Non-zero Inner Product Encryptions: Strong Security under Standard Assumptions

Tapas Pal and Ratna Dutta

Department of Mathematics, Indian Institute of Technology Kharagpur,
Kharagpur-721302, India
tapas.pal@iitkgp.ac.in, ratna@maths.iitkgp.ernet.in

Abstract. Non-zero inner product encryption (NIPE) allows a user to encrypt a message with its attribute vector and decryption is possible using a secret-key associated with a predicate vector if the inner product of the vectors is non-zero. The concept of NIPE was put forth by Katz, Sahai and Waters (EUROCRYPT 2008). Following that many NIPE constructions were proposed along with interesting applications. The security of all these works is based on hardness assumptions in pairing-friendly groups. Recently, Katsumata and Yamada (PKC 2019) built a NIPE relying on the Learning-with-Errors (LWE) problems, however, their system practically lags behind for providing only *selective* security with significantly large sizes of master public-key, secret-keys and ciphertexts. Despite its cryptographic importance, past history of NIPE is not convincing in terms of both security and practical efficiency as the schemes are either selectively secure or depend on bilinear maps.

In this paper, our goal is to construct adaptively secure efficient NIPEs. Firstly, we provide *adaptively* secure public-key NIPE under the standard Decision Diffie-Hellman (DDH) assumption that enables one to encrypt messages of sufficiently small length. To overcome this limitation we rely on the Decision Diffie-Hellman-f (DDH-f) and the Hard Subgroup Membership (HSM) assumptions proposed by Castagnos et al. in ASIACRYPT 2018. Consequently, we construct two pNIPEs, adaptively secure under the DDH-f and HSM assumptions respectively, both are capable of encrypting large messages with inner products over integers. We upgrade these two pNIPEs so that it can encrypt messages with unbounded inner products modulo an arbitrary large prime p. In addition, utilizing inner product functional encryptions we provide attribute-hiding public-key NIPEs depending on the DDH, DDH-f, HSM, LWE, Decision Composite Reciprocity assumptions and establish full-hiding private-key NIPEs based on the Decision linear and Symmetric External Diffie-Hellman assumptions.

Keywords: non-zero inner product encryptions, adaptive security, payload-hiding, attribute-hiding, full-hiding.

1 Introduction

Using plain public-key encryption (PKE), a receiver learns the whole message if he possesses the correct secret-key, otherwise gains nothing about the plaintext. In many real-life applications, it may be necessary to reveal only a function of the original message instead of the full plaintext. Suppose an encrypter (say the CEO of a company) wants to disclose only a certain portion (say emails received except from the domain *@.ac.in*) of his message (all the emails) to a particular person or group, then he cannot give away the secret-key corresponding to the public-key that was used to encrypt the message as that would unveil the whole plaintext. In such a scenario, plain PKEs are unable to fulfil users' requirement. To remedy *all-or-nothing* type encryption, plain PKEs are refined over the years into more advanced primitives like *identity-based encryption* (IBE) [36,8], *broadcast encryption* (BE) [18], *attribute-based encryption* (ABE) [35,24]. All these primitives can be combined into a single class of encryptions called *functional encryption* (FE) introduced much later by Boneh et al. [10]. Realizing FE for general class of functions [21,19] employs heavy cryptographic tools like multi-linear maps or obfuscation, and as a result these existing constructions are inefficient for day-to-day use. However, there are FEs for certain type of functionalities such as Boolean formulae, inner product predicate, keyword search [24,26,7] that are accomplished from standard and well-understood assumptions, hence are eligible for practical implementation.

In ABE, a secret-key $\mathsf{sk}_{\boldsymbol{y}}$ is generated corresponding to a predicate $\boldsymbol{y}$ and a ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ for a message M is associated with an attribute $\boldsymbol{x}$. The decryption successfully recovers the message M from $\mathsf{sk}_{\boldsymbol{y}}$ and $\mathsf{CT}_{\boldsymbol{x}}$ if a relation $R(\boldsymbol{x}, \boldsymbol{y})$ holds. This paper studies a particular type of ABE, called *non-zero inner product encryption* (NIPE) [26] that considers the predicate and attribute space to be $\mathbb{Z}^l$ (resp. $\mathbb{Z}_p^l$ for some prime p) for a natural number l and the relation R is defined as $R(\boldsymbol{x}, \boldsymbol{y}) = 1$ if and only if $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0$ over $\mathbb{Z}$ (resp. over $\mathbb{Z}_p$). Specifically, a public-key NIPE (pNIPE) depends on a trusted authority that generates a master secret-key MSK and a master public-key MPK. Corresponding to each predicate vector $\boldsymbol{y}$, the authority provides a predicate secret-key $\mathsf{sk}_{\boldsymbol{y}}$ computed using MSK. An encrypter uses MPK to produce a ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ for a message M (which is an integer) associated with an attribute $\boldsymbol{x}$. Given a secret-key $\mathsf{sk}_{\boldsymbol{y}}$, any user gets the message M from the ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ if $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0$, otherwise learns nothing about the message. In private-key NIPE (sNIPE), the authority publishes a public parameter instead of a master public-key and a message is encrypted using the master secret-key.

1.1 Security of NIPE

In literature, there are three indistinguishability based security notions [4,34] for a pNIPE: *selectively*, *co-selectively* and *adaptively* secure. In selective security the adversary $\mathcal{A}$ has to commit on the challenge attribute before the setup phase, but $\mathcal{A}$ can adaptively query for the secret-keys corresponding to a polynomial number of predicate vectors. A dual of this model is called co-selective security where $\mathcal{A}$ declares its secret-key queries before the setup phase, but $\mathcal{A}$ can select the challenge attribute based on the information gained from the secret-key queries. In contrast, when the adversary has the freedom to choose the challenge attribute as well as the predicate vectors (to be queried for the secret-keys) after the setup phase, the model is referred to adaptive security. The strongest, at the same time a practical model is adaptive security which is the main interest of this paper. In *payload-hiding* pNIPE (PH-pNIPE), it is required that no probabilistic polynomial time (PPT) adversary having (adaptively) chosen two messages M_0, M_1 and a challenge attribute $\boldsymbol{x}^*$ can distinguish encryption of one of these messages with a probability significantly greater than 1/2. The adversary may query for secret-keys $\mathsf{sk}_{\boldsymbol{y}}$ for a polynomial number of predicate vectors $\boldsymbol{y}$ satisfying $\langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0$. Sometimes, a user may want the ciphertext corresponding to his message to not leak any information about the attribute $\boldsymbol{x}$ (except for the fact that $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is zero or non-zero) as $\boldsymbol{x}$ may contain sensitive information about the user's credentials. This additional security feature is guaranteed by the *attribute-hiding*[1] pNIPE (AH-pNIPE) where the adversary is asked to submit two attribute-message pairs $(\boldsymbol{x}_b, M_b)$ for $b \in \{0, 1\}$. Given encryption for a pair $(\boldsymbol{x}_b, M_b)$, it is required that the probability of the distinguishing advantage (or guessing b) of any PPT adversary is less than 1/2. The secret-key queries for the predicate vectors $\boldsymbol{y}$ are restricted to satisfy that $\langle \boldsymbol{x}_0, \boldsymbol{y} \rangle = \langle \boldsymbol{x}_1, \boldsymbol{y} \rangle = 0$ if $M_0 \neq M_1$, else $\langle \boldsymbol{x}_0 - \boldsymbol{x}_1, \boldsymbol{y} \rangle = 0$.

We introduce the *full-hiding* security for a sNIPE (FH-sNIPE) which is very similar to the predicate privacy notion of an inner product functional encryption (IPFE) in private-key setting [37]. This is the strongest notion of security for a NIPE one can think of as it possesses an additional power where the secret-keys hide the predicate vectors. The challenge bit b is chosen before any query from the adversary's end. The adversary gets $\mathsf{sk}_{\boldsymbol{y}_b^{(j)}}$ during predicate key queries for a pair of vectors $(\boldsymbol{y}_0^{(j)}, \boldsymbol{y}_1^{(j)})$ and receives $\mathsf{CT}_{\boldsymbol{x}_b^{(\iota)}}$ during ciphertext queries for two attribute-message pairs $(\boldsymbol{x}_0^{(\iota)}, M_0), (\boldsymbol{x}_1^{(\iota)}, M_1)$ with a restriction that for all (polynomially bounded integers) j, ι, it holds $\langle \boldsymbol{x}_0^{(j)}, \boldsymbol{y}_0^{(\iota)} \rangle = \langle \boldsymbol{x}_1^{(j)}, \boldsymbol{y}_1^{(\iota)} \rangle = 0$ if $M_0 \neq M_1$ or $\langle \boldsymbol{x}_0^{(j)}, \boldsymbol{y}_0^{(\iota)} \rangle = \langle \boldsymbol{x}_1^{(j)}, \boldsymbol{y}_1^{(\iota)} \rangle$ if $M_0 = M_1$. Full-hiding security demands that given all the queries, no PPT adversary can guess b with a probability significantly greater than 1/2.

1.2 Motivation to our work

In recent years IPEs have emerged with a number of applications in identity-based encryption, polynomial evaluation, disjunctions/conjunctions equality test, proxy-re-encryption [26,11,27] etc. Since non-zero IPE is a *negated* subclass of IPE, the above primitives with negation (such as identity-based revocation, polynomial non-equality and so on) are captured in applications of pNIPEs [4,3]. The

[1] This *full attribute-hiding* notion is adopted from [34] where they have also considered a weak form of attribute-hiding called *weak attribute-hiding*. In weak attribute-hiding the case $M_0 = M_1$ is excluded. However, the case $\langle \boldsymbol{x}_0, \boldsymbol{y} \rangle \neq \langle \boldsymbol{x}_1, \boldsymbol{y} \rangle$ is not considered in this work.

non-zero *inner product encryption* (IPE) was introduced by Katz et al. [26], while the first pNIPE construction was given by Attrapadung and Libert [4]. The security of their pNIPE is based on Decision Linear (DLIN) and the Decision Bilinear Diffie-Hellman (DBDH) assumptions. Despite its involvement in realizing many useful primitives, the security of pNIPEs has not much improved in standard models. Most of the prior works[2][4,33,5,39,16,15] have focused in reducing the size of ciphertexts or secret-keys (or both) of pNIPEs, but they end up with a paring based system that is secure either in co-selective (not adaptive) or selective model. The only direct construction of pNIPE in [34] supporting adaptive security is based on pairing, in particular dual pairing vector spaces (DPVS). One may consider pNIPE as a particular case of the lattice-based ABE schemes (for circuits or branching program) of [9,22,23], but the resulting constructions would be inefficient as we need to rely on either sub-exponential learning with errors (LWE) assumption or branching program. Recently, a direct lattice-based pNIPE is proposed in [25] which is selectively secure and capable of one-bit encryption. In the multi-bit variant of the scheme, sizes of the master public-keys, ciphertexts and secret-keys increase at least linearly with the bit-length of the message. Moreover, their scheme suffers from a complex parameter selection where the noise to modulus ratio is exponentially large in the dimension of attribute vectors and the ciphertext-size is significantly greater than the square of this dimension. Consequently, their pNIPE is not ideal for encrypting a message of large bit-length.

Although the generic construction of [25] delivers adaptively payload-hiding secure pNIPEs via the public-key inner product encryptions (pIPFEs) of [2] in standard models, they are restricted with an *unwanted* condition that an attribute vector when multiplied by a message becomes an eligible candidate for the attribute space of the underlying pIPFE. Due to this fact, the message space and the attribute space cannot be independently chosen, and the message space should be taken as polynomial-size (or logarithmic-size) except for their DCR based scheme. Therefore, it is rear to find an efficient *direct* construction of pNIPE having no such constraint (and without pairing or lattices) in the literature that is capable of encrypting long messages along with adaptive security under well-known assumptions. Moreover, recent developments in achieving attribute-hiding or full-hiding functional encryptions [34,38,28] naturally draw our interest to instantiate similar features also for NIPEs which has not been discussed in previous works (according to the best of our knowledge).

1.3 Our contributions

In this paper, we describe several efficient and practical constructions of NIPE over $\mathbb{Z}$ and $\mathbb{Z}_p$ that are adaptively secure under various assumptions. We propose a payload-hiding pNIPE system adaptively secure under the standard Decisional Diffie-Hellman (DDH) assumption in a cyclic group of prime order. This DDH based pNIPE is based on a pairing free group where the message is independently encrypted rather than embedding it into the attribute vector (of an

[2] We provide detail literature review in the supplementary material SM-1.

pIPFE scheme) as in [25]. As this construction is inspired by the DDH based pIPFE of [2], we require an inherent restriction for the message space to be polynomially bounded so that the original message can be efficiently recovered while decryption.

To overcome this limitation, we utilize a DDH *group with an easy* DL *subgroup* [13]. In such a group, the DDH assumption is modified to a weaker assumption called the DDH-f assumption [14]. We build an adaptively secure PH-pNIPE scheme with inner products over $\mathbb{Z}$ based on the DDH-f assumption. Modifying the key generation and decryption of this construction, we get another PH-pNIPE which is adaptively secure and computes inner products modulo a prime p. Furthermore, we propose more efficient PH-pNIPEs one evaluating inner products over $\mathbb{Z}$ and another over $\mathbb{Z}_p$, both providing adaptive security under the Hard Subgroup Membership (HSM) assumption [14] in a DDH *group with an easy* DL subgroup. We note that the DDH-f is weaker than the both DDH and HSM assumptions [14]. Therefore, our DDH-f based pNIPEs enjoy the most desirable adaptive security under the weakest assumption (known till date). To instantiate a DDH group with an easy DL subgroup one can use class groups of imaginary quadratic fields as shown in [13]. As pNIPEs are known to imply identity-based revocation (IBR) system, our pNIPEs (over $\mathbb{Z}_p$) describe IBR mechanisms without pairing and lattices, that are adaptively secure under standard assumptions like DDH-f and HSM. One noteworthy advantage of our pNIPEs (except the DDH based pNIPE) over the existing schemes is that they can efficiently recover a message irrespective of its size. In comparison with the recent LWE based selectively secure scheme of [25], our pNIPEs are more efficient concerning the sizes of master secret-key, master public-key, predicate secret-keys, ciphertext and in addition, they are adaptively secure (see Table 1 for comparison).

Next, we discuss the attribute-hiding pNIPEs (AH-pNIPE) and full-hiding sNIPEs (FH-sNIPE). We show that a simple modification to the generic construction of [25] yields an AH-pNIPE from a pIPFE with indistinguishability based security (IND-pIPFE) [2] and a FH-sNIPE from a private-key IPFE (sIPFE) with full-hiding security [38]. Making use of IND-pIPFEs of [2,14], our generic approach leads to first pNIPE schemes with attribute-hiding ability based on various standard assumptions such as DDH, DDH-f, HSM, LWE and DCR. Similarly, instantiating our generic construction with efficient FH-sIPFEs of [38,28], we present first sNIPEs having full-hiding feature that are secure under DLIN and Symmetric External Diffie-Hellman (SXDH) assumption. Here, we emphasize that the generic construction of [25] can only provide payload-hiding security whereas our approach has the flexibility to furnish attribute-hiding or full-hiding security for a NIPE system.

1.4 Overview of techniques

We briefly present our techniques below.

PH-pNIPE from the DDH assumption. The idea behind our DDH-based pNIPE and its security proof are taken from the DDH-based pIPFE of [2] which indeed based on the hash proof systems [17]. A cyclic group G of prime order q is employed to construct master public-keys as $\mathsf{MPK} = \{g, h, \{h_i = g^{u_i} h^{v_i}\}_{i=1}^{l}\}$ where g, h are any arbitrary generators of G and the master secret-key is taken as $(\boldsymbol{u}, \boldsymbol{v}) \in \mathbb{Z}_q^l \times \mathbb{Z}_q^l$. For any predicate vector $\boldsymbol{y} \in \mathbb{Z}_q^l$, secret-keys are computed as $\mathsf{sk}_{\boldsymbol{y}} = (\langle \boldsymbol{u}, \boldsymbol{y} \rangle, \langle \boldsymbol{v}, \boldsymbol{y} \rangle) \in \mathbb{Z}^2$ and encryption of a message $M \in \mathbb{Z}_q$ with an attribute $\boldsymbol{x} = (x_1, \cdots, x_l) \in \mathbb{Z}_q^l$ is given by $\mathsf{CT}_{\boldsymbol{x}} = \{g^r, h^r, \mathsf{ct} = g^M h^{rt}, \mathsf{ct}_i = \{h^{rtx_i} h_i^r\}_{i=1}^{l}\}$ where r, t are two arbitrary $\mathbb{Z}_q$-elements. If $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0$, one can compute the randomness $t \in \mathbb{Z}_q$ from $\mathsf{CT}_{\boldsymbol{x}}$ using the secret-key $\mathsf{sk}_{\boldsymbol{y}}$. Recovery of t helps the decrypter to extract g^M form ct and finally the message M is obtained by applying discrete logarithm with base g which is possible in polynomial time if the message lies in a sufficiently bounded range. We note that our ciphertext needs only one extra group element than in [2], which makes it as efficient as the pIPFE of [2]. Encrypting the message follows the classical *ElGamal* technique where the randomness t binds M with the attribute $\boldsymbol{x}$ so that one can get the randomness via the secret-key only if the inner product of predicate and attribute vectors is non-zero. To prove the pNIPE provides adaptively payload-hiding security, we first show that the master public-key and queried secret-keys leave no information about the *binding* randomness t and then use the distribution of t (which is uniform over $\mathbb{Z}_q$) to conclude that the message is statistically hidden from the adversary's view.

While the above pNIPE encrypts a polynomially bounded message, we overcome this constraint using a DDH group of composite order n having an easy DL subgroup of prime order p. More precisely, we construct PH-pNIPEs for inner products over $\mathbb{Z}$ based on each of $\mathsf{DDH\text{-}f}$ and HSM assumption [14]. Then modifying the key-generation and decryption, these pNIPEs are converted to operate for inner products over $\mathbb{Z}_p$.

PH-pNIPE from the $\mathsf{DDH\text{-}f}$ assumption. Our $\mathsf{DDH\text{-}f}$ based pNIPE is a careful combination of *linearly homomorphic encryption scheme* of [13] and $\mathsf{DDH\text{-}f}$ based pIPFE of [14]. As shown in [14], an algebraic structure of class groups of imaginary quadratic fields can be used to generate a cyclic group G of unknown order which contains an easy DL subgroup of known order. In particular, we take a cyclic group G of order $n = p \cdot s$ with an unknown integer s such that $\gcd(p, s) = 1$ and a subgroup F of order p where DL problem can be solved in polynomial time via an algorithm named as Solve [13]. The master public-key of our pNIPE consists of $\{g, f, h, \{h_i = g^{u_i} h^{v_i}\}_{i=1}^{l}\}$ where g, f are the generators of G, F respectively, $h \in G$ is chosen randomly and $(\boldsymbol{u}, \boldsymbol{v}) \in \mathbb{Z}^l \times \mathbb{Z}^l$ forms the master secret-key. A predicate secret-key sk_y for a vector $\boldsymbol{y} \in \mathbb{Z}^l$ is computed as $(\langle \boldsymbol{u}, \boldsymbol{y} \rangle, \langle \boldsymbol{v}, \boldsymbol{y} \rangle) \in \mathbb{Z}^2$. The encryption of a message $M \in \mathbb{Z}$ with an attribute $\boldsymbol{x} = (x_1, \ldots, x_l) \in \mathbb{Z}^l$ is defined as $\{g^r, h^r, \mathsf{ct} = f^M h^{rt}, \{\mathsf{ct}_i = f^{tx_i} h_i^r\}_{i=1}^{l}\}$ for some integers r, t sampled from a uniform distribution over $\mathbb{Z}_n$ and $\mathbb{Z}_p$ respectively. While decryption, we employ Solve to recover t using $\{\mathsf{ct}_i\}_{i=1}^{l}$ and the secret-key $\mathsf{sk}_{\boldsymbol{y}}$. Again applying the Solve algorithm on f^M which was extracted from ct using the recovered randomness t, one gets the original message. We note that t can be recovered only if $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0$ over $\mathbb{Z}$.

The DDH-f problem in G is described as: given g^x, g^y for some randomly chosen x, y from $\mathbb{Z}_n$, it is hard to distinguish between the tuples $\{g^x, g^y, f^a g^{xy}\}$ and $\{g^x, g^y, g^{xy}\}$ where a is uniform modulo p. Since the factor s is unknown, one should wonder how to sample elements from G. It can be realized by a distribution statistically close to the uniform distribution over G if an upper bound for s is known. We show that the pNIPE provides adaptively payload-hiding security if the DDH-f problem is hard in G. The security proof starts with the same strategy what we have followed in our DDH based construction, i.e., if the randomness t is information theoretically hidden knowing master public-key and queried secret-keys then the adversary cannot learn anything about M. We follow the proof technique of [14] (which indeed takes inspiration from [2]) for analysing the entropy loss occurred due to secret-key queries and show that it is still not enough for the adversary to determine t from the challenge ciphertext.

PH-pNIPE from the HSM assumption. We propose another pNIPE adaptively payload-hiding secure under HSM assumption. This also depends on similar kind of DDH group G with an easy DL subgroup F as in our DDH-f based pNIPE. We use the fact that G can be written as the direct product of two subgroups G^p and F where $G^p = \{h^p : h \in G\}$ is a cyclic group of order s. The HSM problem states that it is hard to distinguish an element of G^p in the parent group G. We blend the HSM-CL encryption of [14] with the HSM based pIPFE of the same paper to achieve our HSM based pNIPE. The building techniques and the security proof are analogous to that of our DDH-f based system.

PH-pNIPE for inner products modulo a prime. By construction, our DDH based pNIPE computes inner products modulo a prime and the decryption gets succeeded as long as the message is polynomially bounded. For our DDH-f and HSM based pNIPEs, it is necessary to revise the key-generation phase as in [2,14] so that the schemes remain secure while computing inner products modulo a prime p. We note that secret-keys are computed over $\mathbb{Z}$ whereas the predicate vectors belong to $\mathbb{Z}_p^l$. Suppose a set of l vectors from $\mathbb{Z}_p^l$ is linearly dependent, hence the adversary can get secret-keys for each of these vectors (when we are in the original schemes). But, it may happen that the set forms a basis for $\mathbb{Z}^l$. In such a scenario, the master secret-key is vulnerable to the adversary knowing all secret-key queries. If the scheme discloses secret-keys corresponding to only $(l-1)$ independent vectors over $\mathbb{Z}_p^l$, then the master secrete-key appears uniform in adversary's view. Therefore, the key-generation center is required to maintain a list of predicate vectors so that not more than $(l-1)$ independent predicate vectors are allowed to be queried for secret-keys. In particular, the key-generation is now *stateful*. This modification upgrades our original schemes into PH-pNIPEs for inner products modulo a prime. The pNIPEs are adaptively secure under each of DDH-f and HSM assumptions.

As mentioned in [14], the attack from [12] leaves no effect to our constructions based on class groups of imaginary quadratic fields as the attack is possible for the cryptosystems whose security is based on the factorization of a discriminant, whereas this factorization is public in our schemes. The complexity of best-known algorithms for DL problem in an imaginary class group run in subexponential time[3] of $\mathcal{O}(L_{1/2})$ [6] while the factorization or DL problem in a finite field can be solved in $\mathcal{O}(L_{1/3})$ [1]. Therefore our master secret-keys can be chosen shorter without the security breach of the schemes.

Technical difference with the generic construction of [25]. We note that our technique of encrypting a message along with an attribute is different from the generic construction of [25]. For a message $M \in \mathbb{Z}_q$ and an attribute $\boldsymbol{x} \in \mathbb{Z}_q^l$, the pNIPE of [25] directly encrypts the attribute vector $M \cdot \boldsymbol{x}$ using any pIPFE to produce the ciphertext. Therefore, $M \cdot \boldsymbol{x}$ should be an eligible attribute vector for the underlying pIPFE. As a result, the message space and the attribute space of pNIPE cannot be independently chosen and it is necessary to have the message space contained in the domain of the inner product space for successful decryption. To instantiate the generic pNIPE of [25] using any pIPFE, one can note that this *unwanted* constraint may shorten the message and attribute space of the pNIPE to balance the attribute space of the pIPFE and we may need to apply encryption algorithm for many times to encrypt a long message with large attribute vector by dividing the message into smaller parts. On the other hand, our technique surpasses this limitation by providing a new approach with a cost of one extra group element compared to [25]. The message and the attribute are independently embedded into different group elements which are combined via common randomness.

AH-pNIPEs and FH-sNIPEs from generic construction. We propose a generic construction for NIPEs from IPFEs which provides attribute-hiding security in public-key setting (AH-pNIPE) and full-hiding security in private-key setting (FH-sNIPE). Specifically, we start with the generic construction of [25] which provides only payload-hiding security for pNIPEs. The setup and key generation of our NIPE resemble the underlying IPFE. Apart from computing one IPFE-ciphertext $\mathsf{ct}_{M\cdot\boldsymbol{x}}$ corresponding to a vector $M \cdot \boldsymbol{x} = (Mx_1, \ldots, Mx_l)$ as in [25], our scheme adds another IPFE-ciphertext $\mathsf{ct}_{\boldsymbol{x}}$ for the attribute vector $\boldsymbol{x}$ itself. To recover the message M, one divides the value obtained from the decryption of IPFE for the ciphertext $\mathsf{ct}_{M\cdot\boldsymbol{x}}$ by the decryption of IPFE for the ciphertext $\mathsf{ct}_{\boldsymbol{x}}$ using a secret-key $\mathsf{sk}_{\boldsymbol{y}}$. We note that the decrypter computes $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ without knowing the attribute $\boldsymbol{x}$ in case of AH-pNIPE. Moreover, if the sIPFE is full-hiding then it can evaluate the inner product unaware of the attribute $\boldsymbol{x}$ and the predicate $\boldsymbol{y}$. The decrypter successfully recovers the message only if the inner product is non-zero. We note that the same *unwanted* constraint is used in our AH-pNPEs and FH-pNIPEs. But, it can be observed that one can utilize our techniques employed in PH-pNIPEs to avoid such limitation in our generic constructions but with a similar cost of increasing the ciphertext size.

1.5 Comparing our PH-pNIPEs with existing schemes

In Table 1 we compare our PH-pNIPEs with the existing schemes (excluding the generic constructions of [25] as they obtain PH-pNIPEs from existing pIPFEs by a natural embedding of messages into attribute vectors) in terms of security,

[3] L_α is the abbreviation of $L_{\alpha,c}(x) = \exp((c + o(1))\ \log(x)^\alpha\ \log(\log(x))^{1-\alpha})$

Table 1. Comparison with various PH-pNIPEs where $|S|$ denotes the size of an element from the set S, and $|\mathsf{MPK}|$, $|\mathsf{sk}_{\boldsymbol{y}}|$ and $|\mathsf{CT}|$ indicate the sizes of master public-key, secret-key and ciphertext respectively. Here, C_m denotes a cyclic group of order m, λ is a security parameter, l is the length of predicate/attribute vectors and l_M denotes the bit-length of a message M. Note that p, q are prime numbers and $n = p \cdot s$ with $\gcd(p, s) = 1$. We denote P by a pairing operation and E by a scalar multiplication(resp. exponentiation) in an additive group(resp. multiplicative group).

Reference	Security	Assump.	$\lvert\mathsf{MPK}\rvert$	$\lvert\mathsf{sk}_{\boldsymbol{y}}\rvert$	$\lvert\mathsf{CT}\rvert$	Decryption cost
[4]	co-selective	DLIN +DBDH	$(l+1)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	$(l+6)\lvert\mathbb{G}\rvert$	$9\lvert\mathbb{G}\rvert + 6\lvert\mathbb{G}_T\rvert$	9P+lE
[34]with short CT	adaptive	DLIN	$(8l+23)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	$(4l+5)\lvert\mathbb{G}\rvert$	$13\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	13P+4(l-1)E
[34]with short $\mathsf{sk}_{\boldsymbol{y}}$	adaptive	DLIN	$(8l+23)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	$13\lvert\mathbb{G}\rvert$	$(4l+5)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	13P+4$(l-1)$E
[16]	selective	DBDH	$(l^2+l+1)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	$(l+1)\lvert\mathbb{G}\rvert$	$(l+1)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	2P+E
[15]	selective	l-DBDHE	$(2l+1)\lvert\mathbb{G}\rvert + 2l\lvert\hat{\mathbb{G}}\rvert$	$\lvert\hat{\mathbb{G}}\rvert$	$2\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	2P + (l^2+l+1)E
[25]	selective	LWE	$\tilde{O}((\lambda^2 l + \lambda l_M)\log q)$	$\tilde{O}(\lambda^2 \log q)$	$\tilde{O}((\lambda + l + l_M)\log q)$	$\lambda\lceil\log q\rceil(l + 2l_M)$E
This work Sec. 3	adaptive	DDH	$(l+2)\lvert C_q\rvert$	$\tilde{O}(l(q-1)^2)$	$(l+3)\lvert C_q\rvert$	$(l+3)$E
This work Sec. 4.1	adaptive	DDH-f	$(l+2)\lvert C_n\rvert + \lvert C_p\rvert$	$\tilde{O}(np\sqrt{\lambda l})$	$(l+3)\lvert C_n\rvert$	$(l+3)$E
This work Sec. 5.1	adaptive	HSM	$(l+1)\lvert C_s\rvert + \lvert C_p\rvert$	$\tilde{O}(np\sqrt{\lambda l})$	$(l+2)\lvert C_n\rvert + \lvert C_s\rvert$	$(l+2)$E

hardness assumption, decryption cost and sizes of master public-key (MPK), secret-key($\mathsf{sk}_{\boldsymbol{y}}$), ciphertext($\mathsf{CT}$). To distinguish a pairing based construction with prime order groups, we denote the group by $\mathbb{G}$ for symmetric pairing and represent the pair of groups by $\mathbb{G}, \hat{\mathbb{G}}$ for asymmetric pairing, while $\mathbb{G}_T$ remains as the target group. The cyclic group C_n is a DDH group with an easy DL subgroup C_p and C_n can be written as the direct product of C_p and C_s. As a pairing computation is more expensive than modular exponentiation, our pNIPEs perform faster than the existing pairing based constructions. In case of the LWE based construction of [25], the sizes of MPK and CT significantly increases whenever the bit-length l_M of the message surpasses $\tilde{O}(\lambda)$, whereas these sizes are independent of l_M in our pNIPEs. It can be observed that our pNIPEs not only provide the strongest form of security, the sizes of MPK, $\mathsf{sk}_{\boldsymbol{y}}$ and CT are well comparable with the existing schemes. Moreover, in light of the practical implementation presented in [14] (which uses PARI/GP), we may conclude that our HSM based PH-pNIPE provides better results than the DCR based PH-pNIPE of [25] with respect to secret-key size, encryption time and decryption time.

2 Preliminaries

Notation. For an integer $n \in \mathbb{N}$, the notation $[n]$ represents the set $\{1, \ldots, n\}$. We denote by $x \hookleftarrow \mathcal{D}$ the process of sampling a value x according to the distribution of $\mathcal{D}$. We consider $x \hookleftarrow S$ as the process of random sampling a value x according to the uniform distribution over a finite set S. We assume that the predicate and attribute vectors are of same length l. The inner product between two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^l$ is written as $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{l} x_i y_i = \boldsymbol{x}^T \boldsymbol{y}$. For a vector $\boldsymbol{x} = (x_1, \ldots, x_l) \in \mathbb{Z}^l$, the l_2 norm and the infinity norm of $\boldsymbol{x}$ is defined as $\|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + \cdots + x_l^2}$ and $\|\boldsymbol{x}\|_\infty = \max\{|x_i| : i \in [l]\}$ respectively. For any $\lambda > \lambda_0$, if a non-negative function f satisfies $f(\lambda) < 1/\lambda^c$, c is a constant, then f is called a *negligible* function over the positive integers. We denote λ as a security parameter and negl as a negligible function in λ. The definitions related to IPFE and NIPE are shifted to SM-2.

2.1 Lattices

We recall basic definitions on lattices and collect few results from the previous works that are important for our security proof.

Let $\boldsymbol{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ be a set of n linearly independent vectors from $\mathbb{R}^n$. An n-dimensional lattice Λ generated by $\boldsymbol{B}$ is defined as

$$\Lambda = \mathcal{L}(\boldsymbol{B}) = \{\sum_{i=1}^{n} x_i \boldsymbol{b}_i : x_i \in \mathbb{Z} \text{ for } 1 \leq i \leq n\}.$$

In fact, this is the definition of full-rank lattice used in this paper. The set $\boldsymbol{B}$ is called a basis for the lattice. The determinant of a lattice $\Lambda = \mathcal{L}(\boldsymbol{B})$ is defined as $\det(\Lambda) = \sqrt{\det(\boldsymbol{B}^T \boldsymbol{B})}$. The dual lattice of Λ is defined as $\Lambda^* = \{\boldsymbol{x} \in \mathbb{R}^n : \langle \boldsymbol{x}, \boldsymbol{z} \rangle \in \mathbb{Z}, \forall \boldsymbol{z} \in \Lambda\}$. The minimum distance of a lattice Λ is defined (in the Euclidean l_2 norm) as $\lambda_1(\Lambda) = \min\{\|\boldsymbol{x}\|_2 : \boldsymbol{x} \in \Lambda \setminus \{\boldsymbol{0}\}\}$. More generally, the i-th successive minimum for a lattice Λ can be defined as $\lambda_i(\Lambda) = \min\{r : \dim(\mathrm{span}(\Lambda \cap \overline{\mathsf{B}}(\boldsymbol{0}, r))) \geq i\}$ where $\overline{\mathsf{B}}(\boldsymbol{0}, r) = \{\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x}\|_2 \leq r\}$.

Gaussian measures. Let $\boldsymbol{c} \in \mathbb{R}^n$ and $\sigma > 0$ be any real number. Then the Gaussian function over $\mathbb{R}^n$ with center at $\boldsymbol{c}$ and parameter σ is defined as $\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x} - \boldsymbol{c}\|_2^2/\sigma^2)$, $\forall \boldsymbol{x} \in \mathbb{R}^n$. If $\boldsymbol{c} = \boldsymbol{0}$, then we simply write it as $\rho_\sigma(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x}\|_2^2/\sigma^2)$. For any n-dimensional lattice Λ, the discrete Gaussian distribution over Λ with center $\boldsymbol{c} \in \mathbb{R}^n$ and parameter $\sigma > 0$ is defined as $\mathcal{D}_{\Lambda,\sigma,\boldsymbol{c}}(\boldsymbol{x}) = \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})/\rho_{\sigma,\boldsymbol{c}}(\Lambda)$, $\forall \boldsymbol{x} \in \mathbb{R}^n$ where $\rho_{\sigma,\boldsymbol{c}}(\Lambda) = \sum_{\boldsymbol{x} \in \Lambda} \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})$. For $\epsilon > 0$ the smoothing parameter $\eta_\epsilon(\Lambda)$ of an n-dimensional lattice Λ is defined to be the smallest positive real σ such that $\rho_{1/\sigma}(\Lambda^* \setminus \{\boldsymbol{0}\}) \leq \epsilon$.

Lemma 1 (Hadamard inequality). *Let Λ be a lattice in $\mathbb{R}^n$ and let $\boldsymbol{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ be a basis of Λ. Then we have the inequality* $\det(\Lambda) \leq \prod_{i=1}^{n} \|\boldsymbol{b}_i\|_2$.

Lemma 2. [14] *Let Λ be an n-dimensional lattice and $\boldsymbol{x}(\neq \boldsymbol{0}) \in \Lambda$. Let V be a random variable distributed according to $\mathcal{D}_{\boldsymbol{x}\mathbb{Z},\sigma,\boldsymbol{c}}$ for some $\boldsymbol{c} \in \mathbb{R}^n$ and a real number $\sigma > 0$. Then the random variable S defined as $S = \langle \boldsymbol{x}, V \rangle$ is distributed according to $\mathcal{D}_{\|\boldsymbol{x}\|_2^2\mathbb{Z},\sigma\cdot\|\boldsymbol{x}\|_2,\langle \boldsymbol{c},\boldsymbol{x}\rangle}$.*

Lemma 3. [20] *Let Λ, Λ' be two n-dimensional lattices with $\Lambda' \subset \Lambda$. Then for any $\epsilon \in (0, \frac{1}{2})$, any $\sigma > \eta_\epsilon(\Lambda')$ and any $\boldsymbol{c} \in \mathbb{R}^n$, the distribution $\mathcal{D}_{\Lambda,\sigma,\boldsymbol{c}} \bmod \Lambda'$ is within a statistical distance of at most 2ϵ from the uniform distribution over $\Lambda \bmod \Lambda'$.*

Lemma 4. [31] *Let Λ be an n-dimensional lattice and ϵ be a positive real number. Then* $\eta_\epsilon(\Lambda) \leq \sqrt{\frac{ln(2n(1+1/\epsilon))}{\pi}} \cdot \lambda_n(\Lambda)$.

2.2 Cryptographic Assumptions

Definition 1 (DDH assumption) Let G be a cyclic group with order q which is a λ-bit prime. The *decisional Diffie-Hellman* (DDH) problem is to distinguish the distributions $\{(g^x, g^y, g^{xy}) : x, y \hookleftarrow \mathbb{Z}_q\}$ and $\{(g^x, g^y, g^z) : x, y, z \hookleftarrow \mathbb{Z}_q\}$ where g is an arbitrary element of G. The DDH-assumption is that the DDH problem is hard in G. For any prime $\lambda \in \mathbb{N}$, $g \hookleftarrow G$ and $Z_0 = g^{xy}$, $Z_1 = g^z$ where $x, y, z \hookleftarrow \mathbb{Z}_q$, the advantage of a DDH adversary $\mathcal{A}$ is defined as:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DDH}}(\lambda) = \left|2 \cdot \Pr\left[b = \mathcal{A}(G, g, g^x, g^y, Z_b) : b \hookleftarrow \{0,1\}\right] - 1\right|$$

The DDH problem is said to be hard if for any probabilistic polynomial time adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DDH}}(\lambda)$ is negligible.

Definition 2 (Generator of a DDH group with an easy DL subgroup[14]) Let $\mathsf{GenGroup} = (\mathsf{Gen}, \mathsf{Solve})$ be a pair of algorithms working as follows:

- $(p, \tilde{s}, g, f, g_p, G, F, G^p) \leftarrow \mathsf{Gen}(1^\lambda, 1^\mu)$: On input two security parameters λ, μ, the algorithm Gen outputs a cyclic group $(G, .)$ of order $n = ps$ with two subgroups $G^p = \{x^p : x \in G\}$ of order[4] s and F of order p satisfying $G = F \times G^p$ where p is a μ-bit prime, s is an integer such that $\gcd(p, s) = 1$. The algorithm also outputs the prime p, an upper bound $\tilde{s}$ of s (instead of s) and generators g, f, g_p of the groups G, F, G^p respectively.
- $x' \leftarrow \mathsf{Solve}(p, \tilde{s}, g, f, g_p, G, F, G^p, X)$: The deterministic polynomial time algorithm Solve takes as input $(p, \tilde{s}, g, f, g_p, G, F, G^p)$ and an element $X \in F$, and outputs the discrete logarithm (DL) of X, i.e., an element $x \in \mathbb{Z}_p$ such that $X = f^x$. In other words, for any $\lambda, \mu \in \mathbb{N}, (p, \tilde{s}, g, f, g_p, G, F, G^p) \leftarrow \mathsf{Gen}(1^\lambda, 1^\mu)$ and $x \hookleftarrow \mathbb{Z}_p$ we have $\Pr\left[x = \mathsf{Solve}(p, \tilde{s}, g, f, G, F, f^x)\right] = 1$. Indicating param as $(p, \tilde{s}, g, f, G, F)$, we use the notation $\mathsf{Solve}(\mathsf{param}, X)$ (instead of writing all the parameters) in our constructions.

Definition 3 (DDH-f assumption[14]) Let $\mathsf{GenGroup} = (\mathsf{Gen}, \mathsf{Solve})$ be a generator of a DDH group with an easy DL subgroup as defined in Def. 2. The DDH-f problem is to distinguish the distributions $\{(g^x, g^y, g^{xy}) : x, y \hookleftarrow \mathcal{D}\}$ and $\{(g^x, g^y, g^{xy} f^u) : x, y \hookleftarrow \mathcal{D}, u \hookleftarrow \mathbb{Z}_p\}$ where $\mathcal{D}$ is a distribution over the integers such that the distribution $\{g^x : x \hookleftarrow \mathcal{D}\}$ is at a distance less than $2^{-\lambda}$ from the uniform distribution over G. The DDH-f assumption is that the DDH-f problem is hard in G even with access to Solve algorithm. For any $\lambda, \mu \in \mathbb{N}, (p, \tilde{s}, g, f, g_p, G, F, G^p) \leftarrow \mathsf{Gen}(1^\lambda, 1^\mu)$ and $Z_0 = g^{xy}, Z_1 = g^{xy} f^u$ where $x, y \hookleftarrow \mathcal{D}, u \hookleftarrow \mathbb{Z}_p$, the advantage of an DDH-f adversary $\mathcal{A}$ is defined as:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DDH\text{-}f}}(\lambda, \mu) = \left|2 \cdot \Pr\left[b = \mathcal{A}(\mathsf{param}, g^x, g^y, Z_b, \mathsf{Solve}(.)) : b \hookleftarrow \{0,1\}\right] - 1\right|$$

[4] The order is chosen such that the DL problem in G^p takes exponential time.

The $\mathsf{DDH\text{-}f}$ problem is said to be hard if for any probabilistic polynomial time adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DDH\text{-}f}}(\lambda, \mu)$ is negligible.

Definition 4 (HSM assumption[14]) Let $\mathsf{GenGroup} = (\mathsf{Gen}, \mathsf{Solve})$ be a generator of a DDH group with an easy DL subgroup as defined in Def. 2. The *hard subgroup membership* (HSM) problem is to distinguish the elements of G^p in G. The HSM assumption is that the HSM problem is hard in G even with the access to Solve algorithm. Let $\mathcal{D}$ (resp. $\mathcal{D}_p$) be a distribution over the set of integers such that the distribution $\{g^x : x \hookleftarrow \mathcal{D}\}$ (resp. $\{g_p^x : x \hookleftarrow \mathcal{D}_p\}$) is at a distance less than $2^{-\lambda}$ from the uniform distribution over G (resp. G^p). For any $\lambda, \mu \in \mathbb{N}, (p, \tilde{s}, g, f, g_p, G, F, G^p) \leftarrow \mathsf{Gen}(1^\lambda, 1^\mu)$ and $X_0 = g^x, X_1 = g_p^{x'}$ where $x \hookleftarrow \mathcal{D}, x' \hookleftarrow \mathcal{D}_p$, the advantage of an HSM adversary $\mathcal{A}$ is defined as:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{HSM}}(\lambda, \mu) = \left|2 \cdot \Pr\left[b = \mathcal{A}(\mathsf{param}, X_b, \mathsf{Solve}(.)) : b \hookleftarrow \{0,1\}\right] - 1\right|$$

The HSM problem is said to be hard if for any probabilistic polynomial time adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{HSM}}(\lambda, \mu)$ is negligible.

Theorem 1 (Relation among the above assumptions[14]) *The* $\mathsf{DDH\text{-}f}$ *assumption is weaker than the* DDH *and* HSM *assumptions, i.e., the* DDH *assumption implies the* $\mathsf{DDH\text{-}f}$ *assumption and the* HSM *assumption implies the* $\mathsf{DDH\text{-}f}$ *assumption.*

Sampling from G and G^p. To apply $\mathsf{GenGroup}$ in our constructions it is mandatory to explicitly define the distribution $\mathcal{D}$ and $\mathcal{D}_p$. One natural way to sample an element from a cyclic group G of order n with a known generator g is to first pick an integer x uniform modulo n and then return g^x. But, this method cannot be used to implement $\mathcal{D}$ and $\mathcal{D}_p$, since the orders of G and G^p are not revealed by $\mathsf{GenGroup}$. However, an upper bound $\tilde{s}$ (resp. $p \cdot \tilde{s}$) for the order of G^p (resp. G) is known. We can utilize folded uniform or folded gaussian (for batter efficiency) distributions using those bounds.

Lemma 5. [14] *Consider the output of* $\mathsf{Gen}(1^\lambda, 1^\mu)$ *as described in the Def. 2. The distributions $\mathcal{D}$ and $\mathcal{D}_p$ (used in Def. 3,4) that are at a statistical distance less than $2^{-\lambda}$ from the uniform distributions over G and G^p respectively, can be implemented as follows:*

1. *One can obtain $\mathcal{D}$ as the uniform distribution on $\{0, 1, \ldots, 2^{\lambda-2} \cdot p \cdot \tilde{s}\}$.*
2. *More efficiently for smaller sampling value, $\mathcal{D}$ can be taken as the Gaussian distribution $\mathcal{D}_{\mathbb{Z},\sigma}$ with $\sigma = p \cdot \tilde{s} \cdot \sqrt{\lambda}$.*
3. *Similarly, one can choose $\mathcal{D}_p = \mathcal{D}_{\mathbb{Z},\sigma'}$ with $\sigma' = \tilde{s} \cdot \sqrt{\lambda}$.*
4. *Less efficiently, one can define $\mathcal{D}_p = \mathcal{D}$.*
5. *Since $G = F \times G^p$, one can consider $\mathcal{D}$ as $\{f^a \cdot g_p^x : a \hookleftarrow \mathbb{Z}_p, x \hookleftarrow \mathcal{D}_p\}$ using the uniform distribution over $\mathbb{Z}_p$ and the distribution $\mathcal{D}_p$.*

3 Payload-hiding stateless pNIPE based on DDH assumption

In this section, we describe a stateless pNIPE scheme by modifying the DDH-based pIPFE of [2]. Under the DDH assumption, our pNIPE provides adaptively payload-hiding security. The encryption algorithm of our pNIPE adds one extra group element in the ciphertext of [2]. For decryption, we need to solve a DL problem in polynomial time similar to the case of the DDH-based pIPFE where the inner product was taken within a polynomial boundary. This requires our construction to consider a polynomial size message space.

Let λ be the security parameter. We use a cyclic group G with order q which is a μ-bit prime and $\mu \geq \lambda$. The domains of the pNIPE are: predicate space $\mathcal{P} = \mathbb{Z}_q^l$, attribute space $\mathcal{Q} = \mathbb{Z}_q^l$, inner product space $\mathcal{I} = \mathbb{Z}_q$ and a polynomially bounded message space $\mathcal{M} \subset \mathbb{Z}_q$. Additionally, we assume that $|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| < q$ for any $\boldsymbol{x} \in \mathcal{Q}$ and $\boldsymbol{y} \in \mathcal{P}$.

Construction. Our $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ works as follows:

- $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^l)$: A trusted authority computes a master public-key MPK, a master secret-key MSK using the below steps.
  - Generate a cyclic group G having order a μ-bit prime q with $\mu \geq \lambda$ and consider two generators $g, h \hookleftarrow G$
  - Pick two vectors $\boldsymbol{u}, \boldsymbol{v} \hookleftarrow \mathbb{Z}_q^l$ and write $\boldsymbol{u} = (u_1, \ldots, u_l), \boldsymbol{v} = (v_1, \ldots, v_l)$
  - Compute $h_i = g^{u_i} h^{v_i}$ for $1 \leq i \leq l$
  - Return the keys $\mathsf{MSK} = (\boldsymbol{u}, \boldsymbol{v})$ and $\mathsf{MPK} = (G, g, h, \{h_i\}_{i \in [l]})$
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \boldsymbol{y})$: For a predicate vector $\boldsymbol{y} = (y_1, \ldots, y_l) \in \mathcal{P}$, the authority uses $\mathsf{MSK} = (\boldsymbol{u}, \boldsymbol{v})$ to compute the secret-key $\mathsf{sk}_{\boldsymbol{y}} = (u^{(y)}, v^{(y)})$ where $u^{(y)} = \langle \boldsymbol{u}, \boldsymbol{y} \rangle, v^{(y)} = \langle \boldsymbol{v}, \boldsymbol{y} \rangle \in \mathbb{Z}$.
- $\mathsf{CT}_{\boldsymbol{x}} \leftarrow \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}, M)$: A user encrypts his message $M \in \mathcal{M}$ with an attribute vector $\boldsymbol{x} = (x_1, \ldots, x_l) \in \mathcal{Q}$ using the following steps.
  - Pick two random numbers $r, t \hookleftarrow \mathbb{Z}_q$
  - Set $\mathsf{C} = g^r$ and $\mathsf{D} = h^r$
  - Compute $\mathsf{ct} = g^M \mathsf{D}^t$ and $\mathsf{ct}_i = \mathsf{D}^{t x_i} h_i^r$ for $1 \leq i \leq l$
  - Return the ciphertext $\mathsf{CT}_{\boldsymbol{x}} = (\mathsf{C}, \mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i \in [l]})$
- $\perp$ or $\zeta \leftarrow \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A decrypter first computes $\varrho \leftarrow \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ mod q. If $\varrho = 0$, then it returns $\perp$; otherwise it runs with the steps below.
  - Parse $\mathsf{sk}_{\boldsymbol{y}} = (u^{(y)}, v^{(y)})$ and $\mathsf{CT}_{\boldsymbol{x}} = (\mathsf{C}, \mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i \in [l]})$
  - Compute $\mathsf{R}_t = (\prod_{i=1}^l \mathsf{ct}_i^{y_i}) / \mathsf{C}^{u^{(y)}} \cdot \mathsf{D}^{v^{(y)}}$
  - Set $\mathsf{B} \leftarrow \mathsf{ct} / (\mathsf{R}_t)^{1/\varrho}$
  - Return the message as $\log_g \mathsf{B}$

Correctness. By construction, R_t can be expressed as

$$
\begin{aligned}
(\prod_{i \in [l]} \mathsf{ct}_i^{y_i}) / \mathsf{C}^{u^{(y)}} \cdot \mathsf{D}^{v^{(y)}} &= (\prod_{i \in [l]} \mathsf{D}^{t x_i y_i} \cdot g^{r u_i y_i} \cdot h^{r v_i y_i}) / g^{r u^{(y)}} \cdot h^{r v^{(y)}} \\
&= \mathsf{D}^{t \langle \boldsymbol{x}, \boldsymbol{y} \rangle} \cdot g^{r \langle \boldsymbol{u}, \boldsymbol{y} \rangle} \cdot h^{\langle \boldsymbol{v}, \boldsymbol{y} \rangle} / g^{r \langle \boldsymbol{u}, \boldsymbol{y} \rangle} \cdot h^{\langle \boldsymbol{v}, \boldsymbol{y} \rangle} = \mathsf{D}^{t \langle \boldsymbol{x}, \boldsymbol{y} \rangle}
\end{aligned}
$$

If $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0 \bmod q$, then $(\mathsf{R}_t)^{1/\varrho} = \mathsf{D}^t$ and the value of $\mathsf{B} = \mathsf{ct}/(\mathsf{R}_t)^{1/\varrho}$ becomes g^M where M is the original message. Since, the message space is of polynomial size the last step of decryption computes $M = \log_g \mathsf{B}$ in polynomial time.

Theorem 2 *Assuming the hardness of* DDH *problem in the group G, the above pNIPE for inner products over $\mathbb{Z}_q$ provides adaptively payload-hiding security.* (The proof is available in SM-3.)

4 Payload-hiding pNIPE based on DDH-f assumption

We build two NIPE schemes in public-key setting relying on DDH-f assumption. The first one is for inner products over $\mathbb{Z}$ (Sec. 4.1) and the second one is for inner products over a prime field $\mathbb{Z}_p$ (Sec. 4.2). Our constructions can be treated as a modification to the DDH-f-based pIPFE schemes of [14]. We use a DDH group with an easy DL subgroup of prime order p (Def. 2) and encrypt a message into this subgroup. The pNIPE over $\mathbb{Z}$ is stateless whereas we need to maintain a state storing list of vectors in the key generation phase of the pNIPE over $\mathbb{Z}_p$ to prevent some basic attacks.

4.1 DDH-f based stateless pNIPE for Inner Products over $\mathbb{Z}$

We discard the group G^p and its generator g_p from the output of $\mathsf{Gen}(1^\lambda, 1^\mu)$ which is a part of the algorithm $\mathsf{GenGroup} = (\mathsf{Gen}, \mathsf{Solve})$ in Def. 2. That is, $\mathsf{Gen}(1^\lambda, 1^\mu)$ now returns a tuple $(p, \tilde{s}, g, f, G, F)$ where p is a μ-bit prime with $\mu \geq \lambda$.

The domains associated to our pNIPE are: predicate space $\mathcal{P} = \mathbb{Z}^l$, attribute space $\mathcal{Q} = \mathbb{Z}^l$, inner product space $\mathcal{I} = \mathbb{Z}$ and message space $\mathcal{M} = \mathbb{Z}_p$. Moreover, for any $\boldsymbol{y} \in \mathcal{P}$ and any $\boldsymbol{x} \in \mathcal{Q}$ it holds that $\|\boldsymbol{y}\|_\infty < Y$ and $\|\boldsymbol{x}\|_\infty < X$ with $X, Y < (p/l)^{1/2}$.

Construction. Our $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is working as follows:

- $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^\mu, 1^l)$: A trusted authority computes a master public-key MPK and a master secret-key MSK using the below steps.
  - Generate $(p, \tilde{s}, g, f, G, F) \leftarrow \mathsf{Gen}(1^\lambda, 1^\mu)$ where[5] $G = \langle g \rangle, F = \langle f \rangle, |G| = n = ps, |F| = p$ such that $\gcd(p, s) = 1$ and $\tilde{s}$ is an upper bound of s.
  - Sample $\alpha \hookleftarrow \mathcal{D}_{\mathbb{Z},\sigma}$ and compute $h = g^\alpha$ where σ is a real number to be set as $\sigma > p^{3/2} \cdot \tilde{s} \cdot \sqrt{\lambda}$ (for the security proof of Th. 3)
  - Pick two vectors $\boldsymbol{u}, \boldsymbol{v} \hookleftarrow \mathcal{D}_{\mathbb{Z}^l,\sigma}$ and write $\boldsymbol{u} = (u_1, \ldots, u_l), \boldsymbol{v} = (v_1, \ldots, v_l)$
  - Compute $h_i = g^{u_i} h^{v_i}$ for $1 \leq i \leq l$
  - Return the keys $\mathsf{MSK} = (\boldsymbol{u}, \boldsymbol{v})$ and $\mathsf{MPK} = (\mathsf{param}, h, \{h_i\}_{i \in [l]})$ where $\mathsf{param} = (p, \tilde{s}, g, f, G, F)$
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \boldsymbol{y})$: For a predicate vector $\boldsymbol{y} = (y_1, \ldots, y_l) \in \mathcal{P}$, the authority returns the secret-key $\mathsf{sk}_{\boldsymbol{y}} = (u^{(y)}, v^{(y)})$ where $u^{(y)} = \langle \boldsymbol{u}, \boldsymbol{y} \rangle, v^{(y)} = \langle \boldsymbol{v}, \boldsymbol{y} \rangle$ are computed over $\mathbb{Z}$.

[5] We note that s is unknown and the algorithm only outputs an upper bound of s.

- $\mathsf{CT}_{\boldsymbol{x}} \leftarrow \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}, M)$: A user encrypts his message $M \in \mathcal{M}$ with an attribute vector $\boldsymbol{x} = (x_1, \ldots, x_l) \in \mathcal{Q}$ utilizing the following steps.
  - Pick two random numbers $r \hookleftarrow \mathcal{D}_{\mathbb{Z},\sigma}, t \hookleftarrow \mathbb{Z}_p$
  - Set $\mathsf{C} = g^r$ and $\mathsf{D} = h^r$, both belong to G
  - Compute $\mathsf{ct} = f^M \mathsf{D}^t$ and $\mathsf{ct}_i = f^{tx_i} h_i^r$ for i runs from 1 to l
  - Return the ciphertext $\mathsf{CT}_{\boldsymbol{x}} = \{\mathsf{C}, \mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i \in [l]}\}$
- $\perp$ or $\zeta \leftarrow \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A decrypter computes $\varrho \leftarrow \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ and returns $\perp$ if $\varrho = 0$; otherwise it executes the steps below.
  - Parse $\mathsf{sk}_{\boldsymbol{y}} = (u^{(y)}, v^{(y)})$ and $\mathsf{CT}_{\boldsymbol{x}} = \{\mathsf{C}, \mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i \in [l]}\}$
  - Compute $\mathsf{R}_t = (\prod_{i=1}^{l} \mathsf{ct}_i^{y_i}) / \mathsf{C}^{u^{(y)}} \cdot \mathsf{D}^{v^{(y)}}$
  - Run the algorithm Solve to get an output $\eta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{R}_t)$
  - Compute $t \leftarrow \eta \cdot \varrho^{-1}$ over $\mathbb{Z}_p$ and then $\mathsf{ct}' \leftarrow \mathsf{ct}/\mathsf{D}^t$
  - Get an element $\zeta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{ct}')$ of $\mathbb{Z}_p$
  - Return the message as ζ

Correctness. Suppose $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0$. We show that $\mathsf{pNIPE.Dec}$ returns the original message with all but a negligible probability. By the construction, R_t can be written as

$$\begin{aligned}(\prod_{i \in [l]} \mathsf{ct}_i^{y_i}) / \mathsf{C}^{u^{(y)}} \cdot \mathsf{D}^{v^{(y)}} &= (\prod_{i \in [l]} f^{tx_i y_i} \cdot g^{ru_i y_i} \cdot h^{rv_i y_i}) / g^{ru^{(y)}} \cdot h^{rv^{(y)}} \\ &= f^{t\langle \boldsymbol{x}, \boldsymbol{y} \rangle} \cdot g^{r\langle \boldsymbol{u}, \boldsymbol{y} \rangle} \cdot h^{\langle \boldsymbol{v}, \boldsymbol{y} \rangle} / g^{r\langle \boldsymbol{u}, \boldsymbol{y} \rangle} \cdot h^{\langle \boldsymbol{v}, \boldsymbol{y} \rangle} = f^{t\langle \boldsymbol{x}, \boldsymbol{y} \rangle}\end{aligned}$$

This means $\mathsf{Solve}(\mathsf{param}, \mathsf{R}_t)$ returns $\eta = t\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ modulo p. As $\|\boldsymbol{x}\|_\infty < X$ and $\|\boldsymbol{y}\|_\infty < Y$ with $X, Y < (p/l)^{1/2}$ we have $|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| < l \cdot X \cdot Y < p$. Thus, $\varrho\ (= \langle \boldsymbol{x}, \boldsymbol{y} \rangle)$ is invertible modulo p. Since, t is uniformly chosen from $\mathbb{Z}_p$, $\eta \neq 0$ happens with all but a negligible probability of $2^{-\mu}$ as p is a μ-bit prime with $\mu \geq \lambda$. Thus, t is recovered by computing $\eta \cdot \varrho^{-1}$ modulo p. Again by the construction, we get $\mathsf{ct}' = \mathsf{ct}/\mathsf{D}^t = f^M$. Finally, applying Solve algorithm to ct' we obtain an integer $\zeta = M$ modulo p. As $M \in \mathbb{Z}_p$, the value of ζ is in fact equals to M. This proves correctness of the pNIPE described above.

Theorem 3 *Assuming the hardness of* DDH-f *problem in the group G, the above pNIPE for inner products over $\mathbb{Z}$ provides adaptively payload-hiding security.*

Proof. We follow the proof technique of [14] adapted into our setting. It starts with a sequence of games and the view of any PPT adversary $\mathcal{A}$ is shown to be indistinguishable in any of the consecutive games. Finally, we end up in a game that statistically hides the challenge bit as required. As usual, Game 0 is the standard payload-hiding security experiment (Def. 10) for the above pNIPE scheme and Game 1, 2 are formalized for our proof to work. All the games are defined in Fig. 1. Since $\mathcal{A}$ is a legitimate adversary, it holds that $\langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0$ for all $\boldsymbol{y} \in \mathcal{P}$ queried by $\mathcal{A}$ where $\boldsymbol{x}^* = (x_1, \ldots, x_l)$ is the challenge attribute. Let E_j be the event that $b = b'$ in Game j for $j = 0, 1, 2$.

Game 0 $\Rightarrow$ Game 1: In Game 1, the challenger directly uses $(\boldsymbol{u}, \boldsymbol{v})$ to compute the challenge ciphertext $\mathsf{CT}_{\boldsymbol{x}^*}^{(b)}$ (see Fig. 1, item 3.4 with $j = 1$) as

$$(\mathsf{C} = g^r, \mathsf{D} = h^r, \mathsf{ct} = f^{M_b} \mathsf{D}^t, \{\mathsf{ct}_i = f^{tx_i} g^{ru_i} h^{rv_i}\}_{i=1}^{l}).$$

```
Game $\boldsymbol{j}$, $j \in \{0,1,2\}$

1. The challenger gets $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^\mu, 1^l)$ where $\mathsf{MSK} = (\boldsymbol{u} = (u_1, \ldots, u_l),\ \boldsymbol{v} = (v_1, \ldots, v_l))$ and $\mathsf{MPK} = (p, \tilde{s}, g, f, h = g^\alpha, G, F, \{h_i = g^{u_i} h^{v_i}\}_{i \in [l]})$
2. The adversary choose $(\boldsymbol{x}^*, M_0, M_1) \leftarrow \mathcal{A}^{\mathsf{pNIPE.KeyGen}(\mathsf{MPK},\mathsf{MSK},\cdot)}(\mathsf{MPK})$
3. The challenger picks a random bit $b$ and encrypts the message $M_b$ with
   the challenge attribute $\boldsymbol{x}^* = (x_1, \ldots, x_l)$ as:
   3.1 If $j = 0,1,2,$          pick $r \hookleftarrow \mathcal{D}_{\mathbb{Z},\sigma}$ and $t \hookleftarrow \mathbb{Z}_p$
   3.2 If $j = 0,1,$            compute $\mathsf{C} = g^r$ and $\mathsf{D} = h^r$
       Else if $j = 2,$         compute $\mathsf{C} = g^r$ and $\mathsf{D} = f^a h^r$ for $a \hookleftarrow \mathbb{Z}_p$
   3.3 If $j = 0,1,2,$          set $\mathsf{ct} = f^{M_b} \mathsf{D}^t$
   3.4 If $j = 0,$              compute $\mathsf{ct}_i = f^{tx_i} h_i^r$ for $1 \le i \le l$
       Else if $j = 1,2,$       compute $\mathsf{ct}_i = f^{tx_i} \mathsf{C}^{u_i} \mathsf{D}^{v_i}$ for $1 \le i \le l$
   3.5 Return $\mathsf{CT}_{\boldsymbol{x}^*}^{(b)} = (\mathsf{G}, \mathsf{H}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i=1}^l)$
4. Finally, the adversary outputs $b' \leftarrow \mathcal{A}^{\mathsf{pNIPE.KeyGen}(\mathsf{MPK},\mathsf{MSK},\cdot)}(\mathsf{CT}_{\boldsymbol{x}^*}^{(b)})$
```

Fig. 1: Sequence of Games used in the proof of Th. 3

So, both these games are identical and we have $\Pr[E_0] = \Pr[E_1]$.

Game 1 $\Rightarrow$ Game 2: The only change occurs in Game 2 is that the term D which is now computed as $f^a h^r$ for some a uniformly picked from $\mathbb{Z}_p$ (see Fig. 1, item 3.2 with $j = 2$). Since $\alpha, r \hookleftarrow \mathcal{D}_{\mathbb{Z},\sigma}$ with $\sigma > p^{3/2} \cdot \tilde{s} \cdot \sqrt{\lambda} > p \cdot \tilde{s} \cdot \sqrt{\lambda}$, by item 2 of Lem. 5 (in Sec 2.2), $h = g^\alpha$ and $\mathsf{C} = g^r$ are both distributed uniformly over G with all but a negligible statistical distance of $2^{-\lambda}$. Therefore in Game 1, $(h = g^\alpha, \mathsf{C} = g^r, \mathsf{D} = g^{\alpha r})$ forms a DH-tuple. Again in Game 2, the same tuple changes to $(h = g^\alpha, \mathsf{C} = g^r, \mathsf{D} = f^a g^{\alpha r})$ where $a \hookleftarrow \mathbb{Z}_p$. The distribution of $(h, \mathsf{C}, \mathsf{D})$ in both of these games are statistically indistinguishable under the $\mathsf{DDH\text{-}f}$ assumption in G and we write $|\Pr[E_1] - \Pr[E_2]| \le \mathsf{Adv}_{\mathcal{B}}^{\mathsf{DDH\text{-}f}}(\lambda, \mu)$ for any PPT adversary $\mathcal{B}$.

Now, we prove that in Game 2, the ciphertext distribution statistically hides the challenge bit b and $|\Pr[E_2] - \frac{1}{2}|$ is negligible in λ. The ciphertext can be expressed as

$$(\mathsf{C} = g^r, \mathsf{D} = f^a h^r, \mathsf{ct} = f^{M_b + at} h^{rt}, \{\mathsf{ct}_i = f^{tx_i + av_i} h_i^r\}_{i=1}^l)$$

The terms C and ct_i information theoretically reveal r modulo n and $z_{t,i} = t \cdot x_i + a \cdot v_i \bmod p$ for all $i \in [l]$, as $\{h_i\}_{i=1}^l$ is a part of the master public-key. If we show that $\boldsymbol{z}_t = t \cdot \boldsymbol{x}^* + a \cdot \boldsymbol{v} \bmod p$ statistically hides $t \bmod p$ from the $\mathcal{A}$'s view (given all predicate key queries), then even if the unbounded adversary can infer $M_b + a \cdot t \bmod p$ from the term ct, the challenge bit b remains statistically hidden since a, t are both uniformly and independently sampled from $\mathbb{Z}_p$ and $a, t \neq 0 \bmod p$ with all but a negligible probability as p is a μ-bit prime with $\mu \ge \lambda$. In particular, we prove that $\langle \boldsymbol{x}^*, \boldsymbol{z}_t \rangle = t \cdot \|\boldsymbol{x}^*\|_2^2 + a \cdot \langle \boldsymbol{x}^*, \boldsymbol{v} \rangle \bmod p$ does not leave sufficient information about $t \bmod p$ to the adversary knowing all predicate key queries made by him. Note that, if $\boldsymbol{x}^* = \boldsymbol{0}$ then $\boldsymbol{z}_t$ is independent of t. We assume that $\boldsymbol{x}^*$ is a non-zero attribute vector.

The predicate vector $\boldsymbol{y}_i = (y_1, \ldots, y_l)$ on the i-th query of $\mathcal{A}$ must satisfy $\langle \boldsymbol{x}^*, \boldsymbol{y}_i \rangle = 0$ where $\boldsymbol{x}^* = (x_1, \ldots, x_l)$ is the challenge attribute. More generally, all queries $\boldsymbol{y}_i$ must belong to the $(l-1)$-dimensional lattice $\boldsymbol{x}^{*\perp} = \{\boldsymbol{y} \in \mathbb{Z}^l :$

$\langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0\}$. Without loss of generality we assume that[6] the first n_0 co-ordinates of $\boldsymbol{x}^*$ are zero and $\gcd(x_{n_0+1}, \ldots, x_l) = 1$. Consider the following matrix

$$\mathbf{Y}_{\mathsf{top}} = \left[\begin{array}{c|ccccc} I_{n_0} & & & & & \\ \hline & -x_{n_0+2} & x_{n_0+1} & & & \\ & & -x_{n_0+3} & x_{n_0+2} & & \\ & & & \ddots & \ddots & \\ & & & & -x_l & x_{l-1} \end{array}\right] \in \mathbb{Z}^{(l-1)\times l}$$

and observe that rows of it form a basis for the lattice $\boldsymbol{x}^{*\perp}$. Let $\mathbf{Y}_{\mathsf{bot}} = \boldsymbol{x}^{*T} \in \mathbb{Z}^{1\times l}$ and take the matrix $\mathbf{Y} = \left[\frac{\mathbf{Y}_{\mathsf{top}}}{\mathbf{Y}_{\mathsf{bot}}}\right] \in \mathbb{Z}^{l\times l}$. It can be shown that $\mathbf{Y}$ is invertible over $\mathbb{Z}_p$. In particular, we observe that

$$\mathbf{Y} \cdot \mathbf{Y}^T = \left[\begin{array}{c|cccc|c} I_{n_0} & & & & & \\ \hline & x_{n_0+1}^2 + x_{n_0+2}^2 & -x_{n_0+1} \cdot x_{n_0+3} & & & \\ & -x_{n_0+1} \cdot x_{n_0+3} & x_{n_0+2}^2 + x_{n_0+3}^2 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & -x_{l-2} \cdot x_l & x_{l-1}^2 + x_l^2 & \\ \hline & & & & & \|\boldsymbol{x}^*\|_2^2 \end{array}\right]$$

and its determinant is given by

$$\det(\mathbf{Y} \cdot \mathbf{Y}^T) = (\prod_{i=n_0+2}^{l-1} x_i^2) \cdot \|\boldsymbol{x}^*\|_2^4.$$

By construction, for all $i = n_0 + 2, \ldots, (l-1)$ it holds that $0 < x_i < p$ as $\|\boldsymbol{x}^*\|_\infty < X < (p/l)^{1/2} < p$ and hence $\prod_{i=n_0+2}^{l-1} x_i^2$ is non-zero modulo p. On the other hand, $0 < \|\boldsymbol{x}^*\|_2 < X \cdot \sqrt{l} < p$ and this implies $\|\boldsymbol{x}^*\|_2^4 \neq 0 \bmod p$. Combining, we have $\det(\mathbf{Y})^2 = \det(\mathbf{Y} \cdot \mathbf{Y}^T)$ is non-zero modulo p. Therefore, $\mathbf{Y}$ is invertible modulo p as $\det(\mathbf{Y}) \neq 0 \bmod p$.

Since $\mathbf{Y}$ is independent of t, it is sufficient to show that $\mathbf{Y} \cdot \boldsymbol{z}_t \bmod p$ statistically hides $t \bmod p$. By construction, each row of $\mathbf{Y}_{\mathsf{top}}$ belongs to $\boldsymbol{x}^{*\perp}$ (i.e. $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{x}^* = 0$) which implies that $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{z}_t$ is independent of t. Therefore, our concern is the last row of $\mathbf{Y} \cdot \boldsymbol{z}_t$, given by $\langle \boldsymbol{x}^*, \boldsymbol{z}_t \rangle = t \cdot \|\boldsymbol{x}^*\|_2^2 + a \cdot \langle \boldsymbol{x}^*, \boldsymbol{v} \rangle \bmod p$.

Next, we show that from $\mathcal{A}$'s view the distribution of $\langle \boldsymbol{x}^*, \boldsymbol{v} \rangle \bmod p$ is close to the uniform distribution modulo p. Eventually, this will imply that the term $\langle \boldsymbol{x}^*, \boldsymbol{z}_t \rangle \bmod p$ statistically hides $t \bmod p$ since $a \hookleftarrow \mathbb{Z}_p$ and $a \neq 0 \bmod p$ with all but a negligible probability as p is a μ-bit prime with $\mu \geq \lambda$.

From the public-key $h_i = g^{u_i} h^{v_i} = g^{u_i + \alpha v_i}$, $i \in [l]$, $\mathcal{A}$ information theoretically learns $\boldsymbol{s} = \boldsymbol{u} + \alpha \cdot \boldsymbol{v} \bmod n$. Knowing $\boldsymbol{s}$ the adversary sees the join distribution of $(\boldsymbol{u}, \boldsymbol{v})$ as $(\boldsymbol{s} - \alpha \cdot \boldsymbol{v} \bmod n, \boldsymbol{v} \bmod n)$ where $\alpha \hookleftarrow \mathcal{D}_{\mathbb{Z},\sigma}, \boldsymbol{v} \hookleftarrow \mathcal{D}_{\mathbb{Z}^l,\sigma}$. Since $\sigma > p^{3/2} \cdot \tilde{s} \cdot \sqrt{\lambda} > p \cdot \tilde{s} \cdot \sqrt{\lambda}$, by item 2 of Lem. 5, the distribution of $\boldsymbol{v} \bmod n$ given $\boldsymbol{s} \bmod n$ remains statistically close to uniform distribution over $\mathbb{Z}^l$.

As the rows of $\mathbf{Y}_{\mathsf{top}}$ form a basis of $\boldsymbol{x}^{*\perp}$, any queried predicate vector $\boldsymbol{y}_i$ can be written as a linear combination of these rows, that is, $\boldsymbol{y}_i = \sum_{j=1}^{l-1} k_{i,j} \boldsymbol{R}_j$

[6] We can always divide $\boldsymbol{x}^*$ by any common divisor of its co-ordinates as it will not change the lattice $\boldsymbol{x}^{*\perp}$.

where $k_{i,j} \in \mathbb{Z}$ and $\boldsymbol{R}_j$ is the j-th row of $\mathbf{Y}_{\mathsf{top}}$. Hence, the secret-key $\mathsf{sk}_{\boldsymbol{y}_i}$ can be expressed as

$$(\langle \boldsymbol{u}, \boldsymbol{y}_i \rangle, \langle \boldsymbol{u}, \boldsymbol{y}_i \rangle) = (\sum_{j=1}^{l-1} k_{i,j} \cdot \boldsymbol{R}_j \cdot \boldsymbol{u}, \sum_{j=1}^{l-1} k_{i,j} \cdot \boldsymbol{R}_j \cdot \boldsymbol{v}).$$

This implies the information learned by $\mathcal{A}$ from the predicate key queries can be completely determined by $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{u}$, $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v} \in \mathbb{Z}^{l-1}$. Again, knowing $\boldsymbol{s} \bmod n$ the distribution of $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{u} \bmod n$ can be seen as

$$\begin{aligned} \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{u} &= \mathbf{Y}_{\mathsf{top}} \cdot (\boldsymbol{s} - \alpha \cdot \boldsymbol{v}) && \bmod n \\ &= \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{s} - \alpha \cdot \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v} && \bmod n \end{aligned}$$

We note that α is independent of $\boldsymbol{v} \bmod n$ and the vectors $\boldsymbol{u}, \boldsymbol{v}$ are sampled independently from $\mathbb{Z}^l$. Thus, the adversary $\mathcal{A}$ cannot achieve more information about $\boldsymbol{v} \bmod n$ from $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{u}$ than what he obtains from $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$. We can ignore $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{u}$ and analyze the distribution of $\boldsymbol{v} \bmod n$ knowing $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$.

Let $\boldsymbol{v}_0 \in \mathbb{Z}^l$ be an arbitrary vector such that $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}_0 = \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$. Given $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$, the distribution of $\boldsymbol{v} \in \mathbb{Z}^l$ is $\boldsymbol{v}_0 + \mathcal{D}_{\Lambda, \sigma, -\boldsymbol{v}_0}$ where $\Lambda = \{\boldsymbol{v} \in \mathbb{Z}^l : \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v} = \mathbf{0}\}$. More precisely, form the adversary's view the master secret-key component $\boldsymbol{v}$ appears as $\boldsymbol{v}_0 + V$ where Now observe that Λ is an 1-dimensional lattice as the rows of $\mathbf{Y}_{\mathsf{top}}$ are linearly independent over $\mathbb{Z}$. Since $\boldsymbol{x}^* \cdot \mathbb{Z} \subset \Lambda$ and $\gcd(x_{n_0+1}, \ldots, x_l) = 1$, it holds that $\Lambda = \boldsymbol{x}^* \cdot \mathbb{Z}$. Thus $\boldsymbol{x}^* (\neq 0) \in \Lambda = \boldsymbol{x}^* \cdot \mathbb{Z}$ and V is a random variable distributed according to $\mathcal{D}_{\Lambda, \sigma, -\boldsymbol{v}_0}$. Applying Lem. 2 of Sec. 2.1, the distribution of $\langle \boldsymbol{x}^*, \boldsymbol{v} \rangle$ knowing $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$ is given by $\langle \boldsymbol{x}^*, \boldsymbol{v}_0 \rangle + \mathcal{D}_{\|\boldsymbol{x}^*\|_2^2 \mathbb{Z}, \sigma \cdot \|\boldsymbol{x}^*\|_2, -\langle \boldsymbol{x}^*, \boldsymbol{v}_0 \rangle}$.

Let us consider the lattice $\Lambda_0' = n \cdot \Lambda_0$ where $\Lambda_0 = \|\boldsymbol{x}\|_2^2 \cdot \mathbb{Z}$. The distribution $\mathcal{D}_{\Lambda_0, \sigma \cdot \|\boldsymbol{x}^*\|_2, -\langle \boldsymbol{x}^*, \boldsymbol{v}_0 \rangle} \bmod \Lambda_0'$, by Lem. 3 of Sec. 2.1, is statistically 2ϵ-close to the uniform distribution over $\Lambda_0 / \Lambda_0' \simeq \mathbb{Z}_n$ if $\eta_\epsilon(\Lambda_0') < \sigma \cdot \|\boldsymbol{x}^*\|_2$. From Lem. 4 (of Sec. 2.1) with $\epsilon = 2^{-\lambda-1}$ we have the following bound

$$\eta_\epsilon(\Lambda_0') \leq \sqrt{\tfrac{\ln(2n(1+1/\epsilon))}{\pi}} \cdot \lambda_1(\Lambda_0') \leq \sqrt{\lambda} \cdot \lambda_1(\Lambda_0') = \sqrt{\lambda} \cdot n \cdot \|\boldsymbol{x}^*\|_2^2.$$

We set $\sigma \cdot \|\boldsymbol{x}^*\|_2 > \sqrt{\lambda} \cdot n \cdot \|\boldsymbol{x}^*\|_2^2$ which implies $\sigma > \sqrt{\lambda} \cdot n \cdot \|\boldsymbol{x}^*\|_2$. By construction $\|\boldsymbol{x}^*\|_2 < X \cdot \sqrt{l} < \sqrt{p}$ and it recommends to choose $\sigma > p^{3/2} \cdot \tilde{s} \cdot \sqrt{\lambda}$ which ensures that the distribution $\langle \boldsymbol{x}^*, \boldsymbol{v} \rangle \bmod n$ is within $2^{-\lambda}$ distance from the uniform distribution over $\mathbb{Z}_n$. Since $n = p \cdot s$ with p, s are co-primes, $\langle \boldsymbol{x}^*, \boldsymbol{v} \rangle \bmod p$ is also distributed close to the uniform distribution over $\mathbb{Z}_p$. Hence, $\boldsymbol{z}_t$ statistically hides $t \bmod p$ and it holds that $|\Pr[E_2] - \frac{1}{2}| \leq 2^{-\lambda}$. Finally, combining all the indistinguishability gaps between the games and using triangular inequality we have $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{PH-pNIPE}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{DDH\text{-}f}}(\lambda, \mu) + 2^{-\lambda}$ which is negligible in λ if the DDH-f assumption holds in G.

4.2 DDH-f based stateful pNIPE for Inner Products over $\mathbb{Z}_p$

Similar to our construction in Sec. 4.1, here also we consider $(p, \tilde{s}, g, f, G, F)$ as the output of $\mathsf{Gen}(1^\lambda, 1^\mu)$ which is a part of $\mathsf{GenGroup}$ algorithms from Def. 2. The prime p is of μ-bit satisfying $\mu \geq \lambda$.

The domains of our pNIPE scheme are taken as predicate space $\mathcal{P} = \mathbb{Z}_p^l$, attribute space $\mathcal{Q} = \mathbb{Z}_p^l$, inner product space $\mathcal{I} = \mathbb{Z}_p$ and message space $\mathcal{M} = \mathbb{Z}_p$. This pNIPE is stateful where the authority is required to maintain a state. The proof of correctness is similar to the previous $\mathsf{DDH\text{-}f}$ construction.

Construction. We describe the $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ where $\mathsf{pNIPE.Setup}$ and $\mathsf{pNIPE.Enc}$ are exactly the same as in the construction of Sec. 4.1 except the parameter σ (used in the distribution $\mathcal{D}_{\mathbb{Z},\sigma}$ or $\mathcal{D}_{\mathbb{Z}^l,\sigma}$) is taken to be greater than $\tilde{s}\cdot\sqrt{\lambda}\cdot p^l\cdot(\sqrt{l})^{l-1}$. Initially the state st in the output of $\mathsf{pNIPE.Setup}$ (for this construction) is empty. The algorithms $\mathsf{pNIPE.KeyGen}$ and $\mathsf{pNIPE.Dec}$ are as follows:

- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$: Let $\boldsymbol{y} \in \mathcal{P}$ be the j-th query to this algorithm. At any stage the internal state st contains at most l tuples of the form $(\boldsymbol{y}_i, \mathsf{sk}_{\boldsymbol{y}_i})$ where $\mathsf{sk}_{\boldsymbol{y}_i}$ is the predicate secret-key corresponding to the predicate vector $\boldsymbol{y}_i$. W.l.o.g. we assume that $j \leq l$ and the state st contains $j-1$ predicate vectors $\{\boldsymbol{y}_i\}_{i=1}^{j-1}$. The authority computes the secret-key $\mathsf{sk}_{\boldsymbol{y}}$ utilizing the below steps.
  - If $\boldsymbol{y}$ is linearly independent of all predicate vectors $\{\boldsymbol{y}_i\}_{i=1}^{j-1} \bmod p$ (present in st) then
    1. Set $\overline{\mathbf{y}} = \boldsymbol{y} \bmod p$ so that $\overline{\mathbf{y}} \in \{0, 1, \ldots, p-1\}^l$
    2. Compute $s_{\overline{\mathbf{y}}} = (u^{(\overline{\mathbf{y}})}, v^{(\overline{\mathbf{y}})}) \in \mathbb{Z}^2$ where $u^{(\overline{\mathbf{y}})} = \langle \boldsymbol{u}, \overline{\mathbf{y}}\rangle$ and $v^{(\overline{\mathbf{y}})} = \langle \boldsymbol{v}, \overline{\mathbf{y}}\rangle$.
    3. Update the state $\mathsf{st} \leftarrow (\mathsf{st}, (\boldsymbol{y}, \mathsf{sk}_{\boldsymbol{y}} = (\overline{\mathbf{y}}, s_{\overline{\mathbf{y}}})))$
  - If there exist integers $k_1, \ldots, k_{j-1}$ such that $\boldsymbol{y} = \sum_{i=1}^{j-1} k_i \boldsymbol{y}_i \in \mathbb{Z}_p^l$ then
    1. Set $\overline{\mathbf{y}} = \sum_{i=1}^{j-1} k_i \overline{\mathbf{y}}_i \in \mathbb{Z}^l$
    2. Compute $s_{\overline{\mathbf{y}}} = (u^{(\overline{\mathbf{y}})}, v^{(\overline{\mathbf{y}})}) \in \mathbb{Z}^2$ where $u^{(\overline{\mathbf{y}})} = \sum_{i=1}^{j-1} k_i u^{(\overline{\mathbf{y}}_i)}$ and $v^{(\overline{\mathbf{y}})} = \sum_{i=1}^{j-1} k_i v^{(\overline{\mathbf{y}}_i)}$
  - Return the secret-key as $\mathsf{sk}_{\boldsymbol{y}} = (\overline{\mathbf{y}}, s_{\overline{\mathbf{y}}})$
- $\perp$ or $\zeta \leftarrow \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A decrypter first sets $\varrho \leftarrow \langle \boldsymbol{x}, \boldsymbol{y}\rangle$. If $\varrho \equiv 0 \pmod p$, then it returns $\perp$; otherwise it executes the steps below.
  - Parse the secret-key $\mathsf{sk}_{\boldsymbol{y}} = (\overline{\mathbf{y}} = (\overline{y}_1, \ldots, \overline{y}_l), s_{\overline{\mathbf{y}}} = (u^{(\overline{\mathbf{y}})}, v^{(\overline{\mathbf{y}})}))$ and the ciphertext $\mathsf{CT}_{\boldsymbol{x}} = \{\mathsf{C}, \mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i\in[l]}\}$
  - Compute $\mathsf{R}_t = (\prod_{i=1}^{l} \mathsf{ct}_i^{\overline{y}_i})/\mathsf{C}^{u^{(\overline{\mathbf{y}})}} \cdot \mathsf{D}^{v^{(\overline{\mathbf{y}})}}$
  - Run the algorithm Solve to obtain $\eta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{R}_t)$
  - Compute $t \leftarrow \eta \cdot \varrho^{-1} \bmod p$ and then $\mathsf{ct}' \leftarrow \mathsf{ct}/\mathsf{D}^t$
  - Get an element $\zeta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{ct}')$ of $\mathbb{Z}_p$
  - Return the message as ζ

Theorem 4 *Assuming the hardness of* $\mathsf{DDH\text{-}f}$ *problem in the group G, the above pNIPE for inner products over $\mathbb{Z}_p$ provides adaptively payload-hiding security.*

Proof. The main structural difference between the pNIPE in the previous subsection and the pNIPE described above is that the key extraction process becomes stateful. Since the encryption technique remains unchanged, one starts with the same sequence of games as explained in Fig. 1, but the key generation oracle is converted into a stateful one which is now denoted as $\mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK},$

$\mathsf{st}, \cdot)$. For any predicate vector $\boldsymbol{y}$ queried by $\mathcal{A}$, it holds that $\langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0 \bmod p$ where $\boldsymbol{x}^* = (x_1, \ldots, x_l) \in \mathbb{Z}_p^l$ is the challenge attribute. We rename Game j and event E_j of Th. 3 into Game j' and event E'_j respectively in this proof. Consequently, one can adopt similar explanations from Th. 3 and claim that

$$\Pr[E'_0] = \Pr[E'_1] \text{ and } |\Pr[E'_1] - \Pr[E'_2]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{DDH\text{-}f}}(\lambda, \mu)$$

To complete the proof, we need to show that the challenge bit is statistically hidden in the ciphertext distribution of Game $2'$. According to the discussion in Game 2 of Th. 3, the challenge ciphertext information theoretically reveals

$$M_b + a \cdot t \bmod p \text{ and } \boldsymbol{z}_t = t \cdot \boldsymbol{x}^* + a \cdot \boldsymbol{v} \bmod p$$

and the challenge bit b is unpredictable if $\boldsymbol{z}_t$ statistically hides $t \bmod p$ from the adversary's view as a, t are uniformly sampled from $\mathbb{Z}_p$ where p is a μ-bit prime with $\mu \geq \lambda$. One of the vital parts in the proof of previous theorem is to construct an invertible matrix $\mathbf{Y} \bmod p$ using the information available to $\mathcal{A}$ and then prove $\mathbf{Y} \cdot \boldsymbol{z}_t \bmod p$ does not leak sufficient information about $t \bmod p$. But, in this case we cannot construct the matrix $\mathbf{Y}$ in the same way that was considered in Game 2, since $\det(\mathbf{Y}\mathbf{Y}^T)$ could be a multiple of p.

According to the construction, the adversary gets secret-keys corresponding to at most $(l-1)$ linearly independent predicate vectors $\{\boldsymbol{y}_j\}_{j=1}^{l-1} \bmod p$. Any other queried secret-key is associated with a predicate vector that can be expressed as a linear combination of $\boldsymbol{y}_j$'s, hence delivers redundant information to the adversary. We will determine the view of $\mathcal{A}$ after it makes j predicate key queries for $0 \leq j \leq (l-1)$ and without loss of generality we assume that j predicate vectors are linearly independent modulo p. Our aim is to show that $t \bmod p$ is statistically hidden from the view of $\mathcal{A}$, for any $j \leq (l-1)$. We prove this by induction on j. Note that if $j = 0$, Game $2'$ is, in fact, same as Game 2 where the adversary has not queried any predicate key. Therefore, the induction hypothesis is true for $j = 0$ from the proof of Th. 3. Next, we take $j \in \{0, 1, \ldots, l-1\}$ and assume that the state $\mathsf{st} = \{(\boldsymbol{y}_i, \mathsf{sk}_{\boldsymbol{y}_i} = (\overline{\mathbf{y}}_i, s_{\overline{\mathbf{y}}_i}))\}_{i \in [j]}$ is independent of $t \bmod p$.

All the predicate vectors queried by $\mathcal{A}$ must belong to the $(l-1)$-dimensional subspace $\boldsymbol{x}^{*\perp p} = \{\boldsymbol{y} \in \mathbb{Z}_p^l : \langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0 \bmod p\}$. In particular, the predicate vectors $\{\overline{\mathbf{y}}_i\}_{i \in [j]}$ is a linearly independent subset of $\boldsymbol{x}^{*\perp p}$ and it can be extended into a set $\{\overline{\mathbf{y}}_i\}_{i \in [l-1]}$ which becomes a basis of $\boldsymbol{x}^{*\perp p}$. One can imagine this as the key generation of dummy predicate vectors $\{\boldsymbol{y}_i\}_{i=j+1}^{l-1}$ made by the challenger to get a smallest spanning set for $\boldsymbol{x}^{*\perp p}$. We define $\mathbf{Y}_{\mathsf{top}} = [\overline{\mathbf{y}}_1 \overline{\mathbf{y}}_2 \cdots \overline{\mathbf{y}}_{l-1}]^T \in \mathbb{Z}^{(l-1) \times l}$. Let $\boldsymbol{y}' \in \mathbb{Z}_p^l \setminus \boldsymbol{x}^{*\perp p}$ be such that it is deterministically computable by the adversary. We define $\mathbf{Y}_{\mathsf{bot}} = \overline{\mathbf{y}}'$ to be the canonical lift of $\boldsymbol{y}'^T$ over the integers and consider the matrix $\mathbf{Y} = \left[\frac{\mathbf{Y}_{\mathsf{top}}}{\mathbf{Y}_{\mathsf{bot}}}\right] \in \mathbb{Z}^{l \times l}$. By construction, $\mathbf{Y}$ is invertible modulo p and independent of t. We show that $\mathbf{Y} \cdot \boldsymbol{z}_t$ statistically hides $t \bmod p$.

Recall that $\boldsymbol{z}_t = t \cdot \boldsymbol{x}^* + a \cdot \boldsymbol{v} \bmod p$ and every rows of $\mathbf{Y}_{\mathsf{top}}$ lies in $\boldsymbol{x}^{*\perp p}$. Therefore, $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{z}_t$ is independent of t and, hence it is sufficient to prove that

$$\mathbf{Y}_{\mathsf{bot}} \cdot \boldsymbol{z}_t = t \cdot \langle \overline{\mathbf{y}}', \boldsymbol{x}^* \rangle + a \cdot \langle \overline{\mathbf{y}}', \boldsymbol{v} \rangle \bmod p$$

statistically hides $t \bmod p$ from the adversary's view. Our goal will be fulfilled if we can demonstrate that the distribution of $\langle \overline{\mathbf{y}}', \boldsymbol{v} \rangle \bmod p$ is statistically close to the uniform distribution modulo p as a is uniformly sampled from $\mathbb{Z}_p$ where p is a μ-bit prime with $\mu \geq \lambda$. Equivalently, if $\boldsymbol{v} \bmod p$ is statistically close to uniform over $\boldsymbol{x}^* \cdot \mathbb{Z}_p$ from the adversary's view, that is, $\boldsymbol{v} = \beta \cdot \boldsymbol{x}^*$ for $\beta \hookleftarrow \mathbb{Z}_p$, then $\langle \overline{\mathbf{y}}', \boldsymbol{v} \rangle = \beta \cdot \langle \overline{\mathbf{y}}', \boldsymbol{x}^* \rangle \bmod p$ would be uniformly distributed modulo p as $\langle \overline{\mathbf{y}}', \boldsymbol{x}^* \rangle \neq 0 \bmod p$.

From the public-key component $h_i = g^{u_i} h^{v_i}, i \in [l]$, the adversary information theoretically learns $\boldsymbol{s} = \boldsymbol{u} + \alpha \cdot \boldsymbol{v} \bmod n$. Consequently, the joint distribution of $(\boldsymbol{u} \bmod n, \boldsymbol{v} \bmod n)$ given $\boldsymbol{s} \bmod n$ becomes $(\boldsymbol{s} - \alpha \cdot \boldsymbol{v} \bmod n, \boldsymbol{v} \bmod n)$ where $\alpha \hookleftarrow \mathcal{D}_{\mathbb{Z},\sigma}$ and $\boldsymbol{v} \hookleftarrow \mathcal{D}_{\mathbb{Z}^l,\sigma}$. Since $\sigma > \tilde{s} \cdot \sqrt{\lambda} \cdot p^l \cdot (\sqrt{l})^{l-1} > p \cdot \tilde{s} \cdot \sqrt{\lambda}$ (for $l \geq 2$), item 2 of Lem. 5 (in Sec. 2.2) implies that $\boldsymbol{s}$ does not leak much information about $\boldsymbol{v} \bmod n$.

As discussed in the proof of Th. 3, the information learned by $\mathcal{A}$ from the predicate key queries can be completely determined by $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{u}$ and $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v} \in \mathbb{Z}^{l-1}$, and $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{u}$ does not give the adversary more information on $\boldsymbol{v} \bmod n$ than what he can gain from $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$. Therefore, it is sufficient to analyze the distribution of $\boldsymbol{v} \bmod n$ knowing $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$.

We define 1-dimensional lattice $\varLambda = \{\boldsymbol{y} \in \mathbb{Z} : \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{y} = \mathbf{0} \in \mathbb{Z}^{l-1}\}$. Since $\boldsymbol{x}^* \in \varLambda$, we can write $\varLambda = \boldsymbol{x}' \cdot \mathbb{Z}$ where $\boldsymbol{x}' = \gamma \cdot \boldsymbol{x}^* \bmod p$ for some $\gamma \in \mathbb{Z}_p^*$. Since $\boldsymbol{x}' / \gcd(x_1', \ldots, x_l') \in \varLambda$, one can assume that all the co-ordinates of $\boldsymbol{x}'$ are co-prime to each other.

Let $\boldsymbol{v}_0$ be an arbitrary vector satisfying $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}_0 = \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$. Then, in adversary's view the distribution of $\boldsymbol{v}$ becomes $\boldsymbol{v}_0 + V$ where V is a random variable following the distribution $\mathcal{D}_{\varLambda,\sigma,-\boldsymbol{v}_0}$. Next, consider the distribution of $\mathcal{D}_{\varLambda,\sigma,-\boldsymbol{v}_0}$ modulo the sublattice $\varLambda' = n \cdot \varLambda$. From Lem 3 of Sec. 2.1, the reduced distribution is 2ϵ-close to the uniform distribution over $\varLambda/\varLambda'$ if $\sigma > \eta_\epsilon(\varLambda')$ for some positive constant ϵ.

Taking $\epsilon = 2^{-\lambda-1}$ in Lem. 4 of Sec. 2.1, we get an upper bound

$$\eta_\epsilon(\varLambda') \leq \sqrt{\tfrac{\ln(2n(1+1/\epsilon))}{\pi}} \cdot \lambda_1(\varLambda') \leq \sqrt{\lambda} \cdot \lambda_1(\varLambda') = \sqrt{\lambda} \cdot n \cdot \|\boldsymbol{x}'\|_2,$$

since $\lambda_1(\varLambda') = n \cdot \|\boldsymbol{x}'\|_2$ as we have $\varLambda' = n \cdot \boldsymbol{x}' \cdot \mathbb{Z}$. Let $\varLambda_{\mathsf{top}}$ be the lattice spanned by the rows of $\mathbf{Y}_{\mathsf{top}} \in \mathbb{Z}^{(l-1)\times l}$. By construction, $\varLambda = \varLambda_{\mathsf{top}}^{\perp}$ is the orthogonal lattice of $\varLambda_{\mathsf{top}}$ and we have $\det(\varLambda) \leq \det(\varLambda_{\mathsf{top}})$ [32]. As the rows of $\mathbf{Y}_{\mathsf{top}}$ are linearly independent vectors over $\mathbb{Z}$, Lem. 1 of Sec. 2.1 implies

$$\det(\varLambda_{\mathsf{top}}) \leq \prod_{i=1}^{l-1} \|\overline{\mathbf{y}}_i\|_2 \leq (\sqrt{l} \cdot p)^{l-1}$$

Again $\|\boldsymbol{x}'\| = \det(\varLambda) \leq (\sqrt{l} \cdot p)^{l-1}$ implies $\eta_\epsilon(\varLambda') \leq \sqrt{\lambda} \cdot \tilde{s} \cdot p^l \cdot (\sqrt{l})^{l-1}$ as $n \leq p \cdot \tilde{s}$. Thus, the distribution $\mathcal{D}_{\varLambda,\sigma,-\boldsymbol{v}_0} \bmod \varLambda'$ is $2^{-\lambda}$-close to the uniform distribution over $\varLambda/\varLambda' \simeq \boldsymbol{x}'\mathbb{Z}_n$ if we set $\sigma > \tilde{s} \cdot \sqrt{\lambda} \cdot p^l \cdot (\sqrt{l})^{l-1}$. This ensures that $\boldsymbol{v} \bmod n$ is within a distance less than $2^{-\lambda}$ from the uniform distribution over $\boldsymbol{x}' \cdot \mathbb{Z}_n$. Moreover, the fact $p < n$ directly implies that $\boldsymbol{v} \bmod p$ is statistically $2^{-\lambda}$-close to the uniform distribution over $\boldsymbol{x}' \cdot \mathbb{Z}_p \simeq \gamma \cdot \boldsymbol{x}^* \mathbb{Z}_p \simeq \boldsymbol{x}^* \cdot \mathbb{Z}_p$ as $\gamma \in \mathbb{Z}_p^*$.

Therefore, the challenge bit b is statistically hidden in $M_b + a \cdot t \bmod p$ and it holds that $|\Pr[E_2] - \frac{1}{2}| \leq 2^{-\lambda}$. By combining all the probabilities one has

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{PH-pNIPE}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{DDH\text{-}f}}(\lambda, \mu) + 2^{-\lambda}$$

which is negligible in λ by our assumption.

5 Payload-hiding pNIPE based on HSM assumption

In this section, we describe two pNIPE schemes using a cyclic group with an easy DL subgroup. These constructions are inspired by the HSM-based pIPFE schemes of [14]. The first construction is for inner products over $\mathbb{Z}$ and is stateless whereas the second scheme is for inner products over $\mathbb{Z}_p$ and is stateful.

5.1 HSM based stateless pNIPE for Inner Products over $\mathbb{Z}$

Here, we employ the GenGroup algorithms (Def. 2) with the output of $\mathsf{Gen}(1^\lambda, 1^\mu)$ as a tuple of the form $(p, \tilde{s}, f, g_p, G, F, G^p)$ where the generator g is ignored. We need p to be a μ-bit prime with $\mu \geq \lambda$.

The domains related to this pNIPE are the same as they were in the DDH-f based pNIPE of Sec. 4.1. The infinity norm-bounds X and Y for the attribute, predicate vectors respectively should satisfy the condition $X, Y < (p/l)^{1/2}$, so that $\langle \boldsymbol{x}, \boldsymbol{y} \rangle < p$ for any $\boldsymbol{x} \in \mathcal{Q}$ and $\boldsymbol{y} \in \mathcal{P}$.

Construction. The stateless $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is described below where the correctness follows from the DDH-f based pNIPE of Sec. 4.1.

- $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^\mu, 1^l)$: A trusted authority generates a master public-key MPK and a master secret-key MSK using the steps below.
  - Generate $(p, \tilde{s}, f, g_p, G, F, G^p) \leftarrow \mathsf{Gen}(1^\lambda, 1^\mu)$, notations are consistent with Def. 2
  - Pick a vector $\boldsymbol{v} \hookleftarrow \mathcal{D}_{\mathbb{Z}^l, \sigma}$ where we set $\sigma > p^{3/2} \cdot \tilde{s} \cdot \sqrt{\lambda}$ (for the security proof of Th. 5) and write $\boldsymbol{v} = (v_1, \ldots, v_l)$
  - Compute $h_i = g_p^{v_i}$ for $1 \leq i \leq l$
  - Return the keys $\mathsf{MSK} = \boldsymbol{v}$ and $\mathsf{MPK} = (p, \tilde{s}, f, g_p, G, F, G^p, \{h_i\}_{i \in [l]})$
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \boldsymbol{y})$: For a predicate vector $\boldsymbol{y} = (y_1, \ldots, y_l) \in \mathcal{P}$, the authority returns the secret-key as $\mathsf{sk}_{\boldsymbol{y}} = \langle \boldsymbol{v}, \boldsymbol{y} \rangle$ which is computed over $\mathbb{Z}$.
- $\mathsf{CT}_{\boldsymbol{x}} \leftarrow \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}, M)$: A user encrypts his message $M \in \mathcal{M}$ with an attribute vector $\boldsymbol{x} = (x_1, \ldots, x_l) \in \mathcal{Q}$ utilizing the following steps.
  - Pick two random numbers $r \hookleftarrow \mathcal{D}_{\mathbb{Z}, \sigma'}, t \hookleftarrow \mathbb{Z}_p$ where $\sigma' > \tilde{s} \cdot \sqrt{\lambda}$ (for the security proof of Th. 5)
  - Set an element of G^p as $\mathsf{D} = g_p^r$
  - Compute $\mathsf{ct} = f^M \mathsf{D}^t$ and $\mathsf{ct}_i = f^{t x_i} h_i^r$ for i runs from 1 to l
  - Return the ciphertext $\mathsf{CT}_{\boldsymbol{x}} = \{\mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i \in [l]}\}$
- $\perp$ or $\zeta \leftarrow \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A decrypter first computes $\varrho \leftarrow \langle \boldsymbol{x}, \boldsymbol{y} \rangle$. If $\varrho = 0$, then it returns $\perp$; otherwise it applies the steps below.

  - Parse $\mathsf{CT}_{\boldsymbol{x}} = \{\mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i\in[l]}\}$
  - Compute $\mathsf{R}_t = (\prod_{i=1}^{l} \mathsf{ct}_i^{y_i})/\mathsf{D}^{\mathsf{sk}_{\boldsymbol{y}}}$
  - Run the algorithm Solve to get an element $\eta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{R}_t)$ of $\mathbb{Z}_p$
  - Compute $t \leftarrow \eta \cdot \varrho^{-1} \bmod p$ and then $\mathsf{ct}' \leftarrow \mathsf{ct}/\mathsf{D}^t$
  - Apply Solve to obtain $\zeta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{ct}')$ where $\zeta \in \mathbb{Z}_p$
  - Return the message as ζ

Theorem 5 *Assuming the hardness of* HSM *problem in the group G, the above pNIPE for inner products over $\mathbb{Z}$ provides adaptively payload-hiding security.* (The proof is available in SM-4)

5.2 HSM based stateful pNIPE for Inner Products over $\mathbb{Z}_p$

We present a stateful pNIPE for inner products over $\mathbb{Z}_p$ which is similar to our DDH-f based construction. As previously discussed, the authority requires to maintain a state in the key generation of predicate vectors for security reasons.

Following the construction in Sec. 5.1, here also we take $(p, \tilde{s}, f, g_p, G, F, G^p)$ as an output of $\mathsf{Gen}(1^\lambda, 1^\mu)$ where p is a μ-bit prime such that $\mu \geq \lambda$. The domains of this pNIPE are exactly the same as in the DDH-f-based pNIPE for inner products over $\mathbb{Z}_p$ (Sec. 4.2).

Construction. We describe the $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ where pNIPE.Setup and pNIPE.Enc are exactly the same as in the construction of Sec. 5.1 except the parameter σ (used in the distribution $\mathcal{D}_{\mathbb{Z}^l,\sigma}$) is now set greater than $\tilde{s} \cdot \sqrt{\lambda} \cdot p^l \cdot (\sqrt{l})^{l-1}$ for the security to hold. The initial state st in the output of pNIPE.Setup is considered to be empty for this scheme. The algorithms pNIPE.KeyGen and pNIPE.Dec are working as follows:

- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$: Let $\boldsymbol{y}$ be the j-th predicate vector for which a secret-key will be derived through this algorithm. At any stage the internal state st contains at most l tuples of the form $(\boldsymbol{y}_i, \mathsf{sk}_{\boldsymbol{y}_i})$ where $\mathsf{sk}_{\boldsymbol{y}_i}$ is the predicate secret-key corresponding to the predicate vector $\boldsymbol{y}_i$. W.l.o.g we assume that $j \leq l$ and the state st contains $j-1$ predicate vectors $\{\boldsymbol{y}_i\}_{i=1}^{j-1}$. The secret-key $\mathsf{sk}_{\boldsymbol{y}}$ is computed by utilizing the below steps.
  - If $\boldsymbol{y}$ is linearly independent of all predicate vectors $\{\boldsymbol{y}_i\}_{i=1}^{j-1}$ modulo p (present in the state) then
    1. Set $\overline{\mathbf{y}} = \boldsymbol{y} \bmod p$ where $\overline{\mathbf{y}} \in \{0, 1, \ldots, p-1\}^l$
    2. Compute $s_{\overline{\mathbf{y}}} = \langle \boldsymbol{v}, \overline{\mathbf{y}} \rangle \in \mathbb{Z}$
    3. Update the state $\mathsf{st} \leftarrow (\mathsf{st}, (\boldsymbol{y}, \mathsf{sk}_{\boldsymbol{y}} = (\overline{\mathbf{y}}, s_{\overline{\mathbf{y}}})))$
  - If there exist integers $k_1, \ldots, k_{j-1}$ such that $\boldsymbol{y} = \sum_{i=1}^{j-1} k_i \boldsymbol{y}_i \in \mathbb{Z}_p^l$ then
    1. Set $\overline{\mathbf{y}} = \sum_{i=1}^{j-1} k_i \overline{\mathbf{y}}_i \in \mathbb{Z}^l$
    2. Compute $s_{\overline{\mathbf{y}}} = \sum_{i=1}^{j-1} k_i s_{\overline{\mathbf{y}}_i} \in \mathbb{Z}$
  - Return the secret-key as $\mathsf{sk}_{\boldsymbol{y}} = (\overline{\mathbf{y}}, s_{\overline{\mathbf{y}}})$
- $\perp$ or $\zeta \leftarrow \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A decrypter first sets $\varrho \leftarrow \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ and returns $\perp$ If $\varrho \equiv 0 \pmod p$; otherwise it executes the following steps.
  - Parse $\mathsf{sk}_{\boldsymbol{y}} = (\overline{\mathbf{y}} = (\overline{y}_1, \ldots, \overline{y}_l), s_{\overline{\mathbf{y}}})$ and $\mathsf{CT}_{\boldsymbol{x}} = \{\mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i\in[l]}\}$
  - Compute $\mathsf{R}_t = (\prod_{i=1}^{l} \mathsf{ct}_i^{\overline{y}_i})/\mathsf{D}^{s_{\overline{\mathbf{y}}}}$

- Run the algorithm Solve to get $\eta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{R}_t)$
- Compute $t \leftarrow \eta \cdot \varrho^{-1} \bmod p$ and then $\mathsf{ct}' \leftarrow \mathsf{ct}/\mathsf{D}^t$
- Get an element $\zeta \leftarrow \mathsf{Solve}(\mathsf{param}, \mathsf{ct}')$ of $\mathbb{Z}_p$
- Return the message as ζ

Theorem 6 *Assuming the hardness of* HSM *problem in the group G, the above pNIPE for inner products over $\mathbb{Z}_p$ provides adaptively payload-hiding security.* (The proof is available in SM-5)

6 Generic Construction of NIPE from IPFE

In this section, we give a generic construction of a NIPE from IPFE and prove that the NIPE is attribute-hiding in public-key setting and full-hiding in private-key setting.

6.1 Generic Transformation of pIPFE to Attribute-hiding pNIPE

We describe how to use the indistinguishability-based security of a pIPFE to achieve the attribute-hiding security for a pNIPE through a generic construction. Let us consider a stateful $\mathsf{pIPFE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ with a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}'$ and an inner product space $\mathcal{I}$. We construct a stateful $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ with the same predicate space $\mathcal{P}$, the attribute space $\mathcal{Q}$, the inner product space $\mathcal{I}$ and a message space $\mathcal{M}$ such that $\mathcal{P}, \mathcal{Q}, \mathcal{Q}' \subseteq \mathcal{I}^l$, $\mathcal{M} \subset \mathcal{I}$ and for any $\boldsymbol{x} = (x_1, \ldots, x_l) \in \mathcal{Q}$, $M \in \mathcal{M}$ it holds that $M \cdot \boldsymbol{x} \in \mathcal{Q}'$ where $M \cdot \boldsymbol{x} = (Mx_1, \ldots, Mx_l)$. It is also required that the division operation can be efficiently executed in $\mathcal{I}$, that is for any product value $\alpha \cdot \beta \in \mathcal{I}$, one can easily compute β if α is known.

Construction. Now, we formally describe our generic transformation below.

- $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^l)$: A trusted authority computes $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pIPFE.Setup}(1^\lambda, 1^l)$ and outputs MPK as the master public-key and $\mathsf{MSK}, \mathsf{st}$ as the master secret-key, state of the pNIPE respectively.
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$: A predicate holder gets the secret-key $\mathsf{sk}_{\boldsymbol{y}}$ corresponding to a predicate vector $\boldsymbol{y} \in \mathcal{P}$ from the trusted authority which computes $\mathsf{sk}_{\boldsymbol{y}}$ as the output of $\mathsf{pIPFE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$. The authority updates the state st if required.
- $\mathsf{CT}_{\boldsymbol{x}} \leftarrow \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}, M)$: A user encrypts a message $M \in \mathcal{M}$ with its attribute $\boldsymbol{x} \in \mathcal{Q}$ by computing $\mathsf{ct}_{\boldsymbol{x}} \leftarrow \mathsf{pIPFE.Enc}(\mathsf{MPK}, \boldsymbol{x})$ and $\mathsf{ct}_{M\cdot\boldsymbol{x}} \leftarrow \mathsf{pIPFE.Enc}(\mathsf{MPK}, M \cdot \boldsymbol{x})$. It publishes the ciphertext as $\mathsf{CT}_{\boldsymbol{x}} = (\mathsf{ct}_{\boldsymbol{x}}, \mathsf{ct}_{M\cdot\boldsymbol{x}})$.
- $\perp$ or $\zeta \leftarrow \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A decrypter who has a secret-key $\mathsf{sk}_{\boldsymbol{y}}$ and a ciphertext $\mathsf{CT}_{\boldsymbol{x}} = (\mathsf{ct}_{\boldsymbol{x}}, \mathsf{ct}_{M\cdot\boldsymbol{x}})$, first sets $\eta \leftarrow \mathsf{pIPFE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}})$. If $\eta = 0$, then it outputs $\perp$; otherwise it computes $\eta' \leftarrow \mathsf{pIPFE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{M\cdot\boldsymbol{x}})$ and returns η'/η.

Correctness. Suppose $\boldsymbol{y} \in \mathcal{P}$ and $\boldsymbol{x} \in \mathcal{Q}$ be such that their inner product is non-zero. Hence, by the correctness of IPFE we have $\eta = \langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0$. By the assumptions on domains of IPFE, $M \cdot \boldsymbol{x} \in \mathcal{Q}'$ which ensures that

$\mathsf{IPFE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{M\cdot\boldsymbol{x}})$ returns $M \cdot \langle \boldsymbol{x}, \boldsymbol{y}\rangle$ with overwhelming probability. Therefore, $\eta' = M\cdot\langle \boldsymbol{x}, \boldsymbol{y}\rangle$ which implies $\eta'/\eta = M$. On the other hand, $\langle \boldsymbol{x}, \boldsymbol{y}\rangle = 0$ implies $\eta = 0$, hence the decryption fails to recover M.

Theorem 7 *Assuming the underlying pIPFE is indistinguishability-based secure under chosen plaintext attacks, the above pNIPE provides adaptively attribute-hiding security.*

Proof. To prove this theorem, we consider the following games. We start with Game 0 which is the standard security AH-pNIPE experiment (Def. 11) where the challenger chooses the random bit as $b = 0$. Then we modify this game in Game 1 and finally end up in Game 2 where the random bit (chosen by the challenger) is changed to $b = 1$. We establish indistinguishability between these games using the security of pIPFE. Let E_i denotes the event $b = b'$ in Game i where b' is the bit output by the adversary $\mathcal{A}$ in **Guessing phase**. Now, we formally describe the games:

Game 0: This is the original security experiment as described in Def. 11. The challenge attribute-message pairs are $(\boldsymbol{x}_0, M_0), (\boldsymbol{x}_1, M_1)$ satisfying $\langle \boldsymbol{x}_0, \boldsymbol{y}\rangle = \langle \boldsymbol{x}_1, \boldsymbol{y}\rangle = 0$ if $M_0 \neq M_1$ or $\langle \boldsymbol{x}_0 - \boldsymbol{x}_1, \boldsymbol{y}\rangle = 0$ if $M_0 = M_1$, for all predicate vectors $\boldsymbol{y}$ queried by the adversary $\mathcal{A}$ in **Query phase 1** and **Query phase 2**. The challenger fixes $b = 0$ and sends the challenge ciphertext as $\mathsf{CT}_{\boldsymbol{x}}^{(0,0)} = (\mathsf{ct}_{\boldsymbol{x}_0}, \mathsf{ct}_{M_0\cdot\boldsymbol{x}_0})$ where $\mathsf{ct}_{\boldsymbol{x}_0} \leftarrow \mathsf{IPFE.Enc}(\mathsf{MPK}, \boldsymbol{x}_0)$, $\mathsf{ct}_{M_0\cdot\boldsymbol{x}_0} \leftarrow \mathsf{IPFE.Enc}(\mathsf{MPK}, M_0\cdot\boldsymbol{x}_0)$. Therefore, Game 0 is identical to $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{AH\text{-}pNIPE}}(1^\lambda, 0)$.

Game 1: In this game the first component $\mathsf{ct}_{\boldsymbol{x}_0}$ of challenge ciphertext is replaced by $\mathsf{ct}_{\boldsymbol{x}_1} \leftarrow \mathsf{pIPFE.Enc}(\mathsf{MPK}, \boldsymbol{x}_1)$. Therefore, the challenge ciphertext becomes $\mathsf{CT}_{\boldsymbol{x}}^{(1,0)} = (\mathsf{ct}_{\boldsymbol{x}_1}, \mathsf{ct}_{M_0\cdot\boldsymbol{x}_0})$. Let $\mathcal{K}$ be a set of all such $\boldsymbol{y} \in \mathcal{P}$ for which $\mathcal{A}$ asked $\mathsf{sk}_{\boldsymbol{y}}$ from the challenger in query phase. Since $\mathcal{A}$ is an admissible adversary, $\langle \boldsymbol{x}_0, \boldsymbol{y}\rangle = \langle \boldsymbol{x}_1, \boldsymbol{y}\rangle$ holds for all $\boldsymbol{y} \in \mathcal{K}$. Therefore, the distinguishing advantage of $\mathcal{A}$ between Game 0 and Game 1 is exactly the same as that in distinguishing between the experiments $\mathsf{Expt}_{\mathcal{B}}^{\mathsf{IND\text{-}pIPFE}}(1^\lambda, 0)$ and $\mathsf{Expt}_{\mathcal{B}}^{\mathsf{IND\text{-}pIPFE}}(1^\lambda, 0)$ (Def. 6), $\mathcal{B}$ is an adversary appointed for the IPFE security experiment. This ensures that $|\Pr[E_0] - \Pr[E_1]| = \mathsf{Adv}_{\mathcal{B}}^{\mathsf{IND\text{-}pIPFE}}(\lambda)$.

Game 2: This game is the same as Game 1 except the second component $\mathsf{ct}_{M_0\boldsymbol{x}_0}$ of the challenge ciphertext is now computed as $\mathsf{ct}_{M_1\cdot\boldsymbol{x}_1} \leftarrow \mathsf{pIPFE.Enc}(\mathsf{MPK}, M_1 \cdot \boldsymbol{x}_1)$. Suppose, $\mathcal{A}$ has a non-negligible advantage in distinguishing between Game 1 and Game 2. We construct an adversary $\mathcal{B}$ against the indistinguishability based security of the underlying pIPFE scheme as follows:

1. **Setup:** The pIPFE challenger gets $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pIPFE.Setup}(1^\lambda, 1^l)$ and gives MPK to $\mathcal{B}$. Then $\mathcal{B}$ passes MPK as the pNIPE master public-key to $\mathcal{A}$.
2. **Query phase 1:** The adversary $\mathcal{A}$ makes secret-key queries for any arbitrary predicate vectors $\boldsymbol{y} \in \mathcal{P}$ which $\mathcal{B}$ forwards to the pIPFE challenger and the challenger returns $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pIPFE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$. Then, $\mathcal{B}$ sends $\mathsf{sk}_{\boldsymbol{y}}$ to $\mathcal{A}$.
3. **Challenge phase:** The adversary $\mathcal{A}$ adaptively outputs attribute-message pairs $(\boldsymbol{x}_0, M_0)$, $(\boldsymbol{x}_1, M_1)$. Then, $\mathcal{B}$ sets $\boldsymbol{X}_0 = M_0 \cdot \boldsymbol{x}_0$, $\boldsymbol{X}_1 = M_1 \cdot \boldsymbol{x}_1$ and

sends $(\boldsymbol{X}_0, \boldsymbol{X}_1)$ to its challenger. We note that $\boldsymbol{X}_0, \boldsymbol{X}_1 \in \mathcal{Q}'$ by construction and hence are eligible for challenge attributes. The challenger picks a random bit b and returns $\mathsf{ct}_{\boldsymbol{X}_b} \leftarrow \mathsf{pIPFE.Enc}(\mathsf{MPK}, \boldsymbol{X}_b)$ to $\mathcal{B}$. Next, $\mathcal{B}$ computes $\mathsf{ct}_{\boldsymbol{x}_1} \leftarrow \mathsf{pIPFE.Enc}(\mathsf{MPK}, \boldsymbol{x}_1)$ and sends $\mathsf{CT}_{\boldsymbol{x}}^{(1,b)} = (\mathsf{ct}_{\boldsymbol{x}_1}, \mathsf{ct}_{\boldsymbol{X}_b})$ as the challenge ciphertext to $\mathcal{A}$.

4. **Query phase 2:** This is the same as Query phase 1.
5. **Guessing phase:** Finally, $\mathcal{A}$ outputs a bit b' which $\mathcal{B}$ returns.

Since $\mathcal{A}$ is a legitimate adversary, it holds that $\langle \boldsymbol{x}_0, \boldsymbol{y} \rangle = \langle \boldsymbol{x}_1, \boldsymbol{y} \rangle = 0$ if $M_0 \neq M_1$, otherwise $\langle \boldsymbol{x}_0 - \boldsymbol{x}_1, \boldsymbol{y} \rangle = 0$ for all predicate vectors $\boldsymbol{y}$ queried by $\mathcal{A}$. Thus, one can observe that

$$M_0 \langle \boldsymbol{x}_0, \boldsymbol{y} \rangle = \begin{cases} M_1 \langle \boldsymbol{x}_0, \boldsymbol{y} \rangle = M_1 \langle \boldsymbol{x}_1, \boldsymbol{y} \rangle & \text{if } M_0 = M_1 \\ M_0 \cdot 0 = M_1 \cdot 0 = M_1 \langle \boldsymbol{x}_1, \boldsymbol{y} \rangle & \text{if } M_0 \neq M_1 \end{cases}$$

Therefore, $\langle \boldsymbol{X}_0, \boldsymbol{y} \rangle = M_0 \cdot \langle \boldsymbol{x}_0, \boldsymbol{y} \rangle = M_1 \cdot \langle \boldsymbol{x}_1, \boldsymbol{y} \rangle = \langle \boldsymbol{X}_1, \boldsymbol{y} \rangle$ for all $\boldsymbol{y}$ queried by $\mathcal{B}$ to its challenger. If the pIPFE challenger chooses the bit $b = 0$, then the challenge ciphertext becomes $\mathsf{CT}_{\boldsymbol{x}}^{(1,0)} = (\mathsf{ct}_{\boldsymbol{x}_1}, \mathsf{ct}_{M_0 \cdot \boldsymbol{x}_0})$ and $\mathcal{B}$ simulates Game 1. If the bit $b = 1$, then the challenge ciphertext is computed as $\mathsf{CT}_{\boldsymbol{x}}^{(1,1)} = (\mathsf{ct}_{\boldsymbol{x}_1}, \mathsf{ct}_{M_1 \cdot \boldsymbol{x}_1})$ and $\mathcal{B}$ simulates Game 2. Thus, the advantage of $\mathcal{A}$ in distinguishing between Game 1 and Game 2 is exactly same for $\mathcal{B}$ in the $\mathsf{IND\text{-}pIPFE}$ game described above and we conclude that $|\Pr[E_1] - \Pr[E_2]| = \mathsf{Adv}_{\mathcal{B}}^{\mathsf{IND\text{-}pIPFE}}(\lambda)$.

It can be observed that Game 2 is the original $\mathsf{AH\text{-}pNIPE}$ experiment where the challenge bit is chosen as 1. By triangular inequality we have $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{AH\text{-}pNIPE}}(\lambda) \leq 2 \cdot \mathsf{Adv}_{\mathcal{B}}^{\mathsf{IND\text{-}pIPFE}}(\lambda)$ which is negligible in λ by our assumption.

Instantiations for pNIPE Here we utilize the existing indistinguishability-based secure pIPFE schemes of [2,14] in the above generic construction and fabricate adaptively secure pNIPE under desirable assumptions like the DDH, DDH-f, HSM, LWE and DCR. In fact, these pNIPEs would be the first of their kind to provide adaptive security with attribute hiding feature. Since the idea of our generic transformation is similar to that of [25], some of our instantiations are also analogous to their adaptively secure pNIPE schemes. However, pNIPEs of [25] are not attribute-hiding. We need to restrict the size of message space to logarithmic when the space of inner products is of polynomial size.

DDH **based construction.** Agrawal et al. [2] proposed a DDH-based stateless pIPFE with the inner products lying in $\mathbb{Z}_q$ where q is a prime number indicating order of the underlying cyclic group G. It is required to bound $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ into polynomial range so that one can efficiently compute $\log_g(g^{\langle \boldsymbol{x}, \boldsymbol{y} \rangle})$ at the end of the decryption. To apply the pIPFE of [2] in our generic construction, we ignore the step of computing discrete logarithm, i.e. the output of pIPFE-decryption is $g^{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$. The domains of our pNIPE are taken as $\mathcal{P} = \mathcal{Q} = \mathcal{Q}' = \mathbb{Z}_q^l$, $\mathcal{I} = \mathbb{Z}_q$ and a polynomial size message space $\mathcal{M} \subset \mathbb{Z}_q$. We cannot ensure that the product $M \cdot \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ always lies in a polynomially range for any $M \in \mathcal{M}$, $\boldsymbol{x} \in \mathcal{Q}$ and $\boldsymbol{y} \in \mathcal{P}$, hence one may not be able to perform a discrete logarithm on $g^{M \cdot \langle \boldsymbol{x}, \boldsymbol{y} \rangle}$ in polynomial time. On receiving a ciphertext $\mathsf{CT}_{\boldsymbol{x}} = (\mathsf{ct}_{\boldsymbol{x}}, \mathsf{ct}_{M \cdot \boldsymbol{x}})$, the decrypter outputs a message $M \in \mathcal{M}$ such that $\eta^M = \eta'$ over $\mathbb{Z}_q$ where $\eta = g^{\langle \boldsymbol{x}, \boldsymbol{y} \rangle} \leftarrow$ $\mathsf{pIPFE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}})$ and $\eta' = g^{M \cdot \langle \boldsymbol{x}, \boldsymbol{y} \rangle} \leftarrow \mathsf{pIPFE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{M \cdot \boldsymbol{x}})$.

Now, we compare the efficiency of our DDH-based pNIPE in Sec. 3 with our generic construction instantiated with [2] as described above. To encrypt a short message $M \in \mathbb{Z}_q$, the former technique computes only $(l+3)$ elements of G whereas the latter needs $(2l+4)$ elements. Also, by construction, the decryption requires less computation in the first method (roughly, the time is doubled in the second method). Therefore, in terms of computational efficiency, the first method is more preferable. However, it only provides adaptively payload-hiding security in contrast to the second method which exhibits an adaptively attribute-hiding pNIPE. Both of these methods can encrypt a polynomial size small message and in order to deal with long messages, one can either break the message into smaller parts and then apply encryption for each of these parts or one can use a key encapsulation mechanism (KEM). In KEM, a small key $K \in \mathbb{Z}_q$ is encrypted using the pNIPE and then use a suitable symmetric-key encryption (SKE) technique to encrypt the long message M using K. Accordingly, the decryption procedure first computes the key K using $\mathsf{pNIPE.Dec}$ and then use it to obtain M by running decryption of the SKE system.

$\mathsf{DDH\text{-}f}$ based constructions. Castagnos et al. [14] gave two indistinguishability-based secure pIPFE schemes using a DDH group G with an easy DL subgroup F (Def. 2), secure under the $\mathsf{DDH\text{-}f}$ assumption. One of their constructions is stateless pIPFE with inner products over $\mathbb{Z}$ and the other is stateful pIPFE with inner products over $\mathbb{Z}_p$. Instantiating our generic construction in Sec. 6.1 with these pIPFEs of [14], we obtain two adaptively attribute-hiding pNIPE schemes under the $\mathsf{DDH\text{-}f}$ assumption. The domains for the stateless pNIPE over $\mathbb{Z}$ can be taken as $\mathcal{I} = \mathbb{Z}$, $\mathcal{P} = \mathcal{Q}' = \{\boldsymbol{y} \in \mathbb{Z}^l : \|\boldsymbol{y}\|_\infty < (p/2l)^{1/2}\}$, $\mathcal{M} = \{M \in \mathbb{Z} : |M| < \mathfrak{B}\}$, $\mathcal{Q} = \{\boldsymbol{x} \in \mathbb{Z}^l : \|\boldsymbol{x}\|_\infty < (p/2l\mathfrak{B}^2)^{1/2}\}$ where the prime p is the order of the cyclic group F and $\mathfrak{B}$ is a polynomially bounded (sufficiently small) integer. For the second stateful pNIPE over $\mathbb{Z}_p$, the domains are $\mathcal{I} = \mathbb{Z}_p$, $\mathcal{P} = \mathcal{Q} = \mathcal{Q}' = \mathbb{Z}_p^l$ and $\mathcal{M} = \mathbb{Z}_p$. We note that the randomness t can be chosen from a sufficiently small interval to ensure the correctness of the scheme. These establish the first attribute-hiding pNIPE schemes that are secure under an assumption weaker than the DDH assumption.

To compare our $\mathsf{DDH\text{-}f}$ based pNIPEs in Sec. 4 with the above pNIPEs built through a generic transformation from pIPFEs of [14], we note that encryption of our first technique requires $(l+3)$ elements of G whereas our second method needs $(2l+4)$ elements. Also, the decryption time in the second approach is roughly doubled that of in the first method. Therefore, the former constructions are more efficient than the latter schemes. However, we emphasize that for the adaptively attribute-hiding security one has to consider the latter pNIPE schemes as the former pNIPEs provide only adaptively payload-hiding security.

HSM based constructions. We use the HSM based pIPFE schemes of [14] in our generic transformation to achieve two adaptively attribute-hiding pNIPE schemes under the same assumption: one is stateless with inner products over $\mathbb{Z}$

and the other is stateful with inner products over $\mathbb{Z}_p$. The domains associated with these pNIPEs are exactly the same as in the above $\mathsf{DDH\text{-}f}$ based constructions. The HSM based pNIPEs in Sec. 5 provide better efficiency than the above conversion as a ciphertext contains $(l+2)$ elements of G in the former method whereas it is $(2l+2)$ in the latter technique. However, in terms of security the former pNIPEs are only adaptively payload-hiding secure and the latter schemes are adaptively attribute-hiding secure. One can observe that the ciphertext of our HSM based pNIPEs (from Sec. 5 and this section) require less elements of G than our $\mathsf{DDH\text{-}f}$ based pNIPEs (from Sec. 4 and this section respectively). Therefore, our HSM based pNIPEs run computationally faster than our $\mathsf{DDH\text{-}f}$ based pNIPEs, although the $\mathsf{DDH\text{-}f}$ is weaker than the HSM assumption.

We describe LWE and DCR based AH-pNIPEs in SM-6 due to the page limit.

6.2 Generic Transformation of sIPFE to Full-hiding sNIPE

Public-key NIPE allows anyone to encrypt a message with an arbitrarily chosen attribute $\boldsymbol{x}$. Knowing a secret-key $\mathsf{sk}_{\boldsymbol{y}}$ one can easily disclose the predicate vector $\boldsymbol{y}$ by setting $\boldsymbol{x}$ suitably and computing the inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$ using ciphertexts corresponding to the attributes. Thus, the full-hiding security may not be feasible for public-key NIPEs. We consider our generic construction of Sec. 6.1 in private-key setting where encryption and key generation are both executed by the central authority in the presence of master secret-key MSK. Instead of a master public-key MPK, we use a public parameter pp which is available to all the users. The domains of the sIPFE and sNIPE satisfy the same conditions described in Sec. 6.1.

Theorem 8 *Assuming the underlying sIPFE is full-hiding, our construction of the sNIPE provides full-hiding security.* (The proof is available in SM-7)

Instantiations for sNIPE We use existing sIPFE schemes [38,28] in our generic construction to achieve full-hiding sNIPE (FH-sNIPE). Almost all full-hiding sIPFE (FH-sIPFE) constructions are proven secure under the SXDH and DLIN assumptions (in a pairing group). We utilize DLIN based scheme of [38] and SXDH based scheme of [28] as these are more efficient among the existing sIPFEs in terms of ciphertext and secret-key sizes. In [38], Tomida et al. provided FH-sIPFE with a secret-key or ciphertext size of $(2l+5)$ group elements whereas the master secret-key size was $(4l^2+18l+20)$. The master secret-key size was $O(l^2)$ in all previous constructions, until Kim et al. [28] achieved a FH-sIPFE where the master secret-key contains only $(6l+4)$ field elements (integers) and each of secret-key or ciphertext requires $(2l+8)$ group elements. We note that l denotes the dimension of the attribute or predicate space. The FH-sNIPEs constructed below is restricted with a logarithmic size message space as the underlying sIPFE computes inner products in a polynomially bounded range.

DLIN based construction. Tomida et al. [38] built a DLIN based FH-sIPFE utilizing dual pairing vector spaces (introduced by Lewko et al. [29]). Their scheme can be used in any type of pairing group of prime order q. The inner products belong to a polynomially bounded subset of $\mathbb{Z}_q$ and the attribute or predicate space is $\mathbb{Z}_q^l$. At the end of decryption algorithm of sIPFE a discrete logarithm is performed on $g_T^{\langle \boldsymbol{x},\boldsymbol{y}\rangle}$ in the target group G_T to recover the inner product. But, this step is not required for the application to our generic construction. So, we slightly modify the sIPFE to output only $g_T^{\langle \boldsymbol{x},\boldsymbol{y}\rangle}$ while decryption. The domains associated to our sNIPE are $\mathcal{P} = \mathcal{Q} = \mathcal{Q}' = \mathbb{Z}_q^l$, $\mathcal{I} = \mathbb{Z}_q$ with a polynomial size message space $\mathcal{M} \subset \mathbb{Z}_q$. On receiving a ciphertext $\mathsf{CT}_{\boldsymbol{x}} = (\mathsf{ct}_{\boldsymbol{x}}, \mathsf{ct}_{M\cdot\boldsymbol{x}})$ corresponding to a message $M \in \mathcal{M}$, a predicate $\boldsymbol{y} \in \mathbb{Z}_q^l$, an attribute $\boldsymbol{x} \in \mathbb{Z}_q^l$, the decryption of our sNIPE first computes $\eta = g_T^{\langle \boldsymbol{x},\boldsymbol{y}\rangle} \leftarrow \mathsf{sIPFE.Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}})$ and recovers the message M by checking the equality $\eta^M = \eta'$ (over $\mathbb{Z}_q$) for $M \in \mathcal{M}$ where $\eta' = g_T^{M\cdot\langle \boldsymbol{x},\boldsymbol{y}\rangle} \leftarrow \mathsf{sIPFE.Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{M\cdot\boldsymbol{x}})$. This results in a DLIN based FH-sNIPE where secret-keys and ciphertexts consist of $(2l + 5)$ and $(4l + 10)$ group elements respectively.

SXDH based construction. Kim et al. [28] defined a new approach (avoiding DPVS) to provide a FH-sIPFE under the SXDH assumption in an asymmetric bilinear pairing group of prime order q. Their scheme computes inner products over $\mathbb{Z}_q$ and the predicate or attribute vectors are in $\mathbb{Z}_q^l$. A discrete logarithm of $g_T^{\langle \boldsymbol{x},\boldsymbol{y}\rangle}$ is determined at the final step of the decryption. We ignore this step of the decryption so that it outputs $g_T^{\langle \boldsymbol{x},\boldsymbol{y}\rangle}$. Employing this sIPFE we achieve a FH-sNIPE based on SXDH assumption via our generic transformation where the domains are taken as $\mathcal{P} = \mathcal{Q} = \mathcal{Q}' = \mathbb{Z}_q^l$, $\mathcal{I} = \mathbb{Z}_q$ with a polynomial size message space $\mathcal{M} \subset \mathbb{Z}_q$. We follow similar approach for the decryption of our sNIPE as in the above construction. Consequently, if $\langle \boldsymbol{x}, \boldsymbol{y}\rangle \neq 0 \bmod q$ then the decryption requires to search in the set $\mathcal{M}$ to find a message M satisfying $\eta^M = \eta'$ in $\mathbb{Z}_q$. The secret-keys and ciphertexts of the resulting sNIPE contain $(2l + 8)$ and $(4l + 16)$ group elements respectively. We note that the master secret-key of this sNIPE has $O(l)$ elements whereas it is $O(l^2)$ in the previous construction.

References

1. L. M. Adleman. The function field sieve. In *International Algorithmic Number Theory Symposium*, pages 108–121. Springer, 1994.
2. S. Agrawal, B. Libert, and D. Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In *Annual International Cryptology Conference*, pages 333–362. Springer, 2016.
3. M. Ambrona, G. Barthe, and B. Schmidt. Generic transformations of predicate encodings: Constructions and applications. In *Annual International Cryptology Conference*, pages 36–66. Springer, 2017.
4. N. Attrapadung and B. Libert. Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation. In *International Workshop on Public Key Cryptography*, pages 384–402. Springer, 2010.
5. N. Attrapadung, B. Libert, and E. De Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *International Workshop on Public Key Cryptography*, pages 90–108. Springer, 2011.
6. J.-F. Biasse, M. J. Jacobson, and A. K. Silvester. Security estimates for quadratic field based cryptosystems. In *Australasian Conference on Information Security and Privacy*, pages 233–247. Springer, 2010.

7. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *International conference on the theory and applications of cryptographic techniques*, pages 506–522. Springer, 2004.
8. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Annual international cryptology conference*, pages 213–229. Springer, 2001.
9. D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit abe and compact garbled circuits. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 533–556. Springer, 2014.
10. D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *Theory of Cryptography Conference*, pages 253–273. Springer, 2011.
11. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *Theory of Cryptography Conference*, pages 535–554. Springer, 2007.
12. G. Castagnos and F. Laguillaumie. On the security of cryptosystems with quadratic decryption: the nicest cryptanalysis. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 260–277. Springer, 2009.
13. G. Castagnos and F. Laguillaumie. Linearly homomorphic encryption from DDH. In *Cryptographers Track at the RSA Conference*, pages 487–505. Springer, 2015.
14. G. Castagnos, F. Laguillaumie, and I. Tucker. Practical fully secure unrestricted inner product functional encryption modulo p. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 733–764. Springer, 2018.
15. J. Chen, B. Libert, and S. C. Ramanna. Non-zero inner product encryption with short ciphertexts and private keys. In *International Conference on Security and Cryptography for Networks*, pages 23–41. Springer, 2016.
16. J. Chen and H. Wee. Doubly spatial encryption from dbdh. *Theoretical Computer Science*, 543:79–89, 2014.
17. R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 45–64. Springer, 2002.
18. A. Fiat and M. Naor. Broadcast encryption. In *Annual International Cryptology Conference*, pages 480–491. Springer, 1993.
19. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 45(3):882–929, 2016.
20. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206. ACM, 2008.
21. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In *Annual Cryptology Conference*, pages 162–179. Springer, 2012.
22. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. *Journal of the ACM (JACM)*, 62(6):45, 2015.
23. S. Gorbunov and D. Vinayagamurthy. Riding on asymmetry: Efficient abe for branching programs. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 550–574. Springer, 2015.

24. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. Acm, 2006.
25. S. Katsumata and S. Yamada. Non-zero inner product encryption schemes from various assumptions: Lwe, ddh and dcr. PKC, 2019.
26. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *annual international conference on the theory and applications of cryptographic techniques*, pages 146–162. Springer, 2008.
27. Y. Kawai and K. Takashima. Fully-anonymous functional proxy-re-encryption. Cryptology ePrint Archive, Report 2013/318, 2013. `https://eprint.iacr.org/2013/318`.
28. S. Kim, J. Kim, and J. H. Seo. A new approach to practical function-private inner product encryption. *Theoretical Computer Science*, 2019.
29. A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 62–91. Springer, 2010.
30. A. Lewko, A. Sahai, and B. Waters. Revocation systems with very small private keys. In *2010 IEEE Symposium on Security and Privacy*, pages 273–285. IEEE, 2010.
31. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing*, 37(1):267–302, 2007.
32. P. Nguyen and J. Stern. Merkle-hellman revisited: a cryptanalysis of the quvanstone cryptosystem based on group factorizations. In *Annual International Cryptology Conference*, pages 198–212. Springer, 1997.
33. T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *Annual cryptology conference*, pages 191–208. Springer, 2010.
34. T. Okamoto and K. Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. *Designs, Codes and Cryptography*, 77(2-3):725–771, 2015.
35. A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 457–473. Springer, 2005.
36. A. Shamir. Identity-based cryptosystems and signature schemes. In *Workshop on the theory and application of cryptographic techniques*, pages 47–53. Springer, 1984.
37. E. Shen, E. Shi, and B. Waters. Predicate privacy in encryption systems. In *Theory of Cryptography Conference*, pages 457–473. Springer, 2009.
38. J. Tomida, M. Abe, and T. Okamoto. Efficient functional encryption for inner-product values with full-hiding security. In *International Conference on Information Security*, pages 408–425. Springer, 2016.
39. S. Yamada, N. Attrapadung, G. Hanaoka, and N. Kunihiro. A framework and compact constructions for non-monotonic attribute-based encryption. In *International Workshop on Public Key Cryptography*, pages 275–292. Springer, 2014.

Supplementary Material

SM-1 History of NIPE

The journey of pNIPE began with its introduction in [26] by Katz et al., although the first concrete pNIPE construction came into light in [4] by Attrapadung and Libert. In [4], two *co-selectively* secure PH-pNIPE schemes were proposed: one is secure under the q-Decision Multi-Exponent Bilinear Diffie-Hellman (q-DMEBDH) assumption whereas the other is secure under the Decision Linear (DLIN) assumption and Decision Bilinear Diffie-Hellman (DBDH) assumption. Additionally, these pNIPEs yield *identity-based revocation* (IBR) systems [30] with $O(1)$-size ciphertext. Even though the size of ciphertexts is constant and independent of the dimension of attribute or predicate space (as long as the attribute vector is not considered as a part of ciphertext), the security model of their NIPE is based on the unrealistic co-selective model. Soon after, Okamoto and Takashima proposed adaptively secure pNIPEs that are weakly attribute-hiding [33] and payload-hiding [34] under the DLIN assumption. They utilize the dual-pairing vector space (DPVS) technique of [29] to obtain constant-size ciphertexts or constant-size secret-keys for the pNIPE [34]. Consequently, this implies an IBR with constant-size ciphertexts or constant-size secret-keys that is adaptively secure in standard model. In [16], a pNIPE was established via *doubly-spatial encryption* technique which is selectively secure under the DBDH alone. More efficient pNIPE realizations (through non-monotonic ABE) were provided in [5,39] that are selectively secure under the n-Decision Bilinear Diffie-Hellman Exponent (n-DBDHE) assumption where n denotes the dimension of vectors used in the system. The first pNIPE simultaneously achieving $O(1)$-size ciphertexts and $O(1)$-size secret-keys was proposed in [15]. The pNIPE is selectively secure under the n-DBDHE. The second pNIPE of [15] with constant size secret-key is selectively secure under non-interactive and falsifiable assumptions in composite order groups. Both these pNIPEs require $O(n)$ exponentiations in a bilinear group along with pairing computations which make it inefficient for practical implementations. The main building block of all these constructions is bilinear maps and the systems are proven secure under certain number theoretic assumptions in a pairing group.

Therefore, constructing pNIPE under standard assumption without pairing was still open, until Katsumata and Yamada [25] came up with a direct approach to build pNIPE (over $\mathbb{Z}$ and $\mathbb{Z}_p$) that is selectively secure under standard LWE assumption. They also gave a generic construction of NIPE from existing IPFEs. Utilizing the public-key IPFEs (pIPFE) of [2], this yields adaptively secure pNIPEs under various assumptions such as DDH, LWE and *decision composite reciprocity* (DCR).

SM-2 Useful Definitions

SM-2.1 Inner Product Functional Encryption

The functional encryption scheme which deals with a particular type of functions that takes input two vectors and produces inner product of the vectors is called inner product functional encryption (IPFE).

Definition 5 (Stateful public-key inner product functional encryption) A stateful public-key inner product functional encryption (pIPFE) scheme for a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}$ and an inner product space $\mathcal{I}$ consists of four PPT algorithms $\mathsf{pIPFE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ satisfying the following requirement:

- $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pIPFE.Setup}(1^\lambda, 1^l)$: A trusted authority runs the setup algorithm taking inputs a security parameter λ, a vector length parameter l (a natural number that is a polynomial in λ) and outputs a master public-key MPK, a master secret-key MSK and an initial state st. The authority publishes MPK and keeps $\mathsf{MSK}, \mathsf{st}$ as secret.
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pIPFE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$: A predicate holder submits a vector $\boldsymbol{y} \in \mathcal{P}$ to an authority that runs the key generation algorithm providing inputs as a master public-key MPK, a master secret-key MSK, a vector $\boldsymbol{y}$ and outputs a secret key $\mathsf{sk}_{\boldsymbol{y}}$ corresponding to the predicate vector $\boldsymbol{y}$ and update the state st if required. The predicate holder gets its secret key $\mathsf{sk}_{\boldsymbol{y}}$ from the authority through a secure channel.
- $\mathsf{ct}_{\boldsymbol{x}} \leftarrow \mathsf{pIPFE.Enc}(\mathsf{MPK}, \boldsymbol{x})$: An encrypter runs the encryption algorithm that takes as input a master public-key MPK, an attribute vector $\boldsymbol{x} \in \mathcal{Q}$ and publishes the ciphertext $\mathsf{ct}_{\boldsymbol{x}}$ corresponding to the attribute $\boldsymbol{x}$.
- $\perp$ or $\zeta \leftarrow \mathsf{pIPFE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}})$: A decrypter runs the decryption algorithm taking as input a master public-key MPK, a secret-key $\mathsf{sk}_{\boldsymbol{y}}$, a ciphertext $\mathsf{ct}_{\boldsymbol{x}}$ and outputs either a message $\zeta \in \mathcal{I}$ or a symbol $\perp$ indicating failure.

Correctness: For any security parameter $\lambda, l(\lambda) \in \mathbb{N}, \boldsymbol{y} \in \mathcal{P}, \boldsymbol{x} \in \mathcal{Q}$, $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pIPFE.Setup}(1^\lambda, 1^l)$, $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pIPFE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$ we have

$$\Pr\big[\langle \boldsymbol{x}, \boldsymbol{y}\rangle = \mathsf{pIPFE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{pIPFE.Enc}(\mathsf{MPK}, \boldsymbol{x}))\big] = 1 - \mathsf{negl}(\lambda)$$

Definition 6 (Indistinguishability-based security for pIPFE) An inner product functional encryption scheme $\mathsf{pIPFE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ for a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}$ and an inner product space $\mathcal{I}$ is said to be adaptively secure under chosen-plaintext attacks ($\mathsf{IND\text{-}pIPFE}$) if, for any PPT adversary $\mathcal{A}$, for any $\lambda \in \mathbb{N}$, the advantage

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}pIPFE}}(\lambda) = \left|\Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{IND\text{-}pIPFE}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{IND\text{-}pIPFE}}(1^\lambda, 1) = 1]\right|$$

is negligible in λ where $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{IND\text{-}pIPFE}}(1^\lambda, b)$ is defined as follows:

1. **Setup phase:** The challenger generates $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pIPFESetup}(1^\lambda, 1^l)$, keeps $\mathsf{MSK}, \mathsf{st}$ as secret and sends MPK to $\mathcal{A}$.
2. **Query phase 1:** The adversary makes secret-key queries corresponding to predicate vectors $\boldsymbol{y} \in \mathcal{P}$. For each $\boldsymbol{y}$, the challenger computes $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pIPFEKeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$ and returns the secret-key $\mathsf{sk}_{\boldsymbol{y}}$ to $\mathcal{A}$.
3. **Challenge phase:** The adversary submits two distinct attribute vectors $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathcal{Q}$ with the restriction that $\langle \boldsymbol{x}_0, \boldsymbol{y}\rangle = \langle \boldsymbol{x}_1, \boldsymbol{y}\rangle$ holds for all predicate vectors $\boldsymbol{y}$ queried in step 2. The challenger then picks a random bit b and computes $\mathsf{ct}_{\boldsymbol{x}_b} \leftarrow \mathsf{pIPFEEnc}(\mathsf{MPK}, \boldsymbol{x}_b)$ which is sent as a challenge ciphertext to $\mathcal{A}$.
4. **Query phase 2:** The adversary may want to repeat **Query phase 1** for arbitrary predicate vectors $\boldsymbol{y} \in \mathcal{P}$ with the same constraint that $\langle \boldsymbol{x}_0, \boldsymbol{y}\rangle = \langle \boldsymbol{x}_1, \boldsymbol{y}\rangle$ for all $\boldsymbol{y}$.
5. **Guessing phase:** Finally, the adversary $\mathcal{A}$ outputs a guess bit b' which is the output of the experiment.

Remark 1 One can similarly define a *stateless* pIPFE scheme where the key generation algorithm is independent of state st. We note that most of the existing pIPFE schemes over $\mathbb{Z}_p$ are *stateful*[2,14]. Inner product functional encryption can be defined in private-key setting where encryption of an attribute is done using the master secret-key. The syntax is almost same as described in definition 2 except the master public-key is replaced by a public parameter pp and the encrypter uses the master secret-key MSK along with pp to produce ciphertext for an attribute.

Definition 7 (Stateless private-key inner product functional encryption) A stateless private-key inner product functional encryption (sIPFE) scheme for a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}$ and an inner product space $\mathcal{I}$ consists of PPT algorithms $\mathsf{sIPFE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ described below:

- $(\mathsf{pp}, \mathsf{MSK}) \leftarrow \mathsf{sIPFE.Setup}(1^\lambda, 1^l)$: A trusted authority runs the setup algorithm which takes as input a security parameter λ, a vector length parameter l (a natural number that is a polynomial in λ) and outputs a public parameter pp and a master secret-key MSK. The authority publishes pp and keeps MSK as secret.
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{sIPFE.KeyGen}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{y})$: A predicate holder submits a vector $\boldsymbol{y} \in \mathcal{P}$ to an authority that runs the key generation algorithm providing inputs as a public parameter pp, a master secret-key MSK, a vector $\boldsymbol{y}$ and output a secret key $\mathsf{sk}_{\boldsymbol{y}}$ corresponding to the predicate vector $\boldsymbol{y}$. The predicate holder gets its secret key $\mathsf{sk}_{\boldsymbol{y}}$ from the authority through a secure channel.
- $\mathsf{ct}_{\boldsymbol{x}} \leftarrow \mathsf{sIPFE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x})$: An encrypter runs the encryption algorithm that takes as input a public parameter pp, an attribute vector $\boldsymbol{x} \in \mathcal{Q}$ and publishes the ciphertext $\mathsf{ct}_{\boldsymbol{x}}$ corresponding to the attribute $\boldsymbol{x}$.
- $\perp$ or $\zeta \leftarrow \mathsf{sIPFE.Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}})$: A decrypter runs the decryption algorithm takes as input a public parameter pp, a secret-key $\mathsf{sk}_{\boldsymbol{y}}$, a ciphertext $\mathsf{ct}_{\boldsymbol{x}}$ and outputs either a message $\zeta \in \mathcal{I}$ or a symbol $\perp$.

Correctness: For any security parameter λ, $l(\lambda) \in \mathbb{N}$, $\boldsymbol{y} \in \mathcal{P}$, $\boldsymbol{x} \in \mathcal{Q}$, $(\mathsf{pp}, \mathsf{MSK}) \leftarrow$ $\mathsf{sIPFE.Setup}$
$(1^\lambda, 1^l)$, $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{sIPFE.KeyGen}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{y})$ we have

$$\Pr\big[\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \mathsf{sIPFE.Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{sIPFE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x}))\big] = 1 - \mathsf{negl}(\lambda)$$

Remark 2 One can similarly define indistinguishability-based security for the IPFE in private-key setting ($\mathsf{IND\text{-}sIPFE}$) where the only requirement is that the challenge ciphertext information theoretically hides the attributes. However, in many real-world applications it may happen that a decrypter only gets $\mathsf{sk}_{\boldsymbol{y}}$ from a predicate holder instead of $(\boldsymbol{y}, \mathsf{sk}_{\boldsymbol{y}})$ and the predicate holder wants the decrypter to learn only $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ as the predicate $\boldsymbol{y}$ may contain some sensitive information. So, the $\mathsf{IND\text{-}sIPFE}$ security model cannot fulfil such requirement of the predicate holder. The predicate hiding feature of an sIPFE was proposed by [37] and the notion is termed as *full-hiding security*. We adopt the definition of full-hiding security from [38].

Definition 8 (Full-hiding security for sIPFE) A private-key IPFE scheme is said to satisfy full-hiding security ($\mathsf{FH\text{-}sIPFE}$) if, for any PPT adversary $\mathcal{A}$, for any $\lambda \in \mathbb{N}$ the advantage

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{FH\text{-}sIPFE}}(\lambda) = \left| \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sIPFE}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sIPFE}}(1^\lambda, 1) = 1] \right|$$

is negligible in λ where $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sIPFE}}(1^\lambda, b)$ is defined as follows:

1. **Setup phase:** The challenger computes $(\mathsf{pp}, \mathsf{MSK}) \leftarrow \mathsf{sIPFE.Setup}(1^\lambda, 1^l)$, sends pp to $\mathcal{A}$ and keeps MSK as secret. It also chooses a random bit b.

2. **Query phase:** The adversary $\mathcal{A}$ may adaptively make polynomial number of queries that are of two types:
   - **Secret-key query:** For the j-th key query, $\mathcal{A}$ submits a pair of predicate vectors $(\boldsymbol{y}_0^{(j)}, \boldsymbol{y}_1^{(j)})$ and the challenger responses with $\mathsf{sk}_{\boldsymbol{y}_b^{(j)}} \leftarrow \mathsf{sIPFE.KeyGen}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{y}_b^{(j)})$.
   - **Ciphertext query:** For the ι-th ciphertext query, $\mathcal{A}$ transmits pair of attribute vectors $(\boldsymbol{x}_0^{(\iota)}, \boldsymbol{x}_1^{(\iota)})$ to which the challenger responses with $\mathsf{ct}_{\boldsymbol{x}_b^{(\iota)}} \leftarrow \mathsf{sIPFE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x}_b^{(\iota)})$.

   The queries made by $\mathcal{A}$ should satisfy $\langle \boldsymbol{x}_0^{(j)}, \boldsymbol{y}_0^{(\iota)} \rangle = \langle \boldsymbol{x}_1^{(j)}, \boldsymbol{y}_1^{(\iota)} \rangle$ for all j and ι.
3. **Guessing phase:** Finally, the adversary $\mathcal{A}$ outputs a guess bit b' which is the output of the experiment.

SM-2.2 Non-zero Inner Product Encryption

Non-zero inner product encryption (NIPE) is another type of functional encryption that encrypts a message M with an attribute $\boldsymbol{x}$. The ciphertext is decrypted to the message using a secret-key $\mathsf{sk}_{\boldsymbol{y}}$ corresponding to a predicate $\boldsymbol{y}$ if the inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$ is non-zero.

Definition 9 (Stateful public-key non-zero inner product encryption) A stateful public-key non-zero inner product functional encryption(pNIPE) scheme for a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}$, an inner product space $\mathcal{I}$ and a message space $\mathcal{M}$ consists of four PPT algorithms $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ operating as follows:

- $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^l)$: A trusted authority runs the setup algorithm which takes as input a security parameter λ, a vector length parameter l (a natural number that is a polynomial in λ) and outputs a master public-key MPK, a master secret-key MSK and an initial state st. The authority makes MPK public and keeps $\mathsf{MSK}, \mathsf{st}$ as secret.
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$: A predicate holder submits a vector $\boldsymbol{y} \in \mathcal{P}$ to an authority that runs the key generation algorithm providing inputs as a master public-key MPK, a master secret-key MSK, a vector $\boldsymbol{y}$ and outputs a secret key $\mathsf{sk}_{\boldsymbol{y}}$ corresponding to the predicate vector $\boldsymbol{y}$ and update the state st if required. The predicate holder gets its secret key $\mathsf{sk}_{\boldsymbol{y}}$ from the authority through a secure channel.
- $\mathsf{CT}_{\boldsymbol{x}} \leftarrow \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}, M)$: An encrypter runs the encryption algorithm that takes as input a master public-key MPK, an attribute vector $\boldsymbol{x} \in \mathcal{Q}$, a message $M \in \mathcal{M}$ and publishes the ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ corresponding to the attribute $\boldsymbol{x}$.
- $\perp$ or $\zeta \leftarrow \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A user runs the decryption algorithm that takes as input a master public-key MPK, a secret-key $\mathsf{sk}_{\boldsymbol{y}}$, a ciphertext $\mathsf{CT}_{\boldsymbol{x}}$, and outputs either a message $\zeta \in \mathcal{M}$ or a symbol $\perp$.

Correctness: For any security parameter λ, $l(\lambda) \in \mathbb{N}$, $\boldsymbol{y} \in \mathcal{P}$, $\boldsymbol{x} \in \mathcal{Q}$, $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^l)$, $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$ we have:

1. $\Pr\big[M = \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}, M)) : \langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0\big] = 1 - \mathsf{negl}(\lambda)$

2. $\Pr\big[\perp = \mathsf{pNIPE.Dec}(\mathsf{MPK}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}, M)) : \langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0\big] = 1 - \mathsf{negl}(\lambda)$

We define two adaptive security models for the pNIPE adopted from [34]. The first one establishes the indistinguishability of the encrypted messages under a challenge attribute (*payload-hiding*) and the second model describes the indistinguishability of the encrypted messages with different attributes (*attribute-hiding*) from the adversary's point of view.

Definition 10 (Adaptively payload-hiding security for pNIPE) A stateful non-zero inner product encryption scheme $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ for a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}$, an inner product space $\mathcal{I}$ and a message space $\mathcal{M}$ is said to follow adaptively payload-hiding security under chosen-plaintext attacks ($\mathsf{PH\text{-}pNIPE}$) if, for any PPT adversary $\mathcal{A}$, for any $\lambda \in \mathbb{N}$, the advantage

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{PH\text{-}pNIPE}}(\lambda) = \left|\Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{PH\text{-}pNIPE}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{PH\text{-}pNIPE}}(1^\lambda, 1) = 1]\right|$$

is negligible in λ, where $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{PH\text{-}pNIPE}}(1^\lambda, b)$ is defined as follows:

1. **Setup phase:** The challenger generates $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^l)$, makes $\mathsf{MSK}, \mathsf{st}$ as secret and sends MPK to $\mathcal{A}$.
2. **Query phase 1:** The adversary asks secret-key queries corresponding to predicate vectors $\boldsymbol{y} \in \mathcal{P}$. For each $\boldsymbol{y}$, the challenger returns $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y})$ to $\mathcal{A}$.
3. **Challenge phase:** The adversary submits two distinct messages $M_0, M_1 \in \mathcal{M}$ and a challenge attribute vectors $\boldsymbol{x}^* \in \mathcal{Q}$ with the restriction that $\langle \boldsymbol{x}^*, \boldsymbol{y}\rangle = 0$ in $\mathcal{I}$ for all predicate vectors $\boldsymbol{y}$ queried in step 2. The challenger computes $\mathsf{CT}_{\boldsymbol{x}^*}^{(b)} \leftarrow \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}^*, M_b)$ for a random bit b. The adversary receives $\mathsf{CT}_{\boldsymbol{x}^*}^{(b)}$ as a challenge ciphertext.
4. **Query phase 2:** The adversary can further query (as in **Query phase 1**) for secret-keys sk_y corresponding to any arbitrary predicate vector $\boldsymbol{y} \in \mathcal{P}$ with the same constraint that $\langle \boldsymbol{x}^*, \boldsymbol{y}\rangle = 0$ in $\mathcal{I}$.
5. **Guessing phase:** Finally, the adversary $\mathcal{A}$ outputs a guess bit b' which is the output of the experiment.

Definition 11 (Adaptively attribute-hiding security for pNIPE) A stateful non-zero inner product encryption scheme $\mathsf{pNIPE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ for a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}$, an inner product space $\mathcal{I}$ and a message space $\mathcal{M}$ is said to follow adaptively attribute-hiding security under chosen-plaintext attacks ($\mathsf{AH\text{-}pNIPE}$) if, for any PPT adversary $\mathcal{A}$, for any $\lambda \in \mathbb{N}$, the advantage

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{AH\text{-}pNIPE}}(\lambda) = \left|\Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{AH\text{-}pNIPE}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{AH\text{-}pNIPE}}(1^\lambda, 1) = 1]\right|$$

is negligible in λ where $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{AH\text{-}pNIPE}}(1^\lambda, b)$ is the same experiment as in Def. 10 with the only difference in the **Challenge phase** and **Query Phase 2** as stated below:

- **Challenge phase:** The adversary submits two attribute-message pairs $(\boldsymbol{x}_0, M_0)$, $(\boldsymbol{x}_1, M_1) \in \mathcal{Q} \times \mathcal{M}$ with the restriction that $\langle \boldsymbol{x}_0, \boldsymbol{y}\rangle = \langle \boldsymbol{x}_1, \boldsymbol{y}\rangle = 0$ if $M_0 \neq M_1$ or $\langle \boldsymbol{x}_0 - \boldsymbol{x}_1, \boldsymbol{y}\rangle = 0$ if $M_0 = M_1$, for all predicate vectors $\boldsymbol{y}$ queried in **Query phase 1**. The challenger computes $\mathsf{CT}_b \leftarrow \mathsf{pNIPE.Enc}(\mathsf{MPK}, \boldsymbol{x}_b, M_b)$ for a random bit b and returns CT_b to $\mathcal{A}$ as a challenge ciphertext.
- **Query phase 2:** The adversary may repeat **Query phase 1** for secret-keys sk_y corresponding to any arbitrary predicate vectors $\boldsymbol{y}$ that should satisfy the same constraint stated in the **Challenge phase** above.

As in the case of IPFE, we define full-hiding security model for NIPEs in private-key setting. A message is now encrypted using the master secret key and the master public-key is treated as a public parameter. Full-hiding security implies that knowing $\mathsf{sk}_{\boldsymbol{y}}$, $\mathsf{CT}_{\boldsymbol{x}}$ one can only infer $\langle \boldsymbol{x}, \boldsymbol{y}\rangle$ and recover the message if the inner product is non-zero. Specifically, no sensitive information about the predicate $\boldsymbol{y}$ and attribute $\boldsymbol{x}$ (other than $\langle \boldsymbol{x}, \boldsymbol{y}\rangle$) can be derived from the predicate secret-key and ciphertext.

Definition 12 (Stateless private-key non-zero inner product encryption) A stateless private-key non-zero inner product encryption (sNIPE) for a predicate space $\mathcal{P}$, an attribute space $\mathcal{Q}$ and an inner product space $\mathcal{I}$ consists of four probabilistic polynomial time (PPT) algorithms $\mathsf{sNIPE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ described below:

- $(\mathsf{pp}, \mathsf{MSK}) \leftarrow \mathsf{sNIPE.Setup}(1^\lambda, 1^l)$: A trusted authority runs the setup algorithm which takes as input a security parameter λ, a vector length parameter l (a natural number that is a polynomial in λ) and outputs a public parameter pp and a master secret-key MSK. The authority publishes pp and keeps MSK as secret.
- $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{sNIPE.KeyGen}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{y})$: A user submits a predicate vector $\boldsymbol{y} \in \mathcal{P}$ to an authority that runs the key generation algorithm with inputs a public parameter pp, a master secret-key MSK, a vector $\boldsymbol{y}$ and outputs a secret key $\mathsf{sk}_{\boldsymbol{y}}$ corresponding to the predicate vector $\boldsymbol{y}$. The predicate holder gets its secret key $\mathsf{sk}_{\boldsymbol{y}}$ from the authority through a secure channel.
- $\mathsf{CT}_{\boldsymbol{x}} \leftarrow \mathsf{sNIPE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x}, M)$: An encrypter runs the encryption algorithm that takes as input a public parameter pp, an attribute vector $\boldsymbol{x} \in \mathcal{Q}$, a message $M \in \mathcal{M}$ and outputs a ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ corresponding to the attribute $\boldsymbol{x}$.
- $\perp$ or $\zeta \leftarrow \mathsf{sNIPE.Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{CT}_{\boldsymbol{x}})$: A decrypter runs the decryption algorithm that takes as input a public parameter pp, a secret-key $\mathsf{sk}_{\boldsymbol{y}}$, a ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ and outputs either a message $\zeta \in \mathcal{I}$ or a symbol $\perp$.

Correctness: For any security parameter λ, $l(\lambda) \in \mathbb{N}$, $\boldsymbol{y} \in \mathcal{P}$, $\boldsymbol{x} \in \mathcal{Q}$, $(\mathsf{pp}, \mathsf{MSK}) \leftarrow \mathsf{sNIPE.Setup}$
$(1^\lambda, 1^l)$, $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{sNIPE.KeyGen}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{y})$ the following conditions hold:

1. $\Pr\big[M = \mathsf{sNIPE.Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{sNIPE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x}, M)) : \langle \boldsymbol{x}, \boldsymbol{y}\rangle \neq 0\big] = 1 - \mathsf{negl}(\lambda)$

2. $\Pr\big[\perp = \mathsf{sNIPE.Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{sNIPE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x}, M)) : \langle \boldsymbol{x}, \boldsymbol{y}\rangle = 0\big] = 1 - \mathsf{negl}(\lambda)$

Definition 13 (Full-hiding security for sNIPE) A private-key NIPE scheme is said to satisfy full-hiding security ($\mathsf{FH\text{-}sNIPE}$) if, for any PPT adversary $\mathcal{A}$, for any $\lambda \in \mathbb{N}$ the advantage

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{FH\text{-}sNIPE}}(\lambda) = \left|\Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sNIPE}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sNIPE}}(1^\lambda, 1) = 1]\right|$$

is negligible in λ, where $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sNIPE}}(1^\lambda, b)$ is defined as follows:

1. **Setup phase:** The challenger computes $(\mathsf{pp}, \mathsf{MSK}) \leftarrow \mathsf{sNIPE.Setup}(1^\lambda, 1^l)$, gives pp to $\mathcal{A}$ and keeps MSK as secret. It also chooses a random bit b.
2. **Query phase:** The adversary $\mathcal{A}$ may adaptively make polynomial number of queries that are of two types:
   - **Secret-key query:** For the j-th key query, $\mathcal{A}$ submits a pair of predicate vectors $(\boldsymbol{y}_0^{(j)}, \boldsymbol{y}_1^{(j)})$ and the challenger responses with $\mathsf{sk}_{\boldsymbol{y}_b^{(j)}} \leftarrow \mathsf{sNIPE.KeyGen}(\mathsf{pp},$ $\mathsf{MSK}, \boldsymbol{y}_b^{(j)})$.

```
Game $\boldsymbol{j}$, $j \in \{0,1,2\}$

1. The challenger gets $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^l)$ where $\mathsf{MSK} = (\boldsymbol{u} = (u_1, \ldots, u_l), \boldsymbol{v} = (v_1, \ldots, v_l))$ and $\mathsf{MPK} = (G, g, h, \{h_i = g^{u_i} h^{v_i}\}_{i \in [l]})$
2. The adversary selects $(\boldsymbol{x}^*, M_0, M_1) \leftarrow \mathcal{A}^{\mathsf{pNIPE.KeyGen(MPK,MSK,\cdot)}}(\mathsf{MPK})$
3. The challenger picks a random bit $b$ and encrypts the message $M_b$ with
   the challenge attribute $\boldsymbol{x}^* = (x_1, \ldots, x_l)$ as:
   3.1 If $j = 0, 1, 2,$        pick $r, t \hookleftarrow \mathbb{Z}_q$
   3.2 If $j = 0, 1,$           compute $\mathsf{C} = g^r$ and $\mathsf{D} = h^r$
       Else if $j = 2,$         compute $\mathsf{C} = g^r$ and $\mathsf{D} = h^{r+r'}$ for $r' \hookleftarrow \mathbb{Z}_q$
   3.3 If $j = 0, 1, 2,$        set $\mathsf{ct} = g^{M_b} \mathsf{D}^t$
   3.4 If $j = 0,$              compute $\mathsf{ct}_i = \mathsf{D}^{tx_i} h_i^r$ for $1 \leq i \leq l$
       Else if $j = 1, 2,$      compute $\mathsf{ct}_i = \mathsf{D}^{tx_i} \mathsf{C}^{u_i} \mathsf{D}^{v_i}$ for $1 \leq i \leq l$
   3.5 Return $\mathsf{CT}^{(b)}_{\boldsymbol{x}^*} = (\mathsf{G}, \mathsf{H}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i \in [l]})$
4. Finally, the adversary outputs $b' \leftarrow \mathcal{A}^{\mathsf{pNIPE.KeyGen(MPK,MSK,\cdot)}}(\mathsf{CT}^{(b)}_{\boldsymbol{x}^*})$
```

Fig. S-2: Sequence of Games used in the proof of Th. 2

- **Ciphertext query:** For the ι-th ciphertext query, $\mathcal{A}$ transmits two attribute-message pairs $(\boldsymbol{x}_0^{(\iota)}, M_0), (\boldsymbol{x}_1^{(\iota)}, M_1) \in \mathcal{Q} \times \mathcal{M}$ to the challenger which responses with $\mathsf{CT}_{\boldsymbol{x}_b^{(\iota)}} \leftarrow \mathsf{sNIPE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x}_b^{(\iota)}, M_b)$.

The queries made by $\mathcal{A}$ should satisfy $\langle \boldsymbol{x}_0^{(\iota)}, \boldsymbol{y}_0^{(j)} \rangle = \langle \boldsymbol{x}_1^{(\iota)}, \boldsymbol{y}_1^{(j)} \rangle = 0$ for all j, ι if $M_0 \neq M_1$ or $\langle \boldsymbol{x}_0^{(\iota)}, \boldsymbol{y}_0^{(j)} \rangle = \langle \boldsymbol{x}_1^{(\iota)}, \boldsymbol{y}_1^{(j)} \rangle$ for all j, ι if $M_0 = M_1$.

3. **Guessing phase:** Finally, the adversary $\mathcal{A}$ outputs a guess bit b' which is the output of the experiment.

SM-3 Proof of Theorem 2

Proof. We follow the same approach from the security proof of DDH based pIPFE scheme in [2]. We consider two sequence of games after the standard security experiment (Def. 10) named as Game 0. The above pNIPE is adaptively payload-hiding secure if the distinguishable gaps between the successive games are all negligible and the final game (Game 2) statistically hides challenge bit b into the ciphertext. We define Game j, $j \in \{0, 1, 2\}$, in Fig. S-2 and assume that all predicate key vectors $\boldsymbol{y} \in \mathcal{P}$ queried by $\mathcal{A}$ should satisfy $\langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0$ where $\boldsymbol{x}^*$ is the challenge attribute. Let E_j be the event $b = b'$ in Game j for $j = 0, 1, 2$.

Game 0 $\Rightarrow$ Game 1: In Game 1, the challenger directly uses $\mathsf{MSK} = (\boldsymbol{u}, \boldsymbol{v})$ to compute the ciphertext $\mathsf{CT}^{(b)}_{\boldsymbol{x}^*}$ as shown in Fig. S-2, item 3.4 when $j = 1$. Therefore, the ciphertext distributions in Game 0 and Game 1 are identical and it holds that $\Pr[E_0] = \Pr[E_1]$.

Game 1 $\Rightarrow$ Game 2: In Game 2, the challenger picks an extra randomness $r' \hookleftarrow \mathbb{Z}_q$ and sets $\mathsf{D} = h^{r+r'}$ (see Fig. S-2, item 3.2 when $j = 2$). Other components $(\mathsf{G}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i \in [l]})$ are computed in the same way as in Game 1. Since $h \in G$ and g is a generator of G, we write $\omega = \log_g h$. Consider the tuple $(h = g^\omega, \mathsf{C} = g^r, \mathsf{D})$ where the component D becomes: $g^{\omega r}$ in Game 1 and $g^{\omega(r+r')}$ in Game 2. The product $\omega r'$ is uniformly distributed modulo q as $\omega \in \mathbb{Z}_q^*$ and $r' \hookleftarrow \mathbb{Z}_q$. By DDH assumption, the tuple $(h, \mathsf{G}, \mathsf{H})$ is indistinguishable in transition from Game 1 to Game 2 and we have $|\Pr[E_1] - \Pr[E_2]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{DDH}}(\lambda)$ for any PPT adversary $\mathcal{B}$.

Now in Game 2, the ciphertext $\mathsf{CT}^{(b)}_{\boldsymbol{x}^*}$ becomes

$$(\mathsf{C} = g^r, \mathsf{D} = h^{r+r'}, \mathsf{ct} = g^{M_b} h^{t(r+r')}, \{\mathsf{ct}_i = h^{t(r+r')x_i + r'v_i} h_i^r\}_{i\in[l]})$$

Therefore, an unbounded adversary information theoretically infer

$$M_b + \omega \cdot t \cdot (r + r') \bmod q \quad \text{and} \quad \boldsymbol{z}_t = t \cdot (r + r') \cdot \boldsymbol{x}^* + r' \cdot \boldsymbol{v} \bmod q$$

from ct and $\{\mathsf{ct}_i\}_{i\in[l]}$, as $\{h_i\}_{i\in[l]}$ is part of MPK. The randomness r, r' are sampled uniformly and independently from $\mathbb{Z}_q$. So, $(r + r') = 0 \bmod q$ happens with all but a negligible probability as q is a μ-bit prime and $\mu \geq \lambda$. If $\boldsymbol{z}_t$ statistically hides $t \bmod q$, then $M_b + \omega \cdot t \cdot (r + r') \bmod q$ information theoretically hides b from $\mathcal{A}$'s view as t is distributed uniform modulo q.

We construct a matrix $\mathbf{Y}_{\mathsf{top}} \in \mathbb{Z}_q^{(l-1)\times l}$ using $\boldsymbol{x}^* \in \mathbb{Z}_q^l$ (as in Th. 3) such that the rows of it form a basis of the $(l-1)$-dimensional subspace

$$\boldsymbol{x}^{*\perp} = \{\boldsymbol{y} \in \mathbb{Z}_q^l : \langle \boldsymbol{x}^*, \boldsymbol{y}\rangle = 0 \bmod q\}$$

Let $\boldsymbol{y}' \in \mathbb{Z}_q^l \backslash \boldsymbol{x}^{*\perp}$ be any arbitrary vector easily computable to $\mathcal{A}$. The matrix defined by $\mathbf{Y} = \left[\frac{\mathbf{Y}_{\mathsf{top}}}{\boldsymbol{y}'}\right] \in \mathbb{Z}_q^{l\times l}$ is invertible modulo q as the rows form a basis of $\mathbb{Z}_q^l$. It is sufficient to prove that $\mathbf{Y} \cdot \boldsymbol{z}_t \in \mathbb{Z}_q^l$ statistically hides $t \bmod q$ from $\mathcal{A}$'s view. As $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{x}^* = \mathbf{0} \bmod q$, the first $(l-1)$ elements of $\mathbf{Y} \cdot \boldsymbol{z}_t$ is independent of t. We analyse the adversary's view on the last row of $\mathbf{Y} \cdot \boldsymbol{z}_t$ which is given by

$$\langle \boldsymbol{y}', \boldsymbol{z}_t\rangle = t \cdot (r + r') \cdot \langle \boldsymbol{x}^*, \boldsymbol{y}'\rangle + r' \cdot \langle \boldsymbol{v}, \boldsymbol{y}'\rangle \bmod q.$$

From the public-key $h_i = g^{u_i} h^{v_i} = g^{u_i + \omega v_i}, i \in [l]$, the adversary information theoretically gains $\boldsymbol{u} + \omega\boldsymbol{v} \bmod q$. The adversary can query keys corresponding to at most $(l-1)$ linearly independent vectors $\{\boldsymbol{y}_i\}_{i=1}^{l-1}$ contained in the subspace $\boldsymbol{x}^{*\perp}$ (other queries only supply redundant information). The predicate secret-key for a vector $\boldsymbol{y}_i$ is computed as $\mathsf{sk}_{\boldsymbol{y}_i} = (\langle \boldsymbol{u}, \boldsymbol{y}_i\rangle, \langle \boldsymbol{v}, \boldsymbol{y}_i\rangle)$. Suppose from the $\mathcal{A}$'s view, the joint distribution of the secret vectors $(\boldsymbol{u}, \boldsymbol{v}) \in \mathbb{Z}_q^l \times \mathbb{Z}_q^l$ be $(\boldsymbol{u}_{\mathcal{A}}, \boldsymbol{v}_{\mathcal{A}})$, then it must satisfy $\boldsymbol{u}_{\mathcal{A}} + \omega \cdot \boldsymbol{v}_{\mathcal{A}} = \boldsymbol{u}_0 + \omega \cdot \boldsymbol{v}_0 \bmod q$ where $\boldsymbol{u}_0 = (u_{0,1}, \ldots, u_{0,l}), \boldsymbol{v}_0 = (v_{0,1}, \ldots, v_{0,l}) \in \mathbb{Z}_q^l$ are arbitrary vectors such that $h_i = g^{u_{0,i}} h^{v_{0,i}}$ for all $i \in [l]$ and $\mathsf{sk}_{\boldsymbol{y}_i} = (\langle \boldsymbol{u}_0, \boldsymbol{y}_i\rangle, \langle \boldsymbol{v}_0, \boldsymbol{y}_i\rangle)$ for all $i \in [l-1]$. Since the predicate queries must satisfy $\langle \boldsymbol{x}^*, \boldsymbol{y}_i\rangle = 0$, the joint distribution of $(\boldsymbol{u}_{\mathcal{A}}, \boldsymbol{v}_{\mathcal{A}})$ can be expressed as

$$\{(\boldsymbol{u}_{\mathcal{A}} = \boldsymbol{u}_0 - \omega \cdot \gamma \cdot \boldsymbol{x}^* \bmod q, \boldsymbol{v}_{\mathcal{A}} = \boldsymbol{v}_0 + \gamma \cdot \boldsymbol{x}^* \bmod q) : \gamma \in \mathbb{Z}_q\}$$

Therefore, from $\mathcal{A}$'s view the distribution of $r' \cdot \langle \boldsymbol{v}, \boldsymbol{y}'\rangle \bmod q$ is

$$\{r' \cdot (\langle \boldsymbol{v}_0, \boldsymbol{y}'\rangle + \gamma \cdot \langle \boldsymbol{x}^*, \boldsymbol{y}'\rangle) \bmod q : \gamma \in \mathbb{Z}_q\}$$

As $\boldsymbol{y}'$ lies outside of $\boldsymbol{x}^{*\perp}$ we have $\langle \boldsymbol{x}^*, \boldsymbol{y}'\rangle \neq 0 \bmod q$ and $r' \hookleftarrow \mathbb{Z}_q$ implies $r' \neq 0 \bmod q$ with all but a negligible probability of $2^{-\mu}, \mu \geq \lambda$. Thus, $r' \cdot \langle \boldsymbol{v}, \boldsymbol{y}'\rangle \bmod q$ is uniformly distributed knowing public-keys and predicate key queries. Consequently, from adversary's view $t \bmod q$ is statistically hidden in $\langle \boldsymbol{y}', \boldsymbol{z}_t\rangle \bmod q$ with all but a negligible probability of $2^{-\lambda}$.

Therefore, $M_b + \omega \cdot t \cdot (r + r') \bmod q$ information theoretically hides the challenge bit b and it holds that $|\Pr[E_2] - \frac{1}{2}| \leq 2^{-\lambda}$. Adding all the indistinguishability gaps and using triangular inequality we have

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{PH-pNIPE}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{DDH}}(\lambda) + 2^{-\lambda}.$$

Game $\boldsymbol{j}$, $j \in \{0,1,2\}$

```
1. The challenger gets $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{pNIPE.Setup}(1^\lambda, 1^\mu, 1^l)$ where
   $\mathsf{MSK} = \boldsymbol{v} = (v_1, \ldots, v_l)$ and $\mathsf{MPK} = (p, \tilde{s}, f, g_p, G, F, \{h_i = g_p^{v_i}\}_{i \in [l]})$
2. The adversary selects $(\boldsymbol{x}^*, M_0, M_1) \leftarrow \mathcal{A}^{\mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \cdot)}(\mathsf{MPK})$
3. The challenger picks a random bit $b$ and encrypts the message $M_b$ with
   the challenge attribute $\boldsymbol{x}^* = (x_1, \ldots, x_l)$ as:
   3.1 If $j = 0, 1, 2$,        pick $r \hookleftarrow \mathcal{D}_{\mathbb{Z}, \sigma'}$ and $t \hookleftarrow \mathbb{Z}_p$
   3.2 If $j = 0, 1$,           compute $\mathsf{D} = g_p^r$
       Else if $j = 2$,         compute $\mathsf{D} = f^a g_p^r$ for $a \hookleftarrow \mathbb{Z}_p$
   3.3 If $j = 0, 1, 2$,        set $\mathsf{ct} = f^{M_b} \mathsf{D}^t$
   3.4 If $j = 0$,              compute $\mathsf{ct}_i = f^{t x_i} h_i^r$ for $1 \leq i \leq l$
       Else if $j = 1, 2$,      compute $\mathsf{ct}_i = f^{t x_i} \mathsf{D}^{v_i}$ for $1 \leq i \leq l$
   3.5 Return $\mathsf{CT}_{\boldsymbol{x}^*}^{(b)} = (\mathsf{D}, \mathsf{ct}, \{\mathsf{ct}_i\}_{i=1}^{l})$
4. Finally, the adversary outputs $b' \leftarrow \mathcal{A}^{\mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \cdot)}(\mathsf{CT}_{\boldsymbol{x}^*}^{(b)})$
```

Fig. S-3: Sequence of Games used in the proof of Th. 5

SM-4 Proof of Theorem 5

Proof. We consider the sequence games defined in Fig. S-3 to prove the payload-hiding security of the pNIPE. The standard payload-hiding security (Def. 10) for the above pNIPE is defined in Game 0. We end up with Game 2 where the challenge bit b remains statistically hidden from the adversary's point of view. The proof technique is borrowed from [14] and modified to make it work into our setting. In all the games, we assume that $\langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0$ for all predicate vectors $\boldsymbol{y}$ queried by the adversary $\mathcal{A}$ and $\boldsymbol{x}^* = (x_1, \ldots, x_l)$ is the challenge attribute. We define E_j to be the event $b = b'$ in the Game j, for $j = 0, 1, 2$.

Game 0 ⇒ Game 1: In Game 1, the challenger directly computes the ciphertexts using $\mathsf{MSK} = \boldsymbol{v}$ (see Fig. S-3, item 3.4 with $j = 1$). As a result, the distribution of ciphertexts are identical in both these games and we have $\Pr[E_0] = \Pr[E_1]$.

Game 1 ⇒ Game 2: The ciphertext component D is computed as $f^a g_p^r$ for $a \hookleftarrow \mathbb{Z}_p, r \hookleftarrow \mathcal{D}_{\mathbb{Z}, \sigma'}$ in Game 2 (see Fig. S-3, item 3.2 with $j = 2$) and the value of σ' is greater than $\tilde{s} \cdot \sqrt{\lambda}$. Thus, according to item 5 of Lem. 5 in Sec. 2.2, the distribution of D is within a statistical distance less than $2^{-\lambda}$ from the uniform distribution over G. Again in Game 1, item 3 of Lem. 5 implies that the distribution $\mathsf{D} = g_p^r$, $r \hookleftarrow \mathcal{D}_{\mathbb{Z}, \sigma'}$ is statistically close to uniform over G^p. Since the HSM assumption tells that it is hard to distinguish an element of G^p in the group G, we have $|\Pr[E_1] - \Pr[E_2]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{HSM}}(\lambda, \mu)$ for some PPT adversary $\mathcal{B}$.

Next, we show that the ciphertext of Game 2 statistically hides the challenge bit b, that is $|\Pr[E_2] - \frac{1}{2}| \leq 2^{-\lambda}$. The distribution of $\mathsf{CT}_{\boldsymbol{x}^*}^{(b)}$ can be expressed as

$$(\mathsf{D} = f^a g_p^r, \mathsf{ct} = f^{M_b + at} g_p^{rt}, \{\mathsf{ct}_i = f^{t x_i + a v_i} g_p^{r v_i}\}_{i=1}^{l})$$

The part of the ciphertext $\mathsf{D} = f^a g_p^r$ information theoretically reveals a mod p and r mod s, since $G = F \times G_p$ and the orders of the cyclic groups F, G_p are p, s respectively. An unbounded adversary may infer $M_b + a \cdot t$ mod p and rt mod s from the component $\mathsf{ct} = f^{M_b + at} g_p^{rt}$ of the ciphertext. The remaining part $\mathsf{ct}_i = f^{t x_i + a v_i} g_p^{r v_i}$ information theoretically discloses $z_{t,i} = t \cdot x_i + a \cdot v_i$ mod p as the term $g_p^{r v_i}$ is fixed by D and the public-key $h_i = g_p^{v_i}$, for all $i \in [l]$. Since $r \hookleftarrow \mathcal{D}_{\mathbb{Z}, \sigma'}$ with $\sigma' > \tilde{s} \cdot \sqrt{\lambda}$ and $t \hookleftarrow \mathbb{Z}_p$, the distribution of r mod s is statistically close to uniform modulo s and the product rt is distributed statistically close to uniform modulo n (using Lem. 5). Again $n = ps$

and $\gcd(p, s) = 1$ implies that the distributions of r, rt modulo s are independent from the distributions of r, rt modulo p. Thus, the adversary cannot get much information about t modulo p even if he knows r, rt modulo s. Therefore, given all the predicate key queries, if $\boldsymbol{z}_t = t \cdot \boldsymbol{x}^* + a \cdot \boldsymbol{v} \bmod p$ statistically hides $t \bmod p$, then $M_b + at$ modulo p hides the challenge bit b as a, t are both uniformly and independently sampled from $\mathbb{Z}_p$, and $at \neq 0 \bmod p$ with all but a negligible probability since p is a μ-bit prime with $\mu \geq \lambda$.

We show that $t \bmod p$ is statistically hidden within $\boldsymbol{z}_t = t \cdot \boldsymbol{x}^* + a \cdot \boldsymbol{v} \bmod p$ given all the information leaked to $\mathcal{A}$ via predicate key queries. All the predicate vectors $\{\boldsymbol{y}_i\}$ for which a predicate secret-key is given to the admissible adversary $\mathcal{A}$ must satisfy $\langle \boldsymbol{x}^*, \boldsymbol{y}_i \rangle = 0$ for all i, where $\boldsymbol{x}^*$ is the challenge attribute. In other words, all queried predicate vectors belong to the lattice $\boldsymbol{x}^{*\perp} = \{\boldsymbol{y} \in \mathbb{Z}^l : \langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0\}$. Let us assume that the first n_0 elements of $\boldsymbol{x}^*$ are zero and $\gcd(x_{n_0+1}, \ldots, x_l) = 1$.

Now, we borrow exactly the same matrix $\mathbf{Y} = \left[\frac{\mathbf{Y}_{\mathsf{top}}}{\mathbf{Y}_{\mathsf{bot}}}\right] \in \mathbb{Z}^{l \times l}$ used in the proof of Th. 3 (in Sec. 4.1), that is

$$\mathbf{Y}_{\mathsf{top}} = \left[\begin{array}{c|ccccc} I_{n_0} & & & & & \\ \hline & -x_{n_0+2} & x_{n_0+1} & & & \\ & & -x_{n_0+3} & x_{n_0+2} & & \\ & & & \ddots & \ddots & \\ & & & & -x_l & x_{l-1} \end{array}\right] \in \mathbb{Z}^{(l-1) \times l}$$

and $\mathbf{Y}_{\mathsf{bot}} = \boldsymbol{x}^{*T} \in \mathbb{Z}^{1 \times l}$. The matrix $\mathbf{Y}$ is invertible modulo p (by the arguments in Th. 3). Since $\mathbf{Y}$ is easily computable to $\mathcal{A}$, the information gained from $\boldsymbol{z}_t$ and $\mathbf{Y} \cdot \boldsymbol{z}_t \bmod p$ are equivalent from the adversary's perspective. Therefore, it is sufficient to analyze the information leaked from $\mathbf{Y} \cdot \boldsymbol{z}_t$ knowing all key queries made by $\mathcal{A}$. The structure of $\mathbf{Y}$ indicates $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{x}^* = 0$ as the rows of $\mathbf{Y}_{\mathsf{top}}$ belong to $\boldsymbol{x}^{*\perp}$ and form a basis of the lattice $\boldsymbol{x}^{*\perp}$. Consequently, $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{z}_t$ is independent of t. Thus we only need to concentrate on the last row of $\mathbf{Y} \cdot \boldsymbol{z}_t \bmod p$, given by $\langle \boldsymbol{x}^*, \boldsymbol{z}_t \rangle = t \cdot \|\boldsymbol{x}^*\|_2^2 + a \cdot \langle \boldsymbol{x}^*, \boldsymbol{v} \rangle \bmod p$. If we can show from the $\mathcal{A}$'s view that the distribution of $\langle \boldsymbol{x}^*, \boldsymbol{v} \rangle \bmod p$ (knowing the master public-key and the queried predicate secret-keys) is statistically close to the uniform distribution modulo p, then $\langle \boldsymbol{x}^*, \boldsymbol{z}_t \rangle \bmod p$ hides t with all but a negligible probability of $2^{-\lambda}$ as a is uniformly sampled from $\mathbb{Z}_p$ as p is a μ-bit prime with $\mu \geq \lambda$.

One can observe that all queried predicate vectors $\boldsymbol{y}_i$ can be expressed as the linear combination of the rows of $\mathbf{Y}_{\mathsf{top}}$, that is, $\boldsymbol{y}_i = \sum_{j=1}^{l-1} k_{i,j} \boldsymbol{R}_j^T$ where $k_{i,j} \in \mathbb{Z}$ and $\boldsymbol{R}_j$ is the jth row of $\mathbf{Y}_{\mathsf{top}}$. For $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v} = [r_1 \cdots r_{l-1}]^T \in \mathbb{Z}^{l-1}$ with $r_j = \langle \boldsymbol{v}, \boldsymbol{R}_j^T \rangle$, the secret-key corresponding to $\boldsymbol{y}_i$ is described as $\boldsymbol{sk}_{\boldsymbol{y}_i} = \langle \boldsymbol{v}, \boldsymbol{y}_i \rangle = \sum_{j=1}^{l-1} k_{i,j} r_j \in \mathbb{Z}$. Thus, the information obtained from predicate key queries is completely determined by $\mathbf{Y} \cdot \boldsymbol{v}$. Moreover, the master public-key component $h_i = g_p^{v_i}$, $1 \leq i \leq l$, information theoretically unveils $v_i \bmod s$. To examine the adversary's view on master secret-key, we consider an arbitrary vector $\boldsymbol{v}_0 = (v_{0,1}, \ldots, v_{0,l}) \in \mathbb{Z}^l$ such that

$$\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}_0 = \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v} \quad \text{and} \quad g_p^{v_{0,i}} = g_p^{v_i},\ \forall i \in [l].$$

Let us consider the lattice $\Lambda = \{\boldsymbol{\nu} \in \mathbb{Z}^l : \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{\nu} = \mathbf{0} \text{ and } \boldsymbol{\nu} \equiv \mathbf{0} (\text{mod } s)\}$. Therefore, in the adversary's eye, the master secret-key $\boldsymbol{v}$ which was sampled from $\mathcal{D}_{\mathbb{Z}^l, \sigma}$, is identical to $\boldsymbol{v}_0 + V$ where V is a random variable distributed as $\mathcal{D}_{\Lambda, \sigma, -\boldsymbol{v}_0}$. More precisely, form the adversary's view the master secret-key component $\boldsymbol{v}$ appears as $\boldsymbol{v}_0 + V$ where V is a random variable that takes values from Λ. The distribution of V is given by

$$\begin{aligned}\Pr[V = \boldsymbol{v}] &= \mathcal{D}_{\mathbb{Z}^l,\sigma,-\boldsymbol{v}_0}(\boldsymbol{v})/\mathcal{D}_{\mathbb{Z}^l,\sigma,-\boldsymbol{v}_0}(\Lambda)\\ &= \frac{\rho_{\sigma,-\boldsymbol{v}_0}(\boldsymbol{v})}{\rho_{\sigma,-\boldsymbol{v}_0}(\mathbb{Z}^l)}\cdot\frac{\rho_{\sigma,-\boldsymbol{v}_0}(\mathbb{Z}^l)}{\rho_{\sigma,-\boldsymbol{v}_0}(\Lambda)}\\ &= \rho_{\sigma,-\boldsymbol{v}_0}(\boldsymbol{v})/\rho_{\sigma,-\boldsymbol{v}_0}(\Lambda)\\ &= \mathcal{D}_{\Lambda,\sigma,-\boldsymbol{v}_0}\end{aligned}$$

We define the lattice $\Lambda^* = \{\boldsymbol{\nu} \in \mathbb{Z}^l : \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{\nu} = \mathbf{0}\}$. This is a 1-dimension lattice as the rows of $\mathbf{Y}_{\mathsf{top}}$ are linearly independent over $\mathbb{Z}$. Furthermore, $\boldsymbol{x}^* \in \Lambda^*$ and it implies that $\Lambda^* = \boldsymbol{x}^* \cdot \mathbb{Z}$ since the non-zero co-ordinates of $\boldsymbol{x}^*$ are co-prime to each other. Therefore, one can rewrite

$$\Lambda = \Lambda^* \cap s \cdot \mathbb{Z}^l = (\boldsymbol{x}^* \cdot \mathbb{Z}) \cap (s \cdot \mathbb{Z}^l) = s \cdot \boldsymbol{x}^* \cdot \mathbb{Z}$$

Applying Lem. 2 of Sec. 2.1 and denoting $\Lambda_0 = s \cdot \|\boldsymbol{x}^*\|_2^2 \cdot \mathbb{Z}$, the distribution of $\langle \boldsymbol{x}^*, \boldsymbol{v}\rangle$ is given by

$$\langle \boldsymbol{x}^*, \boldsymbol{v}_0\rangle + \mathcal{D}_{\Lambda_0,\sigma\cdot\|\boldsymbol{x}^*\|_2,-\langle \boldsymbol{x}^*,\boldsymbol{v}_0\rangle}$$

To show that the distribution of $\langle \boldsymbol{x}^*, \boldsymbol{v}\rangle \bmod p$ is statistically close to uniform distribution over $\mathbb{Z}_p$, we consider the distribution $\mathcal{D}_{\Lambda_0,\sigma\cdot\|\boldsymbol{x}^*\|_2,-\langle \boldsymbol{x}^*,\boldsymbol{v}_0\rangle}$ over Λ_0 modulo the sublattice $\Lambda_0' = p\Lambda_0$ as $\Lambda_0/\Lambda_0' = \mathbb{Z}_p$. Now, from Lem. 3 of Sec. 2.1, with $\epsilon = 2^{-\lambda-1}$, the reduced distribution $\mathcal{D}_{\Lambda_0,\sigma\cdot\|\boldsymbol{x}^*\|_2,-\langle \boldsymbol{x}^*,\boldsymbol{v}_0\rangle} \bmod \Lambda_0'$ is within a statistical distance $2^{-\lambda}$ from the uniform distribution over $\Lambda_0/\Lambda_0' \simeq \mathbb{Z}_p$ if $\sigma \cdot \|\boldsymbol{x}^*\|_2 > \eta_\epsilon(\Lambda_0')$. The minimum distance of the lattice Λ_0' is $\lambda_1(\Lambda_0') = p \cdot s \cdot \|\boldsymbol{x}^*\|_2^2 < p \cdot \tilde{s} \cdot \|\boldsymbol{x}^*\|_2^2$ which can be used in Lem. 4 of Sec. 2.1 to get an upper bound as

$$\eta_\epsilon(\Lambda_0') \le \sqrt{\tfrac{\ln(2n(1+1/\epsilon))}{\pi}} \cdot \lambda_1(\Lambda_0') < \sqrt{\lambda} \cdot \lambda_1(\Lambda_0') = \sqrt{\lambda} \cdot p \cdot \tilde{s} \cdot \|\boldsymbol{x}^*\|_2^2.$$

Choosing $\sigma \cdot \|\boldsymbol{x}^*\|_2 > \sqrt{\lambda} \cdot p \cdot \tilde{s} \cdot \|\boldsymbol{x}^*\|_2^2$ and using the fact $\|\boldsymbol{x}^*\|_2 < \sqrt{l} \cdot X < \sqrt{p}$ we finally set $\sigma > p^{3/2} \cdot \tilde{s} \cdot \sqrt{\lambda}$ to ensure that the distribution $\langle \boldsymbol{x}^*, \boldsymbol{v}\rangle \bmod p$ is within a distance $2^{-\lambda}$ from the uniform distribution over $\mathbb{Z}_p$. As discussed earlier, this implies $t \bmod p$ is statistically hidden within $\boldsymbol{z}_t = t \cdot \boldsymbol{x}^* + a \cdot \boldsymbol{v} \bmod p$ and it holds that $|\Pr[E_2] - \frac{1}{2}| \le 2^{-\lambda}$.

Adding up all the probabilities obtained in the above sequence of games and using triangular inequality, we have

$$\begin{aligned}\mathsf{Adv}_{\mathcal{A}}^{\mathsf{PH-pNIPE}}(\lambda) &= |\Pr[E_0] - \frac{1}{2}|\\ &\le |\Pr[E_0] - \Pr[E_1]| + |\Pr[E_1] - \Pr[E_2]| + |\Pr[E_2] - \frac{1}{2}|\\ &\le \mathsf{Adv}_{\mathcal{B}}^{\mathsf{HSM}}(\lambda, \mu) + 2^{-\lambda}\end{aligned}$$

which is negligible in λ according to the hypothesis that HSM problem is hard in G.

SM-5 Proof of Theorem 6

Proof. The proof begins with the same sequence of games described in Fig. S-3 as the encryption algorithm of the pNIPE is exactly the same as that of the pNIPE in Sec. 5.1 over $\mathbb{Z}$. The key extraction oracle is now stateful and is taken as $\mathsf{pNIPE.KeyGen}(\mathsf{MPK},$

$\mathsf{MSK}, \mathsf{st}, \cdot)$. To maintain this difference we rename Game j and event E_j of Th. 5 by Game j' and E'_j respectively. We note that for any predicate vector $\boldsymbol{y}$ queried by the adversary $\mathcal{A}$ it holds that $\langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0 \bmod p$ where $\boldsymbol{x}^* = (x_1, \ldots, x_l)$ is the challenge attribute.

As explained in Th. 5, Game $0'$ is identical to Game $1'$ and Game $1'$ is indistinguishable from Game $2'$ under the HSM assumption. Formally it is expressed as

$$\Pr[E'_0] = \Pr[E'_1] \text{ and } |\Pr[E'_1] - \Pr[E'_2]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{HSM}}(\lambda, \mu)$$

Recall that the ciphertext $\mathsf{CT}_{\boldsymbol{x}^*}^{(b)}$ in Game $2'$ is given by

$$(\mathsf{D} = f^a g_p^r, \mathsf{ct} = f^{M_b + at} g_p^{rt}, \{\mathsf{ct}_i = f^{tx_i + av_i} g_p^{rv_i}\}_{i=1}^{l})$$

and our aim is to show that the challenge bit b is statistically hidden within it from the adversary's view, that is $|\Pr[E'_2] - \frac{1}{2}| \leq 2^{-\lambda}$. From the discussion in the original Game 2, essentially the challenge ciphertext

$$M_b + a \cdot t \bmod p \;\text{ and }\; \boldsymbol{z}_t = t \cdot \boldsymbol{x}^* + a \cdot \boldsymbol{v} \bmod p$$

in information theoretic sense and $M_b + a \cdot t \bmod p$ hides b if $\boldsymbol{z}_t$ succeeds in hiding $t \bmod p$ given the master public-key and predicate key queries, as a, t are uniformly and independently sampled from $\mathbb{Z}_p$.

We make use of the fact that if $\mathbf{Y}$ is an invertible matrix modulo p available to $\mathcal{A}$, then $\mathbf{Y} \cdot \boldsymbol{z}_t$ and $\boldsymbol{z}_t \bmod p$ leak similar information about $t \bmod p$. Here, one must not borrow the same $\mathbf{Y}$ from Th. 5 as $\det(\mathbf{Y}\mathbf{Y}^T)$ is not guaranteed to be non-zero modulo p. Instead, we use the approach of Th. 4 to construct $\mathbf{Y}$ from a full fledge basis of $\mathbb{Z}_p^l$.

Without loss of generality, we assume that the adversary queried for at most $(l-1)$ predicate keys corresponding to the vectors $\{\boldsymbol{y}_i\}_{i=1}^{l-1}$ which is a linearly independent set over $\mathbb{Z}_p$. Our aim is to show that $t \bmod p$ remains statistically hidden from the view of adversary after it makes j predicate key queries, for any $j \in \{0, 1, \ldots, l-1\}$. We apply induction on the number of predicate key queries made by $\mathcal{A}$. If no predicate key query occurs, then the advantage of $\mathcal{A}$ in Game $2'$ is the same as it was in the original Game 2 of Th. 5. Hence, the induction hypothesis is true for $j = 0$. Next, we show that for any $j \in \{1, \ldots, l-1\}$, $\mathcal{A}$'s view is statistically independent of $t \bmod p$.

Let us fix some $j \in \{1, \ldots, l-1\}$ and assume that, at this point $\mathsf{st} = \{(\boldsymbol{y}_i, \mathsf{sk}_{\boldsymbol{y}_i} = (\overline{\mathbf{y}}_i, s_{\overline{\mathbf{y}}_i}))\}_{i \in [j]}$, is independent of $t \bmod p$. If $j < (l-1)$ then the set $\{\overline{\mathbf{y}}_i\}_{i \in [j]}$ can be deterministically extended to a set $\{\overline{\mathbf{y}}_i\}_{i \in [l-1]}$ so as to form a basis of the $(l-1)$-dimensional subspace

$$\boldsymbol{x}^{*\perp p} = \{\boldsymbol{y} \in \mathbb{Z}_p^l : \langle \boldsymbol{x}^*, \boldsymbol{y} \rangle = 0 \bmod p\}.$$

This can be thought of predicate keys queried by the challenger for dummy vectors $\{\boldsymbol{y}_i\}_{i=j+1}^{l-1}$ and then use the outputs $\{\mathsf{pNIPE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \boldsymbol{y}_i)\}_{i=j+1}^{l-1}$ to get a full basis. We construct a matrix $\mathbf{Y}_{\mathsf{top}} \in \mathbb{Z}^{(l-1) \times l}$ by setting its i-th row as $\overline{\mathbf{y}}_i$ for each $i \in [l-1]$. We find $\boldsymbol{y}' \in \mathbb{Z}_p^l \setminus \boldsymbol{x}^{*\perp p}$ such that it is efficiently computable to $\mathcal{A}$ and set $\mathbf{Y}_{\mathsf{bot}} = \overline{\mathbf{y}}'$ to be the canonical lift of $\boldsymbol{y}'$ over the integers. Therefore, the matrix $\mathbf{Y} = \left[\frac{\mathbf{Y}_{\mathsf{top}}}{\mathbf{Y}_{\mathsf{bot}}}\right] \in \mathbb{Z}^{l \times l}$ is invertible modulo p.

One can observe that $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{z}_t$ is independent of t as each row of $\mathbf{Y}_{\mathsf{top}}$ is orthogonal to $\boldsymbol{x}^*$. Thus, only the last row of $\mathbf{Y} \cdot \boldsymbol{z}_t$ may contain some information about $t \bmod p$ that is useful to $\mathcal{A}$. In continuation to the above discussion, it is sufficient to demonstrate that

$$\mathbf{Y}_{\mathsf{bot}} \cdot \boldsymbol{z}_t = t \cdot \langle \overline{\mathbf{y}}', \boldsymbol{x}^* \rangle + a \cdot \langle \overline{\mathbf{y}}', \boldsymbol{v} \rangle \bmod p$$

statistically hides t mod p from $\mathcal{A}$'s view. In particular, if the distribution of $\langle \overline{\mathbf{y}}', \boldsymbol{v} \rangle$ mod p is statically close to uniform distribution modulo p, then $\mathbf{Y}_{\mathsf{bot}} \cdot \boldsymbol{z}_t$ statistically hides t mod p since $a \hookleftarrow \mathbb{Z}_p$ implying $a \neq 0$ with all but a negligible probability as p is a μ-bit prime with $\mu \geq \lambda$.

As discussed in the proof of Th. 5, the information obtained via predicate key queries can be completely redefined by $\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}$. Also, from the master public-key component $h_i = g_p^{v_i}$, for $1 \leq i \leq l$, the adversary information theoretically learns v_i mod s. Let us take an arbitrary vector $\boldsymbol{v}_0 = (v_{0,1}, \ldots, v_{0,l}) \in \mathbb{Z}^l$ such that

$$\mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v}_0 = \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{v} \quad \text{and} \quad g_p^{v_{0,i}} = g_p^{v_i}, \ \forall i \in [l].$$

Conditionally on $\mathcal{A}$'s view, the distribution of $\boldsymbol{v} \in \mathbb{Z}^l$ is $\boldsymbol{v}_0 + \mathcal{D}_{\Lambda, \sigma, -\boldsymbol{v}_0}$ where

$$\Lambda = \{\boldsymbol{\nu} \in \mathbb{Z}^l : \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{\nu} = \mathbf{0} \text{ and } \boldsymbol{\nu} \equiv \mathbf{0} \pmod{s}\}.$$

We also define an 1-dimensional lattice $\Lambda^* = \{\boldsymbol{\nu} \in \mathbb{Z}^l : \mathbf{Y}_{\mathsf{top}} \cdot \boldsymbol{\nu} = \mathbf{0}\}$ in $\mathbb{Z}^l$. Since $\boldsymbol{x}^* \in \Lambda^*$, the lattice can be expressed as $\Lambda^* = \boldsymbol{x}' \cdot \mathbb{Z}$ for some $\boldsymbol{x}' = \gamma \cdot \boldsymbol{x}^*$ mod p where $\gamma \in (\mathbb{Z}_p)^*$ is chosen to make the co-ordinates of $\boldsymbol{x}'$ co-prime to each other. One now observes that $\Lambda = \Lambda^* \cap s \cdot \mathbb{Z}^l = s \cdot \boldsymbol{x}' \cdot \mathbb{Z}$. Let Λ^*_{top} be the lattice generated by the rows of $\mathbf{Y}_{\mathsf{top}}$. Note that Λ^* is orthogonal to Λ^*_{top}. Then by the property of orthogonal lattices [32] and applying Lem. 1 of Sec. 2.1 we get

$$\|\boldsymbol{x}'\|_2 = \det(\Lambda^*) \leq \det(\Lambda^*_{\mathsf{top}}) \leq \prod_{i=1}^{l-1} \|\overline{\mathbf{y}}_i\|_2 \leq (\sqrt{l} \cdot p)^{l-1}.$$

Next, we take a sublattice $\Lambda' = p\Lambda$. Utilizing Lem. 3 of Sec. 2.1, the reduced distribution $\mathcal{D}_{\Lambda, \sigma, -\boldsymbol{v}_0}$ mod Λ' is within a statistical distance of at most 2ϵ from the uniform distribution over Λ mod Λ' if $\sigma > \eta_\epsilon(\Lambda')$. With $\epsilon = 2^{-\lambda-1}$, we use Lem. 4 of Sec. 2.1 and the bound on $\|\boldsymbol{x}'\|_2$ derived above to get another bound on the smoothing parameter of Λ' as

$$\begin{aligned} \eta_\epsilon(\Lambda') &\leq \sqrt{\frac{\ln(2n(1 + 1/\epsilon))}{\pi}} \cdot \lambda_1(\Lambda') \\ &\leq \sqrt{\lambda} \cdot \lambda_1(\Lambda') = \sqrt{\lambda} \cdot p \cdot s \cdot \|\boldsymbol{x}'\|_2 \\ &\leq \sqrt{\lambda} \cdot p^l \cdot \tilde{s} \cdot (\sqrt{l})^{l-1} \end{aligned}$$

By setting $\sigma > \tilde{s} \cdot \sqrt{\lambda} \cdot p^l \cdot (\sqrt{l})^{l-1}$, the distribution $\mathcal{D}_{\Lambda, \sigma, -\boldsymbol{v}_0}$ mod Λ' becomes $2^{-\lambda}$-close to the uniform distribution over $\Lambda / \Lambda' \simeq \boldsymbol{x}' \cdot \mathbb{Z}_p$ as $\gcd(s, p) = 1$. Again $\boldsymbol{x}' \cdot \mathbb{Z}_p = \gamma \cdot \boldsymbol{x}^* \cdot \mathbb{Z}_p \simeq \boldsymbol{x}^* \cdot \mathbb{Z}_p$ as γ is a non-zero element of $\mathbb{Z}_p$. Consequently, in $\mathcal{A}$'s view the distribution of $\boldsymbol{v}$ mod p is within a statistical distance of $2^{-\lambda}$ from the uniform distribution over $\boldsymbol{x}^* \cdot \mathbb{Z}_p$. So, $\mathcal{A}$ would write $\boldsymbol{v} = \beta \cdot \boldsymbol{x}^*$ where β is uniformly chosen from $\mathbb{Z}_p$ and hence $\langle \overline{\mathbf{y}}', \boldsymbol{v} \rangle = \beta \cdot \langle \overline{\mathbf{y}}', \boldsymbol{x}^* \rangle$ mod p becomes uniformly distributed over $\mathbb{Z}_p$ in $\mathcal{A}$'s view as $\langle \overline{\mathbf{y}}', \boldsymbol{x}^* \rangle$ mod p is non-zero.

Therefore, $\boldsymbol{z}_t$ statistically hides t mod p from the adversary's view. In other words, the challenge bit b is statistically hidden in $M_b + a \cdot t$ mod p with all but a negligible probability of $2^{-\lambda}$ and it holds that $|\Pr[E_2] - \frac{1}{2}| \leq 2^{-\lambda}$.

Finally, adding up all the probability gaps using triangular inequality, one obtains

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{PH-pNIPE}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{HSM}}(\lambda, \mu) + 2^{-\lambda}$$

which is negligible in λ if the HSM problem is hard in G.

SM-6 LWE and DCR based AH-pNIPE

LWE **based constructions.** Agrawal et al. [2] gave two pIPFE constructions from the (multi-hint extended) LWE assumption. Their first pIPFE with inner products over $\mathbb{Z}$ is stateless and the second pIPFE with inner products over $\mathbb{Z}_p$ is stateful. Employing these pIPFEs in our generic construction, we obtain two adaptively attribute-hiding pNIPEs based on the LWE assumption: one is stateless with inner products over $\mathbb{Z}$ and the other is stateful computing inner products over $\mathbb{Z}_p$. For our stateless pNIPE over $\mathbb{Z}$ the domains are taken as $\mathcal{I} = \mathbb{Z}$, $\mathcal{P} = \{0, 1, \cdots, \mathfrak{B}_y - 1\}$, $\mathcal{M} = \{0, 1, \cdots, \mathfrak{B} - 1\}$, $\mathcal{Q} = \{0, 1, \cdots, \mathfrak{B}_x - 1\}$, $\mathcal{Q}' = \{0, 1, \cdots, \mathfrak{B} \cdot \mathfrak{B}_x - 1\}$ where $\mathfrak{B}_x, \mathfrak{B}_y$, and $\mathfrak{B}$ are polynomially bounded integers. In case of our stateful pNIPE, we consider the domains as $\mathcal{I} = \mathbb{Z}_p$, $\mathcal{P} = \mathcal{Q} = \mathcal{Q}' = \mathbb{Z}_p^l$ and $\mathcal{M} = \mathbb{Z}_p$. The choice of parameters required in the underlying pIPFEs can be taken from [2] for the correctness and security of our schemes.

DCR **based constructions.** Agrawal et al. [2] constructed two pIPFEs from the DCR assumption where the first one is stateless with inner products over $\mathbb{Z}$ and the second is stateful with inner products over $\mathbb{Z}_N$. Here, N is a product of two exponentially large safe primes. To apply the first pIPFE of [2] in our generic transformation, we take the domains as $\mathcal{I} = \mathbb{Z}$, $\mathcal{P} = \mathcal{Q}' = \{\boldsymbol{y} \in \mathbb{Z}^l : \|\boldsymbol{y}\|_\infty < (N/2l)^{1/2}\}$, $\mathcal{M} = \{M \in \mathbb{Z} : |M| < \mathfrak{B}\}$, $\mathcal{Q} = \{\boldsymbol{x} \in \mathbb{Z}^l : \|\boldsymbol{x}\|_\infty < (N/2l\mathfrak{B}^2)^{1/2}\}$ where $\mathfrak{B}$ is an integer (possibly exponentially large). This results in an adaptively attribute-hiding stateless pNIPE with inner products over $\mathbb{Z}$ under the DCR assumption. One may consider $\mathcal{I} = \mathbb{Z}_N$, $\mathcal{P} = \mathcal{Q} = \mathcal{Q}' = \mathbb{Z}_N^l$ and $\mathcal{M} = \mathbb{Z}_N$ to get a stateful pNIPE from the second pIPFE. Note that, this conversion works fine if we treat $\mathbb{Z}_N$ as an integral domain which is sensible as getting a zero divisor in the ring $\mathbb{Z}_N$ leads to a factorization of N and hence breaks the DCR problem.

SM-7 Proof of Theorem 8

Proof. We prove this by contradiction, that is, we assume that there exists a PPT adversary $\mathcal{A}$ for the sNIPE such that the advantage in the security experiment in Def. 13 is non-negligible. We construct another PPT adversary $\mathcal{B}$ against the full-hiding security of sIPFE (see Def. 8) using the sNIPE adversary as follows:

1. **Setup:** The sIPFE challenger generates $(\mathsf{pp}, \mathsf{MSK}) \leftarrow \mathsf{sIPFE.Setup}(1^\lambda, 1^l)$ and sends pp to $\mathcal{B}$ that passes it to $\mathcal{A}$. The challenger makes MSK secret and chooses a random bit b.
2. **Query phase:** The adversary may ask the following two types of query:
   - **Secret-key query:** On the j-th predicate key query, $\mathcal{A}$ submits a pair of vectors $(\boldsymbol{y}_0^{(j)}, \boldsymbol{y}_1^{(j)}) \in \mathcal{P} \times \mathcal{P}$ which $\mathcal{B}$ forwards to its challenger to get the secret-key corresponding to the challenge bit. The challenger returns $\mathsf{sk}_{\boldsymbol{y}_b^{(j)}} \leftarrow \mathsf{sIPFE.KeyGen}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{y}_b^{(j)})$. Then $\mathcal{B}$ passes $\mathsf{sk}_{\boldsymbol{y}_b^{(j)}}$ to $\mathcal{A}$.
   - **Ciphertext query:** The adversary $\mathcal{A}$ submits two pairs of attribute vector, message $(\boldsymbol{x}_0^{(\iota)}, M_0), (\boldsymbol{x}_1^{(\iota)}, M_1) \in \mathcal{Q} \times \mathcal{M}$ as its ι-th ciphertext query. Then, $\mathcal{B}$ makes two pair of challenge attribute vectors $(\boldsymbol{x}_0^{(\iota)}, \boldsymbol{x}_1^{(\iota)}), (M_0 \cdot \boldsymbol{x}_0^{(\iota)}, M_1 \cdot \boldsymbol{x}_1^{(\iota)}) \in \mathcal{Q}'$ and sends these (once at a time) to its challenger. The sIPFE challenger returns $\mathsf{ct}_{1,\iota}^{(b)} \leftarrow \mathsf{sIPFE.Enc}(\mathsf{pp}, \mathsf{MSK}, \boldsymbol{x}_b^{(\iota)})$ and $\mathsf{ct}_{2,\iota}^{(b)} \leftarrow \mathsf{sIPFE.Enc}(\mathsf{pp}, \mathsf{MSK}, M_b \cdot \boldsymbol{x}_b^{(\iota)})$ and $\mathcal{B}$ sends $\mathsf{CT}_\iota^{(b)} = (\mathsf{ct}_{1,\iota}^{(b)}, \mathsf{ct}_{2,\iota}^{(b)})$ as the ι-th challenge ciphertext to $\mathcal{A}$.
3. **Guessing phase:** Finally, $\mathcal{B}$ returns a guess b' which is the output of $\mathcal{A}$.

We note that the total number of ciphertexts queried by $\mathcal{B}$ is twice the number of ciphertext queries made by $\mathcal{A}$. So, $\mathcal{B}$ makes a polynomial number of queries to simulate $\mathcal{A}$. As in the proof of Th. 7, the predicates and attributes satisfy $\langle \boldsymbol{x}_0^{(\iota)}, \boldsymbol{y}_0^{(j)} \rangle = \langle \boldsymbol{x}_1^{(\iota)}, \boldsymbol{y}_1^{(j)} \rangle$ and $\langle M_0 \cdot \boldsymbol{x}_0^{(\iota)}, \boldsymbol{y}_0^{(j)} \rangle = \langle M_1 \cdot \boldsymbol{x}_1^{(\iota)}, \boldsymbol{y}_1^{(j)} \rangle$ for all j, ι. Thus, $\mathcal{B}$ is an admissible adversary for $\mathsf{FH\text{-}sIPFE}$ security model.

If the sIPFE challenger chooses $b = 0$, then $\mathcal{B}$ simulates $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sNIPE}}(1^\lambda, 0)$ and if $b = 1$, then it simulates $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{FH\text{-}sNIPE}}(1^\lambda, 1)$. Thus, the advantage of $\mathcal{B}$ in $\mathsf{FH\text{-}sIPFE}$ experiment is the same as that of $\mathcal{A}$ in $\mathsf{FH\text{-}sNIPE}$. Hence, the proof follows.

Remark 3 In our generic construction of NIPEs from IPFEs, one can observe that a decrypter having a secret-key $\mathsf{sk}_{\boldsymbol{y}}$ associated to a predicate vector $\boldsymbol{y}$ and a ciphertext $\mathsf{CT}_{\boldsymbol{x}} = (\mathsf{ct}_{\boldsymbol{x}}, \mathsf{ct}_{M \cdot \boldsymbol{x}})$ corresponding to an attribute $\boldsymbol{x}$ learns $\eta = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ via IPFE decryption. If $\eta \neq 0$, then the message M can be recovered. When $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ contains sensitive information, it becomes vulnerable to the decrypter. Therefore, it is necessary to hide $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ inside η so that the decrypter only learns whether the inner product is zero or not. There are two simple ways to achieve this—one can either randomize the predicate vector while creating the secret-key or randomize the attribute vector while encryption. In the first case, a secret-key associated to a predicate vector $\boldsymbol{y}$ is obtained as $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{IPFE.KeyGen}(\cdot, \mathsf{MSK}, t \cdot \boldsymbol{y})$ where t is uniform over $\mathcal{I} \setminus \{0\}$. In the latter case, the encrypter can select t uniformly at random from $\mathcal{I} \setminus \{0\}$ and computes the components of the ciphertext as $\mathsf{ct}'_{\boldsymbol{x}} \leftarrow \mathsf{IPFE.Enc}(\cdot, t \cdot \boldsymbol{x})$, $\mathsf{ct}'_{M \cdot \boldsymbol{x}} \leftarrow \mathsf{IPFE.Enc}(\cdot, tM \cdot \boldsymbol{x})$. So, the decrypter gets $\eta = t \cdot \langle \boldsymbol{x}, \boldsymbol{y} \rangle \leftarrow \mathsf{IPFE.Dec}(\cdot, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}'_{\boldsymbol{x}})$ and $\eta' = tM \cdot \langle \boldsymbol{x}, \boldsymbol{y} \rangle \leftarrow \mathsf{IPFE.Dec}(\cdot, \mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}'_{M \cdot \boldsymbol{x}})$. Then, if $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \neq 0$ it computes η'/η to recover the message M, otherwise it returns $\perp$. In this scheme, the decrypter learns $t \cdot \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ instead of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ where $t \hookleftarrow \mathcal{I} \setminus \{0\}$. Note that, t can be chosen from a sufficiently small interval to utilize the instantiations described in Sec. 6.