a_i \in \mathscr{A}'} a_i.$$

As only certain subset sums need to be unique, the density bound in Equation (1) does not apply to the k-unique SSP. Although, to the best of our knowledge, there is no related density bound for k-unique SSP instances available in the literature, in practice, the densities of k-unique SSP instances are much higher than those of the unique SSP instances. Further, as every unique SSP instance is also a k-unique SSP instance, the k-unique SSP is at least as hard as the unique SSP.

On the practical side, using the k-unique SSP with small value of $k = o(n)$ also provides significant savings in the public key size, which is in the range of $nk(\log n + \log(\log n))$. This makes $n \approx 1000$ also a feasible case, for which a LLL-algorithm execution is well beyond the currently assumed adversarial computational capabilities.

The Chor-Rivest cryptosystem [5] was the first to use the k-unique SSP. It used the elegant Bose-Chowla theorem [2] to obtain k-unique SSP instances, and mitigated the problem of low density in the Merkle-Hellman scheme. The OTU scheme, which combines the Merkle-Hellman and the Chor-Rivest schemes, achieves similar (medium) density by generating k-unique SSP instances. Concerned mostly with the density-based attacks, both these systems fail to formally identify and define the k-unique SSP.

2.5 Lattice-Based attacks for k-Weight Subset Sums

On the theoretical side, Omura and Tanaka [25] and Nguyen and Stern [22] improve lattice-based attacks on SSP instances for the specialized case of the low values of k used in the OTU scheme. They observe that due to the known low weights of the OTU plaintexts, the density bounds of the Lagarias-Odlyzko as well as CJLOSS attacks can be revised to much higher values. Nguyen and Stern further demonstrate that all the parameter values (n, k) suggested in the original OTU paper are actually susceptible to lattice-based attacks and introduce a new density measure called *pseudo-density* $= k\log_2 n/\log_2(\max_{\mathscr{B}})$ to be used in place of conventional density for low-weight SSPs. Notably, they also suggest the possibility of OTU parameters secure against lattice-based attacks.

To achieve pseudo-density > 1 we need $\max_{\mathscr{B}} < n^k$. In the OTU scheme, $\max_{\mathscr{P}} \gg n$ and the product of the k largest p_i has to be smaller than p. Further, $\max_{\mathscr{B}}$ is nearly equal to p, and $p > (\max_{\mathscr{P}})^k \gg n^k$. Thus pseudo-density > 1 is nearly impossible and we do not consider the OTU scheme to be theoretically secure against lattice-based attacks. However, as discussed above, with large n, practical lattice-based attacks are computationally infeasible. Without any significant reduction in the LLL algorithm complexity in last 25 years since its inception, we believe in security based on the infeasibility of an LLL computation for $n \geq 500$ and use that to define a practical SSP-based system.

In another effort, Izu et al. [13] generalized the CJLOSS attack to improve its non-asymptotic behaviour; they can attack low-weight SSPs of higher density, at a cost of $O(n^r)$ lattice basis reductions of dimension $n - r + 1$, where r is a small constant (typically $r \leq 5$). Considering the very high cost of a single LLL-algorithm call, performing $O(n^r)$ lattice basis reductions is certainly well beyond existing computational capabilities for $n \geq 500$.

3 A Generic Cryptosystem Based on the SSP

Most of the SSP-based cryptosystems use DLP computations in their key generation steps. They assume the existence of an oracle to solve arbitrary DLP (e.g. the OTU scheme [24]) or propose to choose a finite field from a restricted set where the DLP computation is easy (e.g. Merkle-Hellman [20] and Chor-Rivest [5] cryptosystems). This significantly reduces the practicality of the SSP-based cryptosystems, until quantum computers are available to solve arbitrary DLPs in polynomial time.

In this section, we eliminate need of the DLP computation in the key generation step of these cryptosystems. We show that any invertible map with certain homomorphic properties can replace the DLP computation and present a generic construction of an SSP-based cryptosystem based on it.

3.1 Homomorphism in SSP-Based Cryptosystems

SSP-based schemes use the DLP computation over a multiplicative group of a finite field to convert easy factorization problems (with a small set of possible factors) to SSP instances. In such schemes, DLP computa-

tions in the key generation step map secret integers to their exponents. In exactly the opposite manner, in the decryption step, modular exponentiations convert ciphertexts, which are sums of some of these exponents, into products of the corresponding secret integers. The homomorphic property of the DLP computation over a multiplicative group of a finite field, which makes these cryptosystems work, can be expressed as follows.

$$\mathtt{DLP}_g(x_1) + \mathtt{DLP}_g(x_2) \quad \equiv \quad \mathtt{DLP}_g(x_1 * x_2)$$

where $\mathtt{DLP}_g : \mathbb{F}_q^* \mapsto \mathbb{Z}/(q-1)\mathbb{Z}$ such that for every $x \in \mathbb{F}_q^*$ and a generator $g \in \mathbb{F}_q^*$, $y = \mathtt{DLP}_g(x)$ if $x = g^y$.

In these cryptosystems, the DLP itself does not provide the security. It is rather based on hiding the field $\mathbb{F}_q$ and the generator g, as it is infeasible to obtain an integer x (private key) from the corresponding exponent y (public key) without knowing $\mathbb{F}_q$ and g. Except for the homomorphic property in Equation 2, none of other features of the DLP are used in these cryptosystems. Therefore, we can replace the DLP oracle in these schemes with any other invertible additive-multiplicative homomorphism, where computation of the map and its inverse is infeasible without knowing some secret information. Formally, we need a map H_σ with following properties.

1. H_σ should be an invertible map such that

$$H_\sigma : \mathscr{S}_1 \to \mathscr{S}_2$$
$$H_\sigma^{-1} : \mathscr{S}_2 \to \mathscr{S}_1$$

   are homomorphisms, where $(\mathscr{S}_1, *)$ and $(\mathscr{S}_2, +)$ are two Abelian semigroups of the same size.

2. Computation of both H_σ and H_σ^{-1} requires knowledge of a secret parameter σ.

3.2 Generic SSP-based Scheme

We first define the k-unique subset sum property over structures other than the integers.

Definition 3.1 (k-unique Subset Sum property over $(\mathscr{S}, +)$)**.** *Let $(\mathscr{S}, +)$ be an Abelian semigroup, and let $\mathscr{A} = \{a_1, \ldots, a_n\}$ be a set of n elements of $\mathscr{S}$. The $\mathscr{A}$ has the k-unique Subset Sum property over $(\mathscr{S}, +)$ if whenever $\mathscr{A}_1, \mathscr{A}_2 \subseteq \mathscr{A}$ of the same size $\leq k$ and $\sum_{a_i \in \mathscr{A}_1} a_i = \sum_{a_j \in \mathscr{A}_2} a_j$, then we have that $\mathscr{A}_1 = \mathscr{A}_2$.*

We now use the above homomorphic map H_σ to define the generic SSP-based cryptosystem. Although H_σ can replace the DLP computation in any SSP-based cryptosystem, we choose the OTU scheme considering its computational security against all known attacks.

Key Generation:

1. Fix size parameters $n, k \in \mathbb{Z}^+$, with $k < n/2$.
2. Choose two Abelian semigroups $(\mathscr{S}_1, *)$, $(\mathscr{S}_2, +)$ and a mapping $H_\sigma : \mathscr{S}_1 \to \mathscr{S}_2$ (with corresponding secret σ) satisfying the properties listed in Section 3.1.
3. Select at random n elements $p_1, p_2, \ldots, p_n \in \mathscr{S}_1$ such that $P = \{p_1, p_2, \ldots, p_n\}$ has the k-unique Subset Sum property over $(\mathscr{S}_1, *)$, and solving the k-unique SSP on the set P in $(\mathscr{S}_1, *)$ is easy, given that the target subset is of size k.

   An easy way to do this is to have $\mathscr{S}_1$ be the multiplicative group of $GF(p)$, and for $p_1, p_2, \ldots, p_n$ to be such that the product of any k of them (or properly, their least positive representatives) is less than p, and are pairwise coprime. Note that in this case, since $(\mathscr{S}_1, *)$ is a multiplicative group, the SSP here is really referring to finding factors of products instead of addends of sums.
4. Use the secret σ to compute $a_i = H_\sigma(p_i) \in \mathscr{S}_2$ for $1 \leq i \leq n$.

5. Randomly choose an integer $d \in \mathscr{S}_2$. Compute $b_i = a_i + d \in \mathscr{S}_2$, for each $1 \leq i \leq n$. This value d helps to confound an attacker's attempt to brute-force the secret σ.
6. The public key is $(n, k, b_1, b_2, \ldots, b_n)$, and the secret key is $(H_\sigma, H_\sigma^{-1}, \sigma, d, P)$.

Encryption:

1. A plaintext $M = (m_1, \ldots, m_n)$ is a binary vector of length n with exactly k ones. Any message of size $\lfloor \log_2 \binom{n}{k} \rfloor$ bits can be converted to a valid plaintext M by a straightforward encoding [7].
2. The ciphertext $c \in \mathscr{S}_2$ is computed as

$$c = \sum_{i=1}^{n} m_i b_i = \sum_{i \in M'} b_i$$

where M' is the set of nonzero indices of m.

Decryption:

1. Compute $r = c - kd$ where $kd = \underbrace{(d + d + \cdots + d)}_{k \text{ times}} \in \mathscr{S}_2$.
2. Calculate $u = H_\sigma^{-1}(r) \in \mathscr{S}_1$ and solve the k-unique SSP on P and u over $(\mathscr{S}_1, *)$ to obtain a subset $P' \subset P$ of size k. If $p_i \in P'$, then $m_i = 1$, else $m_i = 0$.
3. The recovered plaintext is $M = (m_1, \ldots, m_n)$.

Proof of Soundness: In decryption,

$$\begin{aligned}
u &= H_\sigma^{-1}(r) \\
&= H_\sigma^{-1}(c - kd) \\
&= H_\sigma^{-1}(\sum_{i=1}^{n} m_i b_i - kd) \\
&= H_\sigma^{-1}(\sum_{i=1}^{n} m_i (a_i + d) - kd) \\
&= H_\sigma^{-1}(\sum_{i=1}^{n} m_i a_i) \quad \text{as exactly } k \text{ of the } m_i \text{ are 1} \\
&= \prod_{i=1}^{n} H_\sigma^{-1}(m_i a_i) \\
&= \prod_{i : m_i = 1} p_i
\end{aligned}$$

where $\prod$ and $\sum$ denote application of the semigroup operation over $(\mathscr{S}_1, *)$ and $(\mathscr{S}_2, +)$, respectively. Since $P = \{p_1, \ldots, p_n\}$ was chosen so that it has the k-unique Subset Sum property over $(\mathscr{S}_1, *)$ and that solving the k-unique SSP on P is easy, the decryption step will recover the set P' containing exactly those p_i for which $m_i = 1$, and the proof is complete.

4 Using the Damgård-Jurik Homomorphism

The finite field DLP is the most natural example for the homomorphism H_σ characterized in Section 3.1. But due to its superpolynomial complexity, SSP-based cryptosystems requiring the use of the DLP are not

practical on classical computers. Although it is a workable solution to use finite fields in which the DLP computation is easy, it severely restricts the possible choices for the secret key and makes the cryptanalysis easy. In this section, we show that the Damgård-Jurik generalization [9] of the Paillier cryptosystem [26] provides a homomorphism map satisfying all of the requirements of Section 3.1, and we apply this map to design a practical SSP-based cryptosystem.

Paillier [26] introduced the composite residuosity class problem and presented a probabilistic homomorphic public key cryptosystem with the trapdoor permutation property. For $t = pq$ where p and q are prime and $g \in \mathbb{Z}^*_{t^2}$ having order t, the Paillier cryptosystem provides an isomorphism $\mathscr{E}_g$ such that

$$\begin{aligned} \mathscr{E}_g : \mathbb{Z}_t \times \mathbb{Z}^*_t &\rightarrow \mathbb{Z}^*_{t^2} \\ (x,y) &\mapsto g^x * y^t \pmod{t^2}. \end{aligned}$$

It has a homomorphic property

$$\mathscr{E}_g(x_1,y_1) * \mathscr{E}_g(x_2,y_2) = \mathscr{E}_g(x_1+x_2, y_1*y_2).$$

In the usual use of Paillier, x is the message plaintext, and y is a randomization factor used to achieve semantic security.

Damgård and Jurik [9] generalize the Paillier Cryptosystem using computation modulo t^{s+1}, for an integer s such that $1 \leq s < \min(p,q)$. For an element $g \in \mathbb{Z}^*_{t^{s+1}}$ of order t^s, the corresponding map $\mathscr{E}^s_g$ can be represented as follows:

$$\begin{aligned} \mathscr{E}^s_g : \mathbb{Z}_{t^s} \times \mathbb{Z}^*_t &\rightarrow \mathbb{Z}^*_{t^{s+1}} \\ (x,y) &\mapsto g^x * y^{t^s} \pmod{t^{s+1}}. \end{aligned}$$

$\mathscr{E}^s_g$ has the same homomorphic property as $\mathscr{E}_g$:

$$\mathscr{E}^s_g(x_1,y_1) * \mathscr{E}^s_g(x_2,y_2) = \mathscr{E}^s_g(x_1+x_2, y_1*y_2).$$

To obtain a homomorphism H_σ as defined in Section 3.1, we need to consider the decryption function (say $\mathscr{D}^s_g$) corresponding to $\mathscr{E}^s_g$ such that

$$\mathscr{D}^s_g : \mathbb{Z}^*_{t^{s+1}} \rightarrow \mathbb{Z}_{t^s} \times \mathbb{Z}^*_t.$$

Given a ciphertext $c = g^x y^{t^s} \in \mathbb{Z}^*_{t^{s+1}}$, and the factorization of t, $\mathscr{D}^s_g$ returns $x \in \mathbb{Z}_{t^s}$ and $y \in \mathbb{Z}^*_t$ in polynomial time. $\mathscr{D}^s_g$ provides a homomorphism of the form

$$\mathscr{D}^s_g(c_1 * c_2) = (x_1+x_2, y_1*y_2)$$

We note that this encryption-decryption pair $(\mathscr{E}^s_g, \mathscr{D}^s_g)$ can not be directly used as our desired $(H_\sigma^{-1}, H_\sigma)$, since the homomorphism is additive (as required) on the first coordinate, but multiplicative on the second. We overcome this obstacle by considering the deterministic version of Damgård-Jurik that fixes $y = 1$, yielding a bijection between $(\langle g \rangle, *)$ and $(\mathbb{Z}_{t^s}, +)$. Note that we use the Damgård-Jurik cryptosystem only as a homomorphic map; the security of our cryptosystem does not depend upon the security of the special case of the Damgård-Jurik cryptosystem with $y = 1$.

The modified decryption function $\hat{\mathscr{D}}^s_g$ is a homomorphism such that

$$\begin{aligned} \hat{\mathscr{D}}^s_g(c_1 * c_2) &= (x_1+x_2) \\ &= \hat{\mathscr{D}}^s_g(c_1) + \hat{\mathscr{D}}^s_g(c_2) \end{aligned}$$

and the corresponding encryption function $\hat{\mathscr{E}}^s_g$ satisfies

$$\hat{\mathscr{E}}^s_g(x_1) * \hat{\mathscr{E}}^s_g(x_2) = \hat{\mathscr{E}}^s_g(x_1+x_2).$$

As $(\hat{\mathscr{D}}^s_g, \hat{\mathscr{E}}^s_g)$ satisfies all the required properties of $(H_\sigma, H_\sigma^{-1})$, we use them to define a practical SSP-based cryptosystem, which we present next.

4.1 The Proposed Scheme

Key Generation:

1. Fix size parameters $n, k, s \in \mathbb{Z}^+$, with $k < n/2$ and $k < s$.
2. Randomly choose two primes p and q and compute $t = pq$. Pick an element $\alpha \in_R \mathbb{Z}_{t^s}^*$, and verify that $g = \alpha t + 1 \in \mathbb{Z}_{t^{s+1}}$ has order t^s.
3. Select at random n pairwise coprime integers $1 < p_1, p_2, \ldots, p_n < t^{(s+1)/k}$ such that all of them (considered as elements of $\mathbb{Z}_{t^{s+1}}^*$) belong to the subgroup $\langle g \rangle$. Since $1/t$ of the elements of $\mathbb{Z}_{t^{s+1}}^*$ belong to $\langle g \rangle$, it behooves us to have $n \ll 1/t * t^{(s+1)/k} = t^{(s-k+1)/k}$, and this is why we pick $s > k$. Note that this also rules out the use of the plain Paillier cryptosystem (which is just the Damgård-Jurik cryptosystem with $s = 1$).
4. Compute $a_i = \hat{\mathscr{D}}_g^s(p_i) \in \mathbb{Z}_{t^s}$ for $1 \le i \le n$.
5. Randomly choose an integer $d \in \mathbb{Z}_{t^s}$ and compute $b_i = a_i + d \bmod t^s$ for each $1 \le i \le n$.
6. The public key is $(n, k, b_1, b_2, \ldots, b_n)$, and the private key is $(p, q, t, s, g, d, p_1, p_2, \ldots, p_n)$.

Encryption:

1. A plaintext $M = (m_1, \ldots, m_n)$ is a binary vector of length n with exactly k ones, as before.
2. The ciphertext $c \in \mathbb{Z}_{t^s}$ is computed as $c = \sum_{i=1}^n m_i b_i = \sum_{i \in M'} b_i$ where M' is the set of nonzero indices of m. Note that although the public key consists of n numbers, each s times the size of t, encryption consists simply of adding some k of them together.

Decryption: Compute $r = c - kd \in \mathbb{Z}_{t^s}$ and $u = g^r \in \mathbb{Z}_{t^{s+1}}^*$. Treating u as an integer in the range $0 < u < t^{s+1}$, check if $p_i | u$ for each $1 \le i \le n$. If it does, set $m_i = 1$; else set $m_i = 0$. The decrypted message is then $M = (m_1, \ldots, m_n)$.

Proof of Soundness: In decryption,

$$\begin{aligned} u &\equiv g^r \equiv g^{c-kd} \equiv g^{\sum_{i=1}^n m_i b_i - kd} \equiv g^{\sum_{i=1}^n m_i (a_i + d) - kd} \bmod t^{s+1} \\ &= g^{\sum_{i=1}^n m_i a_i} = \prod_{i \in M'} g^{a_i} = \prod_{i \in M'} p_i \end{aligned}$$

Note that the p_i were selected so that they are pairwise coprime, and that the product of any k of them is less than t^{s+1}. Therefore, determining which of the p_i divide this u will reconstruct the original M', and thus M.

4.2 Variants

In this section, we consider some possible variants of the above scheme, which significantly improve the security of the private key in the scheme.

Picking non-coprime p_i

We suggested above to pick the p_i so that they are pairwise coprime. This condition is sufficient to ensure that a product of some subset of the p_i can be uniquely factored in polynomial time. However, this condition is not strictly necessary. For example, it also suffices that for each of the p_i, there is *some* prime power which divides it, and which divides none of the other p_i. This can be easily checked by ensuring that $p_i^2 \not| \prod_{j=1}^n p_j$ for each i. The non-unique factors of the p_i can be ignored when recovering the factorization of u, and choosing the p_i in this way greatly increases the number of possible keys for given size parameters.

Variable k
Some utility may be obtained by having the sender choose the value of k, rather than it being a fixed part of the public key, as it increases the number of possible ciphertexts for a given plaintext. Note that in order to decrypt correctly, the receiver needs to know k, so that she can calculate $r = c - kd$. Therefore, in this case, k would need to appear as part of the ciphertext. It is important that k does not depend on the plaintext, of course, as that would leak information to an observer. Minor modifications and care are needed in the choosing of the parameters in this case, but we find that these are easily handled.

Prime or multi-prime t
We stated that t should be the product of the two primes p and q. However, it is also possible use a prime t, as the required multiplicative group also exists in that case. Further, as t is part of the *private* key in this system, and not disclosed, it is acceptable to use a larger number of smaller (distinct) primes. This will involve the use of a straightforward extension of the Damgård-Jurik homomorphism to the multi-prime case. Choosing a prime or multi-prime t significantly increases the search space for an attacker.

5 Security Analysis

We observe that the public keys produced by our cryptosystem are instances of the k-unique SSP from Definition 2.4, where only subsets of fixed size at most k need to have unique sums; there will likely be pairs of larger subsets (or pairs of subsets of unequal size) that have equal sums.

Fact *The $\{b_i\}$ portion of every public key obtained using the scheme from Section 4.1 has the k-unique Subset Sum property over $(\mathbb{Z}_{t^s}, +)$.*

Proof. Let $(n, k, b_1, \ldots, b_n)$ be a public key generated by our scheme. Let $B = \{b_1, \ldots, b_n\}$, and let B_1, B_2 be two subsets of B, each of size k, such that $\sum_{b_i \in B_1} b_i = \sum_{b_i \in B_2} b_i$. Let this common sum be c. For $i \in \{1,2\}$, let M_i be the index set of B_i; that is, $M_i = \{j : b_j \in B_i\}$. Because decryption has been proven to be sound, we must have that c has a unique decryption, so $M_1 = M_2$, and therefore $B_1 = B_2$.

Now suppose B_1 and B_2 are subsets of B of the same size $k' < k < n/2$ that have the same sum. Then $B' = B \backslash (B_1 \cup B_2)$ has at least $n - 2k' > k - k'$ elements, since $n > k + k'$. Let B_* be a set of any $k - k'$ elements of B'. Then $B_1 \cup B_*$ and $B_2 \cup B_*$ are subsets of size k with the same sum, and so by the above, $B_1 = B_2$. □

Thus, the decryption problem in our cryptosystem is an instance of the k-unique SSP. Of course, being an instance of k-unique SSP does not provide any evidence towards the hardness of our decryption problem; however, it certainly puts it in the security range of the OTU scheme. Next we discuss concrete security properties of our cryptosystem.

5.1 Semantic Security

The encryption scheme defined in Section 4.1 is not probabilistic in nature and consequently, does not provide semantic security. However, as with other deterministic encryption schemes, it is possible to achieve semantic security by padding the message with a random string before encrypting. Here, a message is padded with random string before deterministically encoding it to a n bit plaintext of weight k. After decryption and subsequent decoding, the random padding is removed to obtain the message sent. Note that the length of the random padding depends upon the security requirements; 80 bits is usually an appropriate value.

5.2 Attacks

We consider the system to be secure against an attack if an adversary needs to do approximately 2^{80} computations for that attack to be successful. We next discuss important attacks on our system and restrictions on our parameters to avoid these attacks.

Lattice-based attacks using the LLL algorithm have been the single most important tool used to cryptanalyze existing SSP-based cryptosystems. [6, 13, 14, 22] As already discussed, the computational complexity of the LLL algorithm is very large, requiring $O(n^6 \log^3(\max_{\mathscr{B}}))$ operations, where $\max_{\mathscr{B}} > t^s$ for our cryptosystem. For $n \geq 500$ and t^s in the 1500-bit range, a single LLL computation is beyond the capacity of an adversary with a computational power of 2^{80} operations.

Note that recent floating-point versions of LLL, such as Schnorr's algorithm [28] and the L^2 algorithm [21] provide a bit smaller complexity bounds, such as $O(n^5 \log^2(\max_{\mathscr{B}}))$. However, these algorithms also have associated floating point errors and the reduction in security is still tolerable in many practical scenarios. Further, lattice-dimension reduction attacks for convolution modular lattices [19] are not applicable to solving our non-convolutional SSP instances, and our choice of $k = \omega(\log n)$ reduces the success probability for these attacks to be negligible in any event.

In order to achieve semantic security, and the ability to encrypt a large number of messages, we would like both the size of our message space and that of the random padding to exceed 80 bits. Therefore, we need to select n and k such that $\binom{n}{k} \geq 2^{160}$. Further, the complexity of the dynamic programming method to solve an SSP instance is at least $\Omega(\max_{\mathscr{B}})$. Here, $\max_{\mathscr{B}} = t^s = 2^{s\tau}$, where $\tau = \log_2 t$, so we choose $s\tau \geq 80$.

5.3 Choosing parameters

An important consideration to the security of our scheme is the choice of the parameters n, k, and s (and relatedly t). Considering various attacks discussed above, we apply following restriction on the parameters n, k, s, and t.

- $\binom{n}{k}$ must be sufficiently large to resist brute-force attacks and also to provide sufficient message size to realize padding-based semantic security.
- n, s, and $\tau = \log t$ must be large enough such that lattice-based attacks, requiring at least $O(n^6 s^3 \tau^3)$ operations, are not possible
- $n \ll t^{(s-k+1)/k}$ (from section 4.1) so that there are a large number of private key choices for given parameter sizes

Strategy

We suggest a following strategy to choose the parameters for our cryptosystem.

1. Choose n sufficiently large so that computational resistance against lattice attacks is feasible.
2. Select n and k large enough to obtain message size $\geq 2^{160}$.
3. Find τ (the bitlength of t) and s such that $s > k$ and $n \ll (2^\tau)^{(s-k+1)/k} = 2^{(s-k+1)\tau/k}$. That is, $\frac{k}{s-k+1} \log_2 n \ll \tau$.
4. Select t to be a square-free integer of size τ such that all of its prime factors are greater than s. Note that it is acceptable for t to be smaller than a usual RSA modulus, since it is not made public. Damgård-Jurik is used for its discrete log homomorphism, not for its cryptographic strength. However, t should still be large enough that an attacker cannot brute-force all values of t (though the confounding factor of d, which is of size $s\tau$ bits, may be enough to thwart that).

The public key size is around $ns\tau$ bits, while encryption requires $O(ks\tau)$ additions.

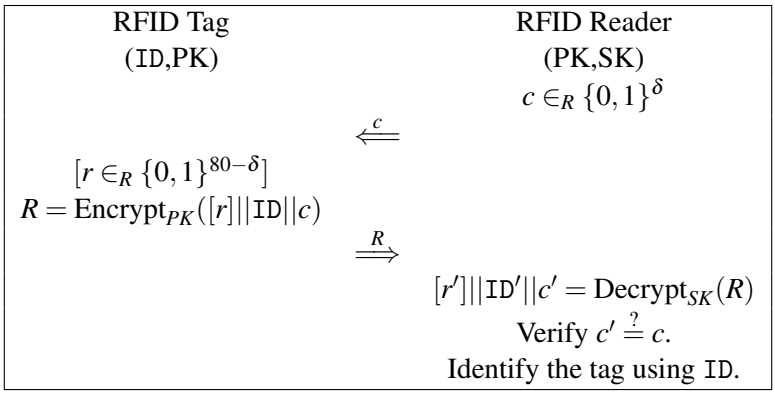

Figure 1: Privacy-preserving Identification Protocol

Example
For the practical security of constrained devices, we find the parameter set $(n,k,s,\tau) = (500,30,35,50)$ to be convenient. Here, $\binom{n}{k}$ is approximately 2^{160}, the public key is 109 KBytes and a lattice-based attack requires around $500^6 35^3 50^3 > 2^{86}$ computations using the LLL algorithm. Also, 2^{40} choices for t (assuming prime and multi-prime t), and $t^s \approx 2^{1750}$ choices for g and d make private-key extraction difficult.

6 Application: RFID Security and Privacy

Cryptosystems based on the knapsack and subset-sum problems provide feasible options for authentication and identification protocols for RFID chips as the encryption steps for these systems are practical for computationally constrained devices. Further, with the availability of a randomness source on a chip, it is easily possible to make these protocols privacy preserving. Cui et al. [8] use the Niederreiter asymmetric encryption scheme [23] to define a privacy-preserving authentication protocol for RFID systems. However, Brickell and Odlyzko [4] have cryptanalyzed the Niederreiter encryption scheme and have suggested two polynomial-time attacks. We observe that our SSP-based cryptosystem, which requires considerably less computation than that of the Niederreiter cryptosystem and is computationally secure against all known attacks on the SSP-based cryptosystems, certainly provides a promising choice. Although previous SSP-based cryptosystems such as OTU [24] were also suitable for RFID chips, the need of a DLP oracle in the key generation step has been a major obstacle in considering them for practical use. Our encryption scheme using the homomorphism of the Damgård-Jurik system overcomes this obstacle and provides a polynomial-time key generation scheme.

6.1 Privacy-Preserving Identification Protocol

In this protocol, we assume that there is a public key-private key pair associated with a valid RFID reader and the reader's public key is embedded in a RFID chip. The RFID chip also contains an identification string ID, which it needs to present to the reader to identify itself. In order to avoid cloning of the RFID signals or the corresponding counterfeiting of an associated object, we need a mechanism for secure and private transfer of the identification string from the chip to the reader. We achieve this using our encryption protocol. In order to avoid a replay attack, where an eavesdropper replays a signal from a legitimate RFID chip, the reader sends randomly generated challenge string c. Subject to the availability of an randomness source on the RFID chip, it is also possible to make the protocol privacy-preserving. Here, the chip pads the identifying string ID and the received challenge c with a random string r, so that its responses for two identical challenges from an active attacker differ. Note that in the absence of a randomness source on

the RFID chip, deterministic responses make privacy-preserving identification impossible against an active attacker. We present a privacy-preserving identification protocol in Figure 1. Here, the reader's challenge c and the optional randomness r combine to form the 80 bits of randomness required for semantic security.

6.2 A Multi-chip Identification Scheme Using the Chinese Remainder Theorem

Although the parameter choices presented in the example in Section 5.3 can be used to develop a practical and secure identification scheme, the large public-key size can be a concern during the implementation. In this section, we mitigate this problem using Chinese remainder theorem (CRT) based public key distribution over multiple RFID chips. In this multi-chip solution, we distribute the RFID reader's public key over multiple RFID chips and then appropriately combine the responses received from the involved chips to decrypt the message and determine the identity of the product those are attached to.

The number of RFID chips to be used in this distributed protocol is determined by the number of pairwise coprime factors of the parameter t^s. For $t = pq$ used in the proposed scheme (Section 4.1), we can use two RFID chips. We divide the public-key parameters $b_i \in \mathbb{Z}_{t^s}$ for $i \in [1,n]$ into $b_i^{(p)} = b_i \bmod p^s$ and $b_i^{(q)} = b_i \bmod q^s$, and along with an identical identity string $\mathtt{ID}$, put all $b_i^{(p)}$ on one chip and all $b_i^{(q)}$ on the other. The identification protocol remains exactly the same except during the decryption step at the reader. During the decryption step, the reader collects ciphertext parts $R^{(p)}$ and $R^{(q)}$ received from the two chips, computes $R^{(p)} \bmod p^s$ and $R^{(q)} \bmod q^s$ and extracts the corresponding $R \bmod t^s$ using CRT. We assume that both chips are physically close to each other so that they both receive the challenge sent by the reader. In other words, we assume that the reader sends the same challenge c to both chips. Further, for this identification to be privacy preserving, both chips have to use the same r value, which is possible using a pseudorandom number generator with the same seed value on both chips.

Note that an active attacker may somehow target just one of the two chips and destroy the synchrony of the generated randomness and similarity of the plaintext at the two chips. However, this is an example of denial of service (DoS) attack. If two chips are in close proximity, then an attacker that can selectively target one of the two chips can also easily destroy the chip by other physical means. Therefore, in practice, this desynchronization attack is not relevant. Next, we describe the proof of soundness, security and advantages of our approach.

Proof of Soundness

We have $\gcd(p^s, q^s) = 1$. Let $M = \mathsf{Encode}([r]||\mathtt{ID}||c)$ be the plaintext of length n with exactly k ones (which will be encrypted by both chips) and M' is the set of nonzero indices in M. Given $R^{(p)} = \sum_{i \in M'} b_i^{(p)}$ and $R^{(q)} = \sum_{i \in M'} b_i^{(q)}$,

$$\begin{aligned} R &= \mathsf{CRT\text{-}Reconstruct}(R^{(p)} \bmod p^s, R^{(q)} \bmod q^s) \\ &= \mathsf{CRT\text{-}Reconstruct}(\sum_{i \in M'} b_i^{(p)} \bmod p^s, \sum_{i \in M'} b_i^{(q)} \bmod q^s) \\ &= \sum_{i \in M'} \mathsf{CRT\text{-}Reconstruct}(b_i^{(p)}, b_i^{(q)}) \\ &= \sum_{i \in M'} b_i \bmod t^s \end{aligned}$$

Here, function $\mathsf{CRT\text{-}Reconstruct}$ represents a reconstruction algorithm that, given the remainders, computes the dividend in the CRT setting. The rest of the soundness proof remains the same as that of the basic scheme (Section 4.1), by substituting $r = R$.

Security Analysis

When compared with the original instance $(b_1, b_2, \cdots, b_n)$, the reduction in the complexity of LLL attacks over each of instances $(b_1^{(p)}, b_2^{(p)}, \cdots, b_n^{(p)})$ and $(b_1^{(q)}, b_2^{(q)}, \cdots, b_n^{(q)})$ is η^3, where η represents the number of parts into which the k-unique SSP instance is divided; η is also the number of pairwise coprime factors of t ($\eta = 2$ here). For $\eta^3 = 8$, an LLL attack over $(b_1^{(p)}, b_2^{(p)}, \cdots, b_n^{(p)})$ or $(b_1^{(q)}, b_2^{(q)}, \cdots, b_n^{(q)})$ is just 8 times faster than that over the original instance. Consequently, as required, LLL attacks remain infeasible. Further, although the attacker now has the additional knowledge that all $b_i^{(p)} < p^s$ and all $b_i^{(q)} < q^s$, guessing the private-key parameters is still beyond the reach of a polynomially bounded attacker. Finally, the complexity of the brute force attack remains the same $\binom{n}{k}$.

Advantages

Although we describe our mutli-chip solution for $\eta = 2$, it can be seamlessly extended to more chips using multi-prime t having more than 2 coprime factors ($\eta > 2$). Further, using the above solution, the size of the public key to be stored per chip gets reduced by a factor of η. For the example described in Section 5.3, the above multi-chip solution reduces the size of public key to $(109/\eta)$ KBytes. The complexity of the encryption step also decreases by a factor of η.

7 Conclusion

In this paper, we provided a general construction for SSP-based cryptosystems and identified the requirements for the underlying homomorphism. We used the homomorphism from the Damgård-Jurik cryptosystem to provide the a polynomial-time key generation procedure for the OTU cryptosystem, leading to a practical SSP-based cryptosystem.

We observed that our cryptosystem is computationally secure against lattice-based attacks and presented an application of these cryptosystems in defining an efficient RFID security and privacy solution.

Acknowledgements

We would like to thank Greg Zaverucha for his helpful discussion throughout this work. This research was supported by a David R. Cheriton graduate scholarship from the University of Waterloo, by MITACS, and by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada.

A The Okamoto-Tanaka-Uchiyama Cryptosystem

Here we show the basic version of the OTU cryptosystem [24]. Note the use of a quantum computation in step 3 of the key generation, which our system avoids.

Key Generation:

1. Fix size parameters n, k from $\mathbb{Z}^+$
2. Randomly choose a prime p, a generator g of the group $\mathbb{Z}_p^*$, and n coprimes $1 < p_1, \ldots, p_n < p$ such that $\prod_{j=1}^{k} p_{i_j} < p$ for any subset $\{p_{i_1}, p_{i_2}, \ldots, p_{i_k}\}$ of $\{p_1, p_2, \ldots, p_n\}$.
3. Use Shor's quantum algorithm to compute $a_i = \log_g p_i$ in $\mathbb{Z}_p^*$ for $1 \leq i \leq n$.
4. Randomly choose an integer $d \in \mathbb{Z}/(p-1)\mathbb{Z}$.
5. Compute $b_i = (a_i + d) \pmod{p-1}$, for each $1 \leq i \leq n$.

6. The public key is $(n, k, b_1, b_2, \ldots, b_n)$, and the secret key is $(g, d, p, p_1, p_2, \ldots, p_n)$.

Encryption:

1. Let the message $M = (m_1, \ldots, m_n)$ be a binary vector with exactly k ones. The ciphertext is computed as
$$c = \sum_{i=1}^{n} m_i b_i = \sum_{i \in M'} b_i$$
where M' is the set of nonzero indices of m.

Decryption:

1. Compute $r = (c - kd) \pmod{p-1}$.
2. Compute u as the least positive representative of $g^r \pmod{p}$.
3. For i from 1 to n, if $p_i \mid u$ then $m_i = 1$, otherwise $m_i = 0$.

Proof of Soundness

In decryption,

$$\begin{aligned} u &\equiv g^r \equiv g^{c-kd} \equiv g^{\sum_{i=1}^n m_i b_i - kd} \equiv g^{\sum_{i=1}^n m_i (a_i + d) - kd} \pmod{t^{s+1}} \\ &= g^{\sum_{i=1}^n m_i a_i} = \prod_{i \in M'} g^{a_i} = \prod_{i \in M'} p_i \end{aligned}$$

Now, by testing which $p_i \mid u$, one can recover the k nonzero bits of M.

References

[1] G. Avoine. Security and privacy in RFID systems. `http://www.avoine.net/rfid/`, 2009. Accessed April 2009.

[2] R. Bose and S. Chowla. Theorems in the additive theory of numbers. *Comment. Math. Helv.*, 37:141–147, 1962.

[3] E. Brickell. Solving Low Density Knapsacks. In *Advances in Cryptology—CRYPTO*, pages 25–37, 1983.

[4] E. Brickell and A. Odlyzko. Cryptanalysis: a survey of recent results. *Proceedings of the IEEE*, 76(5):578–593, 1988.

[5] B. Chor and R. L. Rivest. A knapsack type public key cryptosystem based on arithmetic in finite fields. In *Advances in Cryptology—CRYPTO*, pages 54–65, 1985.

[6] M. Coster, B. LaMacchia, and A. Odlyzko. An Improved Low-Denisty Subset Sum Algorithm. In *Advances in Cryptology—EUROCRYPT*, pages 54–67, 1991.

[7] T. Cover. Enumerative source encoding. *IEEE Trans. on Information Theory*, 19(1):73–77, 1973.

[8] Y. Cui, K. Kobara, K. Matsuura, and H. Imai. Lightweight Asymmetric Privacy-Preserving Authentication Protocols Secure against Active Attack. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2007*, pages 223–228, 2007.

[9] I. Damgård and M. Jurik. A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In *Public Key Cryptography 2001*, pages 119–136, 2001.

[10] N. D. Elkies. An improved lower bound on the greatest element of a sum-distinct set of fixed order. *J. Comb. Theory Ser. A*, 41(1):89–94, 1986.

[11] P. Erdős. Problems and results from additive number theory. *Colloq. Théorie des nombres, Bruxells*, pages 127–137, 1955.

[12] R. K. Guy. *Unsolved Problems in Number Theory*. Springer-Verlag, New York, 2 edition, 1994.

[13] T. Izu, J. Kogure, T. Koshiba, and T. Shimoyama. Low-density attack revisited. *Designs, Codes and Cryptography*, 43(1):47–59, April 2007.

[14] A. Joux and J. Stern. Improving the critical density of the Lagarias-Odlyzko attack against subset sum problems. In *8th International Symposium on Fundamentals of Computation Theory*, pages 258–264, 1991.

[15] A. Juels. RFID security and privacy: a research survey. *IEEE Journal on Selected Areas in Communications*, 24(2):381–394, 2006.

[16] J. Lagarias and A. Odlyzko. Solving Low-Density Subset Sum Problems. In *IEEE Symposium on Foundations of Computer Science*, pages 1–10, 1983.

[17] M. Lai. Knapsack Cryptosystems: The Past and the Future. Technical report, Department of Information and Computer Science,University of California, 2001. `http://www.ics.uci.edu/~mingl/knapsack.html`.

[18] A. K. Lenstra, H. W. Lenstra Jr., and L. Lovász. Factoring Polynomials with Rational Coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[19] A. May and J. H. Silverman. Dimension Reduction Methods for Convolution Modular Lattices. In *Cryptography and Lattices, International Conference (CaLC)*, pages 110–125, 2001.

[20] R. Merkle and M. Hellman. Hiding Information and Signatures in Trapdoor Knapsacks. *IEEE Transactions On Information Theory*, 24(5):525 – 530, 1978.

[21] P. Q. Nguyen and D. Stehlé. Floating-Point LLL Revisited. In *Advances in Cryptology—EUROCRYPT*, pages 215–233, 2005.

[22] P. Q. Nguyen and J. Stern. Adapting Density Attacks to Low-Weight Knapsacks. In *Advances in Cryptology—ASIACRYPT*, pages 41–58, 2005.

[23] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15:159–166, 1986.

[24] T. Okamoto, K. Tanaka, and S. Uchiyama. Quantum Public-Key Cryptosystems. In *Advances in Cryptology—CRYPTO*, pages 147 – 165, 2000.

[25] K. Omura and K. Tanaka. Density Attack to the Knapsack Cryptosystems with Enumerative Source Encoding. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E87-A(6):1564–1569, June 2004.

[26] P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Advances in Cryptology—EUROCRYPT*, pages 223–238, 1999.

[27] C. Papadimitriou. On the complexity of unique solutions. *J. ACM*, 31(2):392–400, 1984.

[28] C. Schnorr. A More Efficient Algorithm for Lattice Basis Reduction. *J. Algorithms*, 9(1):47–62, 1988.

[29] A. Shamir. A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. In *IEEE Symposium on Foundations of Computer Science*, pages 145–152, 1982.

[30] P. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.