Efficient Hidden Vector Encryption with Constant-Size Ciphertext

Tran Viet Xuan Phuong, Guomin Yang, and Willy Susilo[⋆]

Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
Email: tvxp750@uowmail.edu.au, {gyang, wsusilo}@uow.edu.au

Abstract. A Hidden Vector Encryption (HVE) scheme is a special type of anonymous identity-based encryption (IBE) scheme where the attribute string associated with the ciphertext or the user secret key can contain wildcards. In this paper, we introduce two constant-size ciphertext-policy hidden vector encryption (CP-HVE) schemes. Our first scheme is constructed on composite order bilinear groups, while the second one is built on prime order bilinear groups. Both schemes are proven secure in a selective security model which captures plaintext (or payload) and attribute hiding. To the best of our knowledge, our schemes are the first HVE constructions that can achieve constant-size ciphertext among all the existing HVE schemes.

Keywords: Hidden vector encryption, Ciphertext policy, Constant-size ciphertext, Viète's Formulas

1 Introduction

Embedding policy-based access control into modern encryption schemes is an interesting but challenging task that has been intensively studied by the cryptologic research community in recent years. Typical examples of such encryption schemes include Attribute-based Encryption (ABE) [1–4] and Predicate Encryption [5, 6] schemes, which can be treated as special instances of a more general notion called Functional Encryption which was formalized by Boneh, Sahai, and Waters [7].

As a special type of functional encryption, Hidden Vector Encryption (HVE) schemes [5, 6, 8, 9] allow wildcards to appear in either the encryption attribute vector associated with a ciphertext or the decryption attribute vector associated with a user secret key. Similar to ABE schemes, we name the former Ciphertext Policy (CP-) HVE schemes and the latter Key Policy (KP-) HVE schemes. The decryption will work if and only if the two vectors match. That is, for each position, the two vectors must have the same letter (defined in an alphabet Σ) unless a wildcard symbol '$\star$' appears in one of these two vectors at that position. In this paper, we focus on the construction of CP-HVE schemes.

Related Works. All the recent development on functional encryptions can be traced back to the earlier work on identity-based encryption which was introduced by Shamir [10] and first realized by Boneh and Franklin [11] and Cocks [12]. One important extension of IBE is hierarchical IBE (HIBE) [13], which allows users at a level to issue keys to those on the level below.

The notion of Anonymous IBE was introduced by Boneh et al. [14] and later formalized by Abdalla et al. [15]. Compared with the normal IBE, anonymous IBE supports the additional feature of identity/attribute hiding. That is, except the user holding the correct decryption key, no one is able to link a ciphertext with the identity string used to create that ciphertext.

In [16], Abdalla et al. also proposed another extension of IBE called Wildcarded IBE (or WIBE for short). WIBE is closely related to CP-HVE except that the former does not consider the property of identity/attribute hiding when it was introduced in [16]. Abdalla et al. proposed several WIBE constructions based on the Waters HIBE [17], the Boneh-Boyen HIBE [18], and the Boneh-Boyen-Goh HIBE [13]. Recently, to address the identity hiding problem, Abdalla et al. also proposed an anonymous WIBE in [19].

In a predicate encryption system [5, 6] for a (polynomial-time) predicate P, two inputs (besides some public parameters) are required in the encryption process, one is the message M to be encrypted, and

[⋆] This work is partially supported by Australian Research Council Discovery Project (DP130101383).

the other one is an index string i. A decryption key is generated based on a master secret and a key index k. The decryption key can successfully decrypt a valid encryption of (i, M) if and only if $P(k, i) = 1$. IBE can be treated as a special type of predicate encryption where the predicate function simply performs an equality test, while for HVE the predicate function will ignore the positions where wildcard symbols '$\star$' have occurred when doing an equality test.

After the notion of hidden vector encryption was first proposed by Boneh and Waters in [5], several HVE schemes [6, 20–22, 8, 23, 9] have been proposed, most of which are key policy based (i.e., the wildcards '$\star$' appear in the decryption attribute vector). One common drawback in many early HVE schemes (e.g. [5, 6, 21, 22]) is that the ciphertext size and the decryption key size are large (linear in the length of the vector). In [8], Sedghi et al. proposed an HVE scheme that has constant decryption key size and short (but still not constant-size) ciphertext. In [9], Hattori et al. introduced a formal definition for CP-HVE and proposed a CP-HVE scheme based on the anonymous HIBE proposed in [24] and the wildcarded IBE proposed in [16]. Hattori et al.'s CP-HVE scheme also has a linear cipertext size. To the best of our knowledge, there is no HVE scheme proposed in the literature that can achieve constant-size ciphertext.

Our Contributions. We propose two ciphertext policy hidden vector encryption schemes with constant-size ciphertext.

- Our first proposed scheme (CP-HVE1) is constructed on bilinear groups with a composite order $n = pq$ where p, q are prime numbers. The security of the scheme is proven in the standard model under three complexity assumptions: the Decisional L-composite Bilinear Diffie-Hellman Exponent (L-cBDHE) assumption, the L-composite Decisional Diffie Hellman (l-cDDH) assumption, and the Bilinear Subspace Decision (BSD) assumption.
- Additionally, we also construct our second scheme (CP-HVE2), which is built on bilinear groups with a prime order. We note that our second scheme is more efficient compared to the scheme converted from CP-HVE1 by applying the conversion tool from a composite order to a prime order bilinear group. Our second scheme is proven under the Decisional L-Bilinear Diffie-Hellman Exponent (L-BDHE) assumption.

We highlight the differences between our schemes and the previous HVE schemes in Table 1. A more detailed comparison among these schemes is given in Sec. 7.

Table 1. A Comparison on Ciphertext Size and Key Size among HVE Schemes

Scheme	**Type**	**Constant Ciphertext Size**	**Constant Key Size**
Katz et al. [6]	Key Policy	No	No
Shi, Waters [20]	Key Policy	No	No
Ivovino and Persiano [21]	Key Policy	No	No
Sedghi et al. [8]	Key Policy	No	Yes
Lee and Dong [25]	Key Policy	No	Yes
Park [23]	Key Policy	No	Yes
Hattori et al. [9]	Ciphertext Policy	No	No
Ours	Ciphertext Policy	Yes	No

2 Preliminaries

2.1 Bilinear Map on Prime Order Groups

Let $\mathbb{G}$ and $\mathbb{G}_{\mathbb{T}}$ be two multiplicative cyclic groups of same prime order p, and g a generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a bilinear map with the following properties:

1. Bilinearity : $e(u^a, v^b) = e(u^b, v^a) = e(u, v)^{ab}$ for all u,$v \in \mathbb{G}$ and a,$b \in \mathbb{Z}_p$.
2. Non-degeneracy : $e(g, g) \neq 1$

Notice that the map e is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

Decision L-BDHE Assumption. The Decision L-BDHE problem in $\mathbb{G}$ is defined as follows: Let $\mathbb{G}$ be a bilinear group of prime order p, and g, h two independent generators of $\mathbb{G}$. Denote $\overrightarrow{y}_{g,\alpha,L} = (g_1, g_2, \ldots, g_L, g_{L+2}, \ \ldots, g_{2L}) \in \mathbb{G}^{2L-1}$ where $g_i = g^{\alpha^i}$ for some unknown $\alpha \in \mathbb{Z}_p^*$. We say that the L-BDHE assumption holds in $\mathbb{G}$ if for any probabilistic polynomial-time algorithm A

$$|\Pr[A(g, h, \overrightarrow{y}_{g,\alpha,L}, e(g_{L+1}, h)) = 1] - \Pr[A(g, h, \overrightarrow{y}_{g,\alpha,L}, T) = 1]| \leq \epsilon(k)$$

where the probability is over the random choive of g, h in $\mathbb{G}$, the random choice $\alpha \in \mathbb{Z}^*{}_p$, the random choice $T \in \mathbb{G}_T$, and $\epsilon(k)$ is negligible in the security parameter k.

2.2 Bilinear Map on Composite Order Groups

Let p, q be two large prime numbers and $n = pq$. Let $\mathbb{G}, \mathbb{G}_T$ be cyclic groups of order n, We say $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is bilinear map on composite order groups if e satisfies the following properties:

1. Bilinearity : $e(u^a, v^b) = e(u^b, v^a) = e(u, v)^{ab}$. for all u,$v \in \mathbb{G}$ and a,$b \in \mathbb{Z}_p$.
2. Non-degeneracy : $e(g, g) \neq 1$

Let $\mathbb{G}_p$ and $\mathbb{G}_q$ be two subgroups of $\mathbb{G}$ of order p and q, respectively. Then $\mathbb{G} = \mathbb{G}_p \times \mathbb{G}_q$, $\mathbb{G}_T = \mathbb{G}_{T,p} \times \mathbb{G}_{T,q}$. We use g_p and g_q to denote generators of $\mathbb{G}_p$ and $\mathbb{G}_q$, respectively. $e(h_p, h_q) = 1$ for all elements $h_p \in \mathbb{G}_p$ and $h_q \in \mathbb{G}_q$ since $e(h_p, h_q) = e(g_p^a, g_q^b) = e(g^{qa}, g^{pb}) = e(g, g)^{pqab} = 1$ for a generator g of $\mathbb{G}$.

Below are three complexity assumptions defined on composite order bilinear groups: the decisional L-composite bilinear Diffie-Hellman exponent (L-cBDHE) assumption, the L-composite Decisional Diffie-Hellman (L-cDDH) assumption, and the bilinear subspace decision (BSD) assumption.

The Decisional $L-$cBDHE assumption:

$$\begin{array}{l} \text{Let } g_p, h \xleftarrow{R} \mathbb{G}_p, g_q \xleftarrow{R} \mathbb{G}_q, \alpha \xleftarrow{R} \mathbb{Z}_n \\ Z = (g_p, g_q, h, g_p^{\alpha}, \ldots, g_p^{\alpha^L}, g_p^{\alpha^{L+2}}, \ldots, g_p^{\alpha^{2L}}), \\ T = e(g_p, h)^{\alpha^{L+1}}, \text{ and } R \leftarrow \mathbb{G}_{T,p} \end{array}$$

We say that the decisional $L-$cBDHE assumption holds if for any probabilistic polynomial-time algorithm A

$$|\Pr[A(Z, T) = 1] - \Pr[A(Z, R) = 1]| \leq \epsilon(k)$$

where $\epsilon(k)$ denotes an negligible function of k.

The $L - c$DDH assumption:

$$\begin{array}{l} \text{Let } g_p \xleftarrow{R} \mathbb{G}_p, g_q, R_1, R_2, R_3 \xleftarrow{R} \mathbb{G}_q, \alpha, \beta \xleftarrow{R} \mathbb{Z}_n \\ Z = (g_p, g_q, g_p^{\alpha}, \ldots, g_p^{\alpha^L}, g_p^{\alpha^{L+1}} R_1, g_p^{\alpha^{L+1}\beta} R_2) \\ T = g_p^{\beta} R_3, \text{ and } R \leftarrow \mathbb{G} \end{array}$$

We say that the $L - c$DDH assumption holds if for any probabilistic polynomial-time algorithm A

$$|\Pr[A(Z, T) = 1] - \Pr[A(Z, R) = 1]| \leq \epsilon(k)$$

where $\epsilon(k)$ denotes an negligible function of k.

The BSD assumption:

$$\begin{array}{l} \text{Let } g_p \leftarrow \mathbb{G}_p, g_q \leftarrow \mathbb{G}_q \\ Z = (g_p, g_q) \\ T \leftarrow \mathbb{G}_{T,p}, \text{ and } R \leftarrow \mathbb{G}_{T,p} \end{array}$$

We say that the BSD assumption holds if for any probabilistic polynomial-time algorithm A

$$|\Pr[A(Z, T) = 1] - \Pr[A(Z, R) = 1]| \leq \epsilon(k)$$

where $\epsilon(k)$ denotes an negligible function of k.

2.3 The Viète's formulas

Both of our schemes introduced in this paper are based on the Viète's formulas [8] which is reviewed below. Consider two vectors $\overrightarrow{v} = (v_1, v_2, \ldots, v_L)$ and $\overrightarrow{z} = (z_1, z_2, \ldots, z_L)$. Vector v contains both alphabets and wildcards, and vector z only contains alphabets. Let $J = \{j_1, \ldots, j_n\} \subset \{1, \ldots, L\}$ denote the positions of the wildcards in vector $\overrightarrow{v}$. Then the following two statements are equal:

$$\begin{aligned} & v_i = z_i \vee v_i = * \text{ for } i = 1 \ldots L \\ & \sum_{i=1, i \notin J}^{L} v_i \prod_{j \in J} (i - j) = \sum_{i=1}^{L} z_i \prod_{j \in J} (i - j). \end{aligned} \tag{1}$$

Expand $\prod_{j \in J} (i - j) = \sum_{k=0}^{n} a_k i^k$, where a_k are the coefficients dependent on J, then (1) becomes:

$$\sum_{i=1, i \notin J}^{L} v_i \prod_{j \in J} (i - j) = \sum_{k=0}^{n} a_k \sum_{i=1}^{L} z_i i^k \tag{2}$$

To hide the computations, we choose random group elemen H_i and put v_i, z_i as the exponents of group elements: $H_i^{v_i}, H_i^{z_i}$. Then (2) becomes:

$$\prod_{i=1, i \notin J}^{L} H_i^{v_i \prod_{j \in J}(i-j)} = \prod_{k=0}^{n} (\prod_{i=1}^{L} H_i^{z_i i^k})^{a_k} \tag{3}$$

Using Viète's formulas we can construct the coefficient a_k in (2) by:

$$a_{n-k} = (-1)^k \sum_{1 \le i_1 < i_2 < \ldots < i_k \le n} j_{i_1} j_{i_2} \ldots j_{i_k}, \ 0 \le k \le n. \tag{4}$$

where $n = |J|$. If we have $J = \{j_1, j_2, j_3\}$, the polynomial is $(x - j_1)(x - j_2)(x - j_3)$, then:

$$\begin{aligned} a_3 &= 1 \\ a_2 &= -(j_1 + j_2 + j_3) \\ a_1 &= (j_1 j_2 + j_1 j_3 + j_2 j_3) \\ a_0 &= -j_1 j_2 j_3. \end{aligned}$$

3 Ciphertext-Policy Hidden Vector Encryption

A ciphertext-policy hidden vector encryption (CP-HVE) scheme consists of the following four probabilistic polynomial-time algorithms:

- **Setup**$(1^k, \Sigma, L)$: on input a security parameter 1^k, an alphabet Σ, a vector-length L, the algorithm outputs a public key PK and master secret key MSK.
- **Encryption**$(PK, \overrightarrow{v}, M)$: on input a public key PK, a message M, a vector $\boldsymbol{v} \in \Sigma_L^*$ where Σ^* denotes $\Sigma \cup \{*\}$, the algorithm outputs a ciphertext CT.
- **KeyGen**$(MSK, \overrightarrow{x})$: on input a master secret key MSK , a vector $\overrightarrow{x} \in \Sigma_L$, the algorithm outputs a decryption key SK.
- **Decryption**(CT, SK): on input a ciphertext CT and a secret key SK, the algorithm outputs either a message M or a special symbol $\perp$.

Security Model. The security model for a CP-HVE scheme is defined via the following game between an adversary A and a challenger B.

- **Init**: The adversary A chooses two target patterns,

$$\overrightarrow{v_0^*} = (v_{0,1}, v_{0,2}, \ldots, v_{0,L}) \text{ and } \overrightarrow{v_1^*} = (v_{1,1}, v_{1,2}, \ldots, v_{1,L})$$

under the restriction that the wildcards '*' must appears at the same positions.

- **Setup**: The challenger B run $\mathbf{Setup}(k, \Sigma, L)$ to generate the PK and MSK. PK is then passed to A.
- **Query Phase 1**: A adaptively issues key queries for $\overrightarrow{\sigma} = (\sigma_1, \ldots, \sigma_L) \in \Sigma_L$ under the restriction that $\overrightarrow{\sigma}$ does not match $\overrightarrow{v_0^*}$ or $\overrightarrow{v_1^*}$. That is, there exist $i, j \in \{1, \ldots, L\}$ such that $v_{0,i}^* \neq * \wedge v_{0,i}^* \neq \sigma_i$, and $v_{1,j}^* \neq * \wedge v_{1,j}^* \neq \sigma_j$. The challenger runs $\mathbf{KeyGen}(MSK, \overrightarrow{\sigma})$ and returns the corresponding decryption key to A.
- **Challenge**: A outputs two equal-length messages M_0^*, M_1^*. B picks $\beta \leftarrow \{0, 1\}$ and runs Encrypt$(PK, \overrightarrow{v_\beta^*}, M_\beta^*)$ to generate a challenge ciphertext C^*. B then passes C^* to A.
- **Query Phase 2**: same as Learning Phase 1.
- **Output**: A outputs a bit β' as her guess for β.

Define the advantage of A as

$$\mathbf{Adv}_A^{\mathsf{CP-HVE}}(k) = \Pr[\beta' = \beta] - 1/2.$$

4 CP-HVE Scheme 1

In this section, we present our first CP-HVE under composite order bilinear groups. Let $\overrightarrow{v}$ denote the attribute vector associated with the ciphertext and $\overrightarrow{z}$ the attribute vector associated with the user secret key. The expression of these two vectors is designed based on the idea The Viète's formulas. To do encryption, we represent each component of the vector $\overrightarrow{v}$ by $(g^{v_i})^{\prod_{j \in J}(i-j)}$ where J denotes all the wildcard positions and is attached to the ciphertext. Notice that $\prod_{j \in J}(i - j) = \sum_{k=0}^{n} a_k i^k$ according to the Viète's formulas. In the decryption process, based on J, the decryptor can reconstruct the coefficients a_k, and generate $\prod_{j \in J} g^{z_i i^k a_k} = (g^{z_i})^{\prod_{j \in J}(i-j)}$ for each component of $\overrightarrow{z}$. In this way, whether $v_i = z_i$ will not affect the decryption if $i \in J$.

► $\mathbf{Setup}(1^k, \Sigma, L)$: The setup algorithm first chooses $N << L$ where N is the maximum number of wildcards that are allowed in an encryption vector. It then picks large primes p, q, generates bilinear groups $\mathbb{G}, \mathbb{G}_T$ of composite order $n = pq$, and selects generators $g_p \in \mathbb{G}_p, g_q \in \mathbb{G}_q$. After that, it selects random elements:

$$\begin{array}{l} g, f, v, v', h_1, \ldots, h_L, h'_1, \ldots, h'_L, w \in \mathbb{G}_p, \\ R_g, R_f, R_v, R_{v'}, R_{h_1}, \ldots, R_{h_L}, R_{h'_1}, \ldots, R_{h'_L} \in \mathbb{G}_q, \end{array}$$

and computes :

$$\begin{array}{l} G = gR_g, F = fR_f, V = vR_v, V' = v'R_{v'}, \\ H_1 = h_1 R_{h_1}, \ldots, H_L = h_L R_{h_L}, \\ H'_1 = h'_1 R_{h'_1}, \ldots, H'_L = h'_L R_{h'_L}, \\ E = e(g, w). \end{array}$$

Then it creates the public key and master secret key as:

$$\begin{array}{l} PK = \{g_p, g_q, G, F, V, V', (H_1, \ldots, H_L), (H'_1, \ldots, H'_L), E\}, \\ MSK = \{p, q, g, f, v, v', (h_1, \ldots, h_L), (h'_1, \ldots, h'_L), w\}. \end{array}$$

► $\mathbf{Encrypt}(PK, M, \overrightarrow{v} = (v_1, \ldots, v_L) \in \Sigma_L^*)$: Suppose that $\overrightarrow{v}$ contains $\tau \leq N$ wildcards which occur at positions $J = \{j_1, \ldots, j_\tau\}$. The encryption algorithm first chooses:

$$s \in_R \mathbb{Z}_n, \text{ and } Z_1, Z_2, Z_3, Z_4 \in_R \mathbb{G}_q.$$

Using formulas (3) and (4), compute a_k for $k = 1, 2, \cdots, \tau$, and $t = a_0$. Then set:

$$\begin{array}{l} C_0 = M \cdot E^s, C_1 = G^{\frac{s}{t}} Z_1, C_2 = F^s Z_2, \\ C_3 = ((\prod_{i=1}^{L} V H_i^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{t}} \cdot Z_3, C_4 = ((\prod_{i=1}^{L} V'(H'_i)^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{t}} \cdot Z_4, \\ J = \{j_1, j_2, \ldots, j_\tau\}, \end{array}$$

and ciphertext $CT = \{C_0, C_1, C_2, C_3, C_4, J\}$.

- **KeyGen**$(MSK, \overrightarrow{z} = (z_1, \ldots, z_L) \in \Sigma_L)$: The key generation algorithm chooses r_1, r_1', r_2 randomly in Z_n, and computes:

$$K_1 = g^{r_1}, K_2 = g^{r_1'}, K_3 = g^{r_2}, \begin{pmatrix} K_{4,0} = & w(\prod_{i=1}^{L} h_i^{z_i} v)^{r_1} (\prod_{i=1}^{L} (h_i')^{z_i} v')^{r_1'} f^{r_2}, \\ K_{4,1} = & (\prod_{i=1}^{L} h_i^{z_i} v)^{ir_1} (\prod_{i=1}^{L} (h_i')^{z_i} v')^{ir_1'}, \\ \ldots & \\ K_{4,N} = & (\prod_{i=1}^{L} h_i^{z_i} v)^{i^N r_1} (\prod_{i=1}^{L} (h_i')^{z_i} v')^{i^N r_1'} \end{pmatrix}.$$

The secret key is $SK = \{K_1, K_2, K_3, K_{4,0}, \ldots, K_{4,N}\}$.

- **Decrypt**(CT, SK): The decryption algorithm first applies the Viète's formulas to compute

$$a_{\tau-k} = (-1)^k \sum_{1 \le i_1 < i_2 < \ldots < i_k \le \tau} j_{i_1} j_{i_2} \ldots j_{i_k}, 0 \le k \le \tau$$

and then outputs:

$$M = \frac{e(K_1, C_3) \cdot e(K_2, C_4) \cdot e(K_3, C_2)}{e(\prod_{k=0}^{\tau} K_{4,k}^{a_k}, C_1)} \cdot C_0.$$

Correctness:

$$\begin{aligned} e(K_1, C_3) &= e(g^{r_1}, ((\prod_{i=1}^{L} V H_i^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{a_0}} \cdot Z_3) \\ &= \prod_{i=1}^{L} e(g, v)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h_i)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)v_i}{a_0}}. \\ e(K_2, C_4) &= e(g^{r_1'}, ((\prod_{k=1}^{L} V'(H')_i^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{a_0}} \cdot Z_4) \\ &= \prod_{i=1}^{L} e(g, v')^{\frac{sr_1' \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h_i')^{\frac{sr_1' \prod_{k=1}^{\tau}(i-j_k)v_i}{a_0}}. \end{aligned}$$

$$e(K_3, C_2) = e(g^{r_2}, F^s Z_2) = e(g, f)^{r_2 s}.$$

$$\begin{aligned} e(\prod_{k=0}^{\tau} K_{4,k}^{a_k}, C_1) &= e(w^{a_0}(\prod_{k=0}^{\tau}\prod_{i=1}^{L} v^{i^k a_k} h_i^{z_i i^k a_k} v^{i^k a_k})^{r_1} (\prod_{k=0}^{\tau}\prod_{i=1}^{L} (h_i')^{z_i i^k a_k} v'^{i^k a_k})^{r_1'} f^{r_2 a_0}, G^{\frac{s}{a_0}} Z_1) \\ &= e(g, w)^{\frac{sa_0}{a_0}} \cdot e(g, f)^{\frac{sr_2 a_0}{a_0}} \cdot \prod_{i=1}^{L} e(g, h_i)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}} e(g, v)^{\frac{sr_1 \sum_{k=0}^{\tau} i^k a_k}{a_0}} \\ &\cdot \prod_{i=1}^{L} e(g, h_i')^{\frac{sr_1' \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}} e(g, v')^{\frac{sr_1' \sum_{k=0}^{\tau} i^k a_k}{a_0}} \end{aligned}$$

$$\begin{aligned} &= e(g, w)^s \cdot e(g, f)^{sr_2} \cdot \prod_{i=1}^{L} e(g, v)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h_i)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}} \\ &\cdot \prod_{i=1}^{L} e(g, v')^{\frac{sr_1' \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h_i')^{\frac{sr_1' \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}}. \end{aligned}$$

Then we have

$$\begin{aligned}
&e(K_1, C_3) \cdot e(K_2, C_4) \cdot e(K_3, C_2) \\
= &\prod_{i=1}^{L} e(g, v)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h_i)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)v_i}{a_0}} \cdot \prod_{i=1}^{L} e(g, v')^{\frac{sr'_1 \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h'_i)^{\frac{sr'_1 \prod_{k=1}^{\tau}(i-j_k)v_i}{a_0}}, \\
&e(\prod_{k=0}^{\tau} K_{4,k}^{a_k}, C_1) \\
= &\prod_{i=1}^{L} e(g, v)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h_i)^{\frac{sr_1 \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}} \cdot \prod_{i=1}^{L} e(g, v')^{\frac{sr'_1 \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, h'_i)^{\frac{sr'_1 \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}} \\
&\cdot e(g, w)^s \cdot e(g, f)^{sr_2},
\end{aligned}$$

and can recover message M by:

$$\frac{e(K_1, C_3) \cdot e(K_2, C_4) \cdot e(K_3, C_2)}{e(\prod_{k=0}^{\tau} K_{4,k}^{a_k}, C_1)} \cdot C_0 = \frac{e(g, f)^{r_2 s} \cdot M \cdot e(g, w)^s}{e(g, w)^s \cdot e(g, f)^{sr_2}} = M.$$

Theorem 1. *Our CP-HVE Scheme 1 is secure if the Decisional $L-$cBDHE assumption, the $L-cDDH$ assumption, and the BSD assumption hold.*

We prove Theorem 1 by the following sequence of games.

$$\begin{aligned}
Game_0 &: [C_0, C_1, C_2, C_3, C_4] \\
Game_1 &: [C_0 \cdot R_p, C_1, C_2, C_3, C_4] \\
Game_2 &: [R_0, C_1, C_2, C_3, C_4] \\
Game_3 &: [R_0, C_1, C_2, R_3, C_4] \\
Game_4 &: [R_0, C_1, C_2, R_3, R_4],
\end{aligned}$$

where R_p is a randomly chosen from $\mathbb{G}_{T,p}$, R_0 is uniformly distributed in $\mathbb{G}_T$, and R_0, R_3, R_4 are uniformly distributed in $\mathbb{G}$.

We will prove the following Lemmas. Notice that in $Game_4$ the challenge ciphertext is independent of the message and the encryption vector, which means the adversary has no advantage in winning the game over random guess.

Lemma 1. *Assume that the Decisional $L-$cBDHE assumption holds, then for any PPT adversary, the difference between the advantages in $Game_0$ and $Game_1$ is negligible.*

Lemma 2. *Assume that the BSD assumption holds, then for any PPT adversary, the difference between the advantages in $Game_1$ and $Game_2$ is negligible.*

Lemma 3. *Assume that the $L-$cDDH assumption holds, then for any PPT adversary, the difference between the advantages in $Game_2$ and $Game_3$ is negligible.*

Lemma 4. *Assume that the $L-$cDDH assumption holds, then for any PPT adversary, the difference between the advantages in $Game_3$ and $Game_4$ is negligible.*

4.1 Proof of Lemma 1

Assume that the Decisional $L-$cBDHE assumption holds, then for any PPT adversary, the difference between the advantages in $Game_0$ and $Game_1$ is negligible.

We assume that adversary A's advantage has a difference ϵ between $Game_0$ and $Game_1$. The simulator B will use A to solve the Decisional $L-$cBDHE problem. B is given a challenge instance Z, T' of the problem, where $Z = (g_p, g_q, h, g_p^{\alpha}, \ldots, g_p^{\alpha^L}, g_p^{\alpha^{L+2}}, \ldots, g_p^{\alpha^{2L}})$ and T' is either $T = e(g_p, h)^{\alpha^{L+1}}$ or $R \in_R \mathbb{G}_{T,p}$.

In the rest of the proof, we denote $W(\overrightarrow{v}) = \{1 \leq i \leq L | v_i = *\}$ and $\overline{W}(\overrightarrow{v}) = \{1 \leq i \leq L | v_i \neq *\}$, and $W(\overrightarrow{v})|_j^k$ as $\{i \in W(\overrightarrow{v}) | j \leq i \leq k\}$.

- **Init**: A declares two challenge alphabet vectors $\overrightarrow{v_0^*} = (v_{0,1}^*, \ldots, v_{0,L}^*)$ and $\overrightarrow{v_1^*} = (v_{1,1}^*, \ldots, v_{1,L}^*)$ under the restriction that $W(\overrightarrow{v_0^*}) = W(\overrightarrow{v_1^*})$.
- **Setup**: In this phase, B generates:

$$\gamma, y, y', \psi, \{u_i, u_i'\}_{i \in |L|} \xleftarrow{R} \mathbb{Z}_n, R_g, R_f, R_v, R_{v'}, R_{h_1}, \ldots, R_{h_L}, R_{h_1'}, \ldots, R_{h_L'} \xleftarrow{R} \mathbb{G}_q$$

Then B flips a coin $\mu \in \{0,1\}$ and sets:

$$G = g_p R_g, F = g_p^{\psi} R_f, E = e(g_p^{\alpha}, g_p^{\alpha^L} g_p^{\gamma}), V = g_p^y \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*} R_v, V' = g_p^{y'} R_{v'},$$
$$\{H_i = g_p^{u_i - \alpha^{L+1-i}} R_{h,i}\}_{i \in \overline{W}(\overrightarrow{v_\mu^*})}, \{H_i = g_p^{u_i} R_{h,i}\}_{i \in W(\overrightarrow{v_\mu^*})}, \{H_i' = g_p^{u_i} R_{h',i}\}_{i=1}^{L}.$$

Then the corresponding elements of the master secret key are: $g = g_p$, $f = g_p^{\psi}$, $\{h_i = g_p^{u_i - \alpha^{L+1-i}}\}_{i \in \overline{W}(\overrightarrow{v_\mu^*})}$, $\{h_i = g_p^{u_i}\}_{i \in W(\overrightarrow{v_\mu^*})}$, $\{h_i' = g_p^{u_i'}\}_{i=1}^{L}$, $v' = g_p^{y'}$, $v = g_p^y \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*}$.

The master key component w is $g_p^{\alpha^{L+1} + \alpha\gamma}$. Since B does not have $g_p^{\alpha^{L+1}}$, B cannot compute w directly.
- **Query Phase 1**: A queries the user secret key for $\overrightarrow{\sigma_u} = (\sigma_1, \sigma_2, \ldots, \sigma_u)$ that does not match the challenge patterns. Let $k \in \overline{W}(\overrightarrow{v_\mu^*})$ be the smallest integer such that $\sigma_k \neq v_{\mu,k}^*$.
B needs to simulate the user key generation process. We start from $K_{4,i}$.

$$\begin{aligned} K_{4,0} &= w(\prod_{i=1}^{L} h_i^{\sigma_i} v)^{r_1} (v' \prod_{i=1}^{L} (h_i')^{\sigma_i})^{r_1'} f^{r_2} \\ &= g_p^{\alpha^{L+1} + \alpha\gamma} (\prod_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} g_p^{(u_i - \alpha^{L+1-i})\sigma_i} \cdot \prod_{W(\overrightarrow{v_\mu^*})|_1^k} (g_p^{u_i})^{\sigma_i} \cdot g_p^{y + \sum_{i \in \overline{W}(\overrightarrow{v_\mu^*})}^{\alpha^{L+1-i} v_{\mu,i}^*}} \cdot)^{r_1} (v' \prod_{i=1}^{L} (h_i')^{z_i})^{r_1'} f^{r_2}. \\ &\overset{def}{=} g_p^{\alpha^{L+1} + \alpha\gamma} (g_p^X)^{r_1} (v' \prod_{i=1}^{L} (h_i')^{z_i})^{r_1'} f^{r_2} \end{aligned}$$

where

$$X = \sum_{\overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^* + y + \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i \sigma_i.$$

Since

$$\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i \sigma_i = \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (-\alpha^{L+1-i}\sigma_i) + \sum_{i=1}^{k} u_i \sigma_i$$

and recall $\sigma_i = v_{\mu,i}^*$ for $i \in \overline{W}(\overrightarrow{v_\mu^*})|_1^{k-1}$ and $\sigma_k \neq v_{\mu,k}^*$. Hence, we have

$$X = \alpha^{L+1-k} \varDelta_k + \textstyle\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^{L}} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{i=1}^{k} x_i \sigma_i + y$$

where $\varDelta_k = v_{\mu,k}^* - \sigma_k$. Then we choose $r_1', \hat{r_1}, r_2$ randomly in $\mathbb{Z}_n$, and set $r_1 = \frac{-\alpha^k}{\varDelta_k} + \hat{r_1}$. $K_{4,0}$ can be represented as

$$\begin{aligned} &K_{4,0} \\ &= g_p^{\alpha^{L+1} + \alpha\gamma} \cdot g_p^{-\alpha^{L+1}} \cdot g_p^{\sum_{i \in \overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^{L}}^{\frac{-\alpha^{L+1-i+k} v_{\mu,i}^*}{\varDelta_k}}} \cdot g_p^{a^k(-\frac{\sum_{i=1}^{k} x_i \sigma_i + y \cdot}{\varDelta_k})} \cdot (v \prod_{i=1}^{k} h_i^{\sigma_i})^{\hat{r_1}} (v' \prod_{i=1}^{L} (h_i')^{\sigma_i})^{r_1'} \cdot f^{r_2} \\ &= g_p^{\alpha\gamma} \cdot g_p^{\sum_{i \in \overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^{L}}^{\frac{-\alpha^{L+1-i+k} v_{\mu,i}^*}{\varDelta_k}}} \cdot g_p^{a^k(-\frac{\sum_{i=1}^{k} x_i \sigma_i + y \cdot}{\varDelta_k})} \cdot (v \prod_{i=1}^{k} h_i^{\sigma_i})^{\hat{r_1}} \cdot (v' \prod_{i=1}^{L} (h_i')^{\sigma_i})^{r_1'} \cdot f^{r_2}. \end{aligned}$$

For $\hat{k} = 1$ to N, we compute

$$K_{4,\hat{k}} = ((\prod_{\overline{W}(\overrightarrow{v_\mu^*})|_1^{k-1}} (g_p^{u_i - \alpha^{L+1-i}})^{\sigma_i} \cdot \prod_{W(\overrightarrow{v_\mu^*})|_1^{k-1}} (g_p^{u_i})^{\sigma_i} \cdot g_p^{y + \sum_{i \in \overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^*})^{\frac{-\alpha^k i^{\hat{k}}}{\Delta_k} + \hat{r_1} i^{\hat{k}}} \cdot (v' \prod_{i=1}^{L} (h'_i)^{\sigma_i})^{r'_1 i^{\hat{k}}}.$$

Other elements in the key can also be simulated:

$$K_1 = g_p^{r_1} = (g_p^{\alpha^k})^{-1/\Delta_k} \cdot g_p^{\hat{r_1}}, K_2 = g_p^{r'_1}, K_3 = g_p^{r_2}.$$

- **Challenge**: A sends to message M_0, M_1 to B. B generates $Z_1, Z_2, Z_3, Z_4 \xleftarrow{R} \mathbb{G}_q$ and then sets using Viète's formulas
$$a_{\tau-k} = (-1)^k \sum_{i \le i_1 < i_2 < \ldots < i_k \le \tau} j_{i_1} j_{i_2} \ldots j_{i_k}, 0 \le k \le \tau.$$
Let $t = a_0$. It creates ciphertext as:
$$C_0 = M_b \cdot T' \cdot e(g_p^\alpha, h)^\gamma, C_1 = h^{1/t} \cdot Z_1, C_2 = h^\psi \cdot Z_2,$$
$$C_3 = ((h^{y + \sum_{i=1}^{L} u_i v_{\mu,i}^*})^{\prod_{k=1}^{\tau}(i - j_k)})^{\frac{1}{t}} \cdot Z_3, C_4 = ((h^{y' + \sum_{i=1}^{L} u'_i v_{\mu,i}^*})^{\prod_{k=1}^{\tau}(i - j_k)})^{\frac{1}{t}} \cdot Z_4.$$
If $T' = T = e(g_p, h)^{\alpha^{L+1}}$, where $h = g_p^c$ for some unknown $c \in \mathbb{Z}_p$, then
$$C_0 = M_b \cdot e(g_p, g_p^c)^{\alpha^{L+1}} \cdot e(g_p^\alpha, g_p^c)^\gamma = M_b \cdot e(g_p, g_p^{\alpha^{L+1}})^c \cdot e(g_p^\alpha, g_p^\gamma)^c = M_b \cdot E^c,$$
$$C_1 = (g_p^c)^{1/t} \cdot Z_1 = G^{c/t} \cdot Z'_1, C_2 = (g_p^c)^\psi \cdot Z_2 = F^c \cdot Z'_2,$$
$$C_3 = ((g_p^{y + \sum_{i=1}^{L} u_i v_{\mu,i}^*})^{\prod_{k=1}^{\tau}(i - j_k)})^{\frac{c}{t}} \cdot Z_3 = ((V \prod_{i=1}^{L} H_i^{v_i})^{\prod_{k=1}^{\tau}(i - j_k)})^{\frac{s}{t}} \cdot Z'_3,$$
$$C_4 = ((g_p^{y' + \sum_{i=1}^{L} u'_i v_{\mu,i}^*})^{\prod_{k=1}^{\tau}(i - j_k)})^{\frac{c}{t}} \cdot Z_4 = ((V' \prod_{i=1}^{L} (H'_i)^{v_i})^{\prod_{k=1}^{\tau}(i - j_k)})^{\frac{s}{t}} \cdot Z'_4.$$
Hence, A is in $Game_0$. Otherwise, if $T' = R_p = e(g_p, h)^{\alpha^{L+1}} \cdot R'_p$ for some random $R'_p \in \mathbb{G}_{T,p}$, then A is in $Game_1$.
- **Query Phase 2**: Repeat Phase 1.
- **Guess**: A output $b' \in \{0, 1\}$. If $b' = b$ then B outputs 1; otherwise B outputs 0.

Let $Adv_B(k)$ be the advantage of B to solve the $L-$wDBDHI problem, and $Adv_A^{Game_0}(k)$, $Adv_A^{Game_1}(k)$ be the advantages of A in $Game_0$ and $Game_1$. Then we have

$$\begin{aligned} Adv_B(\lambda) &= |\Pr[B \to 1 | T' = T] - \Pr[B \to 1 | T' = R_p]| \\ &= |\Pr[B \to 1 | Game_0] - \Pr[B \to 1 | Game_1]| \\ &= |(\tfrac{1}{2} + Adv_A^{Game_0}(k) - (\tfrac{1}{2} + Adv_A^{Game_1}(k))| \\ &= \epsilon. \end{aligned}$$

4.2 Proof of Lemma 2

Assume that the BSD assumption holds, then for any PPT adversary, the difference between the advantages in $Game_1$ and $Game_2$ is negligible.

We assume that adversary A's advantage has a difference ϵ between $Game_1$ and $Game_2$. The simulator B is given a challenge instance Z, T' of the BSD problem, where $Z = (g_p, g_q)$ and T' is either $T \xleftarrow{R} \mathbb{G}_{T,p}$ or $R \xleftarrow{R} \mathbb{G}_T$. B simulates the game for A as follows.

- **Init**: A declares two challenge alphabet vectors.

- **Setup**: B follows the setup algorithm and creates the public key and master secret key using g_p and g_q.
- **Query Phase 1**: A queries the user secret key for $\overrightarrow{\sigma}$. B simulate the key generation algorithm honestly by using the master secret key.
- **Challenge**: A sends to message M_0, M_1 to B. B flips a coin $b \overset{R}{\longleftarrow} \{0,1\}$ and returns a normal ciphertext of M_b, with the exception that C_0 is multiplied by T'. If $T' = T \overset{R}{\longleftarrow} \mathbb{G}_{T,p}$ then A is in $Game_1$; otherwise, if $T' = R \overset{R}{\longleftarrow} \mathbb{G}_T$, then A is in $Game_2$.
- **Query Phase 2**: Repeat Phase 1.
- **Guess**: A output $b' \in \{0,1\}$. If $b' = b$ then B outputs 1; otherwise, B outputs 0.

Let $Adv_B(k)$ be the advantage of B to solve the BSD problem, and $Adv_A^{Game_0}(k)$, $Adv_A^{Game_1}(k)$ be the advantages of A in $Game_1$ and $Game_2$. Then we have

$$\begin{aligned}
Adv_B(k) &= |\Pr[B \to 1 | T' = T] - \Pr[B \to 1 | T' = R]| \\
&= |\Pr[B \to 1 | Game_1] - \Pr[B \to 1 | Game_2]| \\
&= |(\tfrac{1}{2} + Adv_A^{Game_1}(k)) - (\tfrac{1}{2} + Adv_A^{Game_2}(k))| \\
&= \epsilon.
\end{aligned}$$

4.3 Proof of Lemma 3

Assume that the $L-$cDDH assumption holds, then for any PPT adversary, the difference between the advantages in $Game_2$ and $Game_3$ is negligible.

We assume that the adversary A has difference ϵ in the advantages between $Game_2$ and $Game_3$. We use A to solve the $L-$cDDH problem. The simulator B is given a challenge instance Z, T' of the L-cDDH problem, where $Z = (g_p, g_q, h, g_p^{\alpha}, \dots, g_p^{\alpha^L}, g_p^{\alpha^{L+1}} \cdot R_1, g_p^{\alpha^{L+1}b} \cdot R_2)$ and T' is either $T = g_p^b \cdot R_3$ or $R \leftarrow \mathbb{G}$. B simulates the game for A as follows:

- **Init**: A declares two challenge alphabet vectors $\overrightarrow{v_0^*} = (v_{0,1}^*, \dots, v_{0,L}^*)$ and $\overrightarrow{v_1^*} = (v_{1,1}^*, \dots, v_{1,L}^*)$ under the restriction that $W(\overrightarrow{v_0^*}) = W(\overrightarrow{v_1^*})$.
- **Setup**: In this phase, B generates:

$$\begin{aligned}
&\gamma, y, y', \psi, \{u_i, u_i'\}_{i \in |L|} \overset{R}{\longleftarrow} \mathbb{Z}_n, \\
&w \overset{R}{\longleftarrow} \mathbb{G}_p, \\
&R_g, R_f, R_{\hat{v}}, R_{v'}, R_{h_1}, \dots, R_{h_L}, R_{h_1'}, \dots, R_{h_L'} \in \mathbb{G}_q.
\end{aligned}$$

Then B flips $\mu \in \{0,1\}$ and sets:

$$\begin{aligned}
&G = g_p R_g, F = g_p^{\psi} R_f, E = e(g_p^{\alpha}, g_p^{\alpha^L} g_p^{\gamma}), \\
&V = (g_p^{\alpha^{L+1}} R_1) \cdot g_p^y \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*} R_{\hat{v}} = g_p^{\alpha^{L+1}} \cdot g_p^y \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*} R_v, \\
&V' = g_p^{y'} \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*} R_{v'}, \\
&\{H_i = g_p^{u_i - \alpha^{L+1-i}} R_{h,i}\}_{i \in \overline{W}(\overrightarrow{v_\mu^*})}, \{H_i = g_p^{u_i} R_{h,i}\}_{i \in W(\overrightarrow{v_\mu^*})}, \\
&\{H_i' = g_p^{u_i' - \alpha^{L+1-i}} R_{h',i}\}_{i \in \overline{W}(\overrightarrow{v_\mu^*})}, \{H_i' = g_p^{u_i'} R_{h',i}\}_{i \in W(\overrightarrow{v_\mu^*})}.
\end{aligned}$$

The corresponding master secret key components are: $g = g_p, f = g_p^{\psi}$, $\{h_i = g_p^{u_i - \alpha^{L+1-i}}\}_{i \in \overline{W}(\overrightarrow{v_\mu^*})}$, $\{h_i = g_p^{u_i}\}_{i \in W(\overrightarrow{v_\mu^*})}, \{h_i' = g_p^{u_i' - \alpha^{L+1-i}}\}_{i \in \overline{W}(\overrightarrow{v_\mu^*})}, \{h_i' = g_p^{u_i'}\}_{i \in W(\overrightarrow{v_\mu^*})}, v = g_p^{\alpha^{L+1}+y} \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*}$, $v' = g_p^{y'} \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*}$. Notice that the master key component v is $g_p^{\alpha^{L+1}+y} \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g_p^{\alpha^{L+1-i} v_{\mu,i}^*}$. Since B does not have $g_p^{\alpha^{L+1}}$, B cannot compute v directly.

- **Query Phase 1**: A queries the user secret key for $\overrightarrow{\sigma_u} = (\sigma_1, \sigma_2, \ldots, \sigma_u)$ that does not match the challenge patterns. Let $k \in \overline{W}(\overrightarrow{v_\mu^*})$ be the smallest integer such that $\sigma_k \neq v_{\mu,k}^*$.
  B first simulates $K_{4,i}$ as follows.

$$\begin{aligned}
K_{4,0} &= w(\prod_{i=1}^{L} h_i^{\sigma_i} v)^{r_1} (\prod_{i=1}^{L} (h_i')^{\sigma_i} v')^{r_1'} f^{r_2} \\
&= w((\prod_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} g_p^{(u_i-\alpha^{L+1-i})\sigma_i} \cdot \prod_{W(\overrightarrow{v_\mu^*})|_1^k} (g_p^{u_i})^{\sigma_i} \cdot g_p^{\alpha^{L+1}+y+\sum_{\overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^*} \cdot)^{r_1} \\
&\quad \cdot ((\prod_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} g_p^{(u_i'-\alpha^{L+1-i})\sigma_i} \cdot \prod_{W(\overrightarrow{v_\mu^*})|_1^k} (g_p^{u_i'})^{\sigma_i} \cdot g_p^{y'+\sum_{i=1}^{L} \alpha^{L+1-i} v_{\mu,i}^*})^{r_1'} f^{r_2} \\
&\overset{def}{=} w(g_p^X)^{r_1} (g_p^Y)^{r_1'} f^{r_2}
\end{aligned}$$

where

$$X = \alpha^{L+1} + y + \sum_{i \in \overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i \sigma_i$$

and

$$Y = y' + \sum_{i \in \overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i' - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i' \sigma_i.$$

Since

$$\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i \sigma_i = \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (-\alpha^{L+1-i}\sigma_i) + \sum_{i=1}^{k} u_i \sigma_i,$$

and

$$\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i' - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i' \sigma_i = \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (-\alpha^{L+1-i}\sigma_i) + \sum_{i=1}^{k} u_i' \sigma_i,$$

and recall $\sigma_i = v_{\mu,i}^*$ for $i \in \overline{W}(\overrightarrow{v_\mu^*})|_1^{k-1}$ and $\sigma_k \neq v_{\mu,k}^*$. Hence,

$$\begin{aligned}
X &= \alpha^{L+1} + \alpha^{L+1-k}\Delta_k + \textstyle\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{i=1}^k u_i \sigma_i + y, \\
Y &= \alpha^{L+1-k}\Delta_k + \textstyle\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{i=1}^k u_i' \sigma_i + y'.
\end{aligned}$$

where $\Delta_k = v_{\mu,k}^* - \sigma_k$. B then randomly chooses $r_1, \hat{r_1'}, r_2$ in $\mathbb{Z}_n$, sets $r_1' = \frac{-\alpha^k r_1}{\Delta_k} + \hat{r_1'}$. Hence, we have:

$$\begin{aligned}
K_{4,0} &= w(g_p^X)^{r_1} (g_p^Y)^{\frac{-\alpha^k r_1}{\Delta_k} + \hat{r_1'}} \cdot f^{r_2} \\
&= w(g_p^{X + \frac{-Y\alpha^k}{\Delta_k}})^{r_1} (g_p^Y)^{\hat{r_1'}} \cdot f^{r_2} \\
&= w(g_p^{\alpha^{L+1-k}\Delta_k + \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{i=1}^k u_i\sigma_i + y} g_p^{-\frac{\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \alpha^{L+1-i+k} v_{\mu,i}^*}{\Delta_k} - \frac{(\sum_{i=1}^k u_i'\sigma_i + y')\alpha^k}{\Delta_k}})^{r_1} \\
&\quad \cdot (v' \prod_{i=1}^{L} (h_i')^{\sigma_i})^{\hat{r_1'}} \cdot f^{r_2}.
\end{aligned}$$

Also, for $\hat{k} = 1$ to N, B sets

$$\begin{aligned}
K_{4,\hat{k}} &= (g_p^{\alpha^{L+1-k}\Delta_k + \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{i=1}^k u_i\sigma_i + y} g_p^{-\frac{\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \alpha^{L+1-i+k} v_{\mu,i}^*}{\Delta_k} - \frac{(\sum_{i=1}^k u_i'\sigma_i + y')\alpha^k}{\Delta_k}})^{r_1 i^{\hat{k}}} \\
&\quad \cdot (\prod_{i=1}^{L} (h_i')^{\sigma_i} v')^{\hat{r_1'} i^{\hat{k}}}.
\end{aligned}$$

And other elements can also be simulated as follows:

$$K_1 = g_p^{r_1}, K_2 = g_p^{r_1'} = g_p^{\frac{-\alpha^k}{\Delta_k} r_1 + \hat{r_1'}}, K_3 = g_p^{r_2}.$$

- **Challenge**: A sends two message M_0, M_1 to B. Then B generates $R_0 \xleftarrow{R} \mathbb{G}_T$, $Z_1, Z_2, Z_3, Z_4 \xleftarrow{R} \mathbb{G}_q$ and sets using Viète's formulas:

$$a_{\tau-k} = (-1)^k \sum_{i \le i_1 < i_2 < \ldots < i_k \le \tau} j_{i_1} j_{i_2} \ldots j_{i_k}, 0 \le k \le \tau.$$

Let $t = a_0$. B creates ciphertext as:

$$\begin{aligned} &C_0 = R_0, C_1 = T'^{1/t} \cdot Z_1, C_2 = T'^{\psi} \cdot Z_2, \\ &C_3 = (((g_p^{\alpha^{L+1}b} R_2)(T^{\prime y+\sum_{i=1}^{L} u_i v^*_{\mu,i}}))^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_3, C_4 = ((T^{\prime y'+\sum_{i=1}^{L} u'_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_4. \end{aligned}$$

If $T' = g_p^b g_q^c$ some unknown $c \in \mathbb{Z}_q$, then we have

$$\begin{aligned} C_1 &= (g_p^b g_q^c)^{1/t} \cdot Z_1 = G^{b/t} \cdot Z'_1, \\ C_2 &= (g_p^b g_q^c)^{\psi} \cdot Z_2 = F^b \cdot Z'_2, , \\ C_3 &= (((g_p^{\alpha^{L+1}b} R_2)(g_p^b g_q^c)^{y+\sum_{i=1}^{L} u_i v^*_{\mu,i}}))^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_3 \\ &= ((g^{\alpha^{L+1}} g^{y+\sum_{i=1}^{L} u_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{b}{t}} \cdot ((R_2 g_q^{c(y+\sum_{i=1}^{L} u_i v^*_{\mu,i})})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_3 \\ &= ((V \prod_{i=1}^{L} H_i^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{t}} \cdot Z'_3, \\ C_4 &= (((g_p^b g_q^c)^{y'+\sum_{i=1}^{L} u'_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_4 \\ &= (((g_p)^{y'+\sum_{i=1}^{L} u'_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{b}{t}} \cdot (((g_q^c)^{y'+\sum_{i=1}^{L} u'_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_4 \\ &= ((V' \prod_{i=1}^{L} (H'_i)^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{t}} \cdot Z'_4. \end{aligned}$$

and CT is in $Game_2$. Otherwise, if $T' = R = g_p^{b'} g_q^{c'}$ for some $b' \in \mathbb{Z}_p, c' \in \mathbb{Z}_q$, then

$$\begin{aligned} C_1 &= (g_p^{b'} g_q^{c'})^{1/t} \cdot Z_1 = G^{b'/t} \cdot Z''_1, \\ C_2 &= (g_p^{b'} g_q^{c'})^{\psi} \cdot Z_2 = F^{b'} \cdot Z''_2, \\ C_3 &= (((g_p^{\alpha^{L+1}b} R_2)(g_p^{b'} g_q^{c'})^{y+\sum_{i=1}^{L} u_i v^*_{\mu,i}}))^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_3, \\ &= ((g_p^{\alpha^{L+1}})^{\delta \prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} ((g_p^{\alpha^{L+1}+y+\sum_{i=1}^{L} u_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{b'}{t}} \cdot ((R_2 g_q^{c'(y+\sum_{i=1}^{L} u_i v^*_{\mu,i})})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_3 \\ &= ((g_p^{\alpha^{L+1}})^{\delta \prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot ((V \prod_{i=1}^{L} H_i^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{t}} \cdot Z''_3, \\ C_4 &= (((g_p^{b'} g_q^{c'})^{y'+\sum_{i=1}^{L} u'_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_4 \\ &= (((g_p)^{y'+\sum_{i=1}^{L} u'_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{b'}{t}} \cdot (((g_q^{c'})^{y'+\sum_{i=1}^{L} u'_i v^*_{\mu,i}})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}} \cdot Z_4 \\ &= ((V' \prod_{i=1}^{L} (H'_i)^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{t}} \cdot Z''_4, \end{aligned}$$

where $\delta = b - b'$ is uniformly distributed in $\mathbb{Z}_n$ for R chosen randomly from $\mathbb{G}$, and hence CT is in $Game_3$.

- **Query Phase 2**: Repeat Phase 1.
- **Guess**: A outputs $b' \in \{0, 1\}$. If $b' = b$, then B outputs 1; otherwise B outputs 0.

Let $Adv_B(k)$ be the advantage of B in solving the $L-$cDDH problem, and $Adv_A^{Game_2}(k)$, $Adv_A^{Game_3}(k)$ be the advantage of A in $Game_2$ and $Game_3$, respectively. Then we have

$$\begin{aligned} Adv_B(k) &= |\Pr[B \to 1 | T' = T] - \Pr[B \to 1 | T' = R]| \\ &= |\Pr[B \to 1 | Game_2] - \Pr[B \to 1 | Game_3]| \\ &= |(\tfrac{1}{2} + Adv_A^{Game_2}(k)) - (\tfrac{1}{2} + Adv_A^{Game_3}(k))| \\ &= \epsilon. \end{aligned}$$

4.4 Proof of Lemma 4

The proof for Lemma 4 is almost the same as that for Lemma 3, where we generate V' as the role of V in Lemma 3. We omit the details of the proof here.

5 CP-HVE Scheme 2

One straightforward approach to obtain a new CP-HVE scheme under prime-order bilinear groups is to apply the conversion technique introduced by Lewko [26]. In this section, we present a new prime-order CP-HVE scheme that is more efficient than the converted scheme.

▶ **Setup**$(1^k, \Sigma, L)$: The setup algorithm chooses $N << L$ to be the maximum number of wildcards that are allowed in an encryption vector. Then it generates other system parameters including:

$$\begin{aligned} & e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T, \\ & L+1 \text{ random elements } V, H_1, \ldots, H_L \in_R \mathbb{G}, \\ & \text{Then chooses randomly generator } g, w, f \in \mathbb{G}, \\ & Y = e(g, w). \end{aligned}$$

The public key and master secret key are set as:

$$\begin{aligned} PK &= (Y, V, (H_1, \ldots, H_L), g, f, p, \mathbb{G}, \mathbb{G}_T, e), \\ MSK &= w. \end{aligned}$$

▶ **Encrypt**$(PK, M, \overrightarrow{v} = (v_1, \ldots, v_L) \in \Sigma_L^*)$: Assume that $\overrightarrow{v} = (v_1, \ldots, v_L)$ contains $\tau \leq N$ wildcards which occur at positions $J = \{j_1, \ldots, j_\tau\}$. The encryption algorithm chooses $s \in_R \mathbb{Z}_p$, and computes using Viete's formulas $t = a_0$. It then computes:

$$C_0 = MY^s, C_1 = g^{\frac{s}{t}}, C_2 = f^s, C_3 = (\prod_{i=1}^{L} VH_i^{v_i})^{\frac{\prod_{k=1}^{\tau}(i-j_k)s}{t}},$$

and set the ciphertext $CT = (C_0, C_1, C_2, C_3, J = \{j_1, j_2, \ldots, j_\tau\})$.

▶ **Key Generation**$(MSK, \overrightarrow{z} = (z_1, \ldots, z_L) \in \Sigma_L)$: given a key vector $\overrightarrow{z} = (z_1, \ldots, z_L)$, the key generation algorithm chooses $r, r_1 \in_R \mathbb{Z}_p$, then it creates secret key SK as:

$$K_1 = g^r, K_2 = g^{r_1}, \begin{pmatrix} K_{3,0} = w(\prod_{i=1}^{L}(H_i^{z_i}V)^r f^{r_1} \\ K_{3,1} = (\prod_{i=1}^{L} H_i^{z_i}V)^{ir} \\ \cdots \\ K_{3,N} = (\prod_{i=1}^{L} H_i^{z_i}V)^{i^N r} \end{pmatrix}.$$

▶ **Decrypt**(CT, SK): The decryption algorithm first applies the Viete formulas on $J = \{j_1, \ldots, j_\tau\}$ included in the ciphertext to compute:

$$a_{\tau-k} = (-1)^k \sum_{1 \leq i_1 < i_2 < \ldots < i_k \leq \tau} j_{i_1} j_{i_2} \ldots j_{i_k}, \text{ for } 0 \leq k \leq \tau$$

and then outputs:

$$M = \frac{e(K_1, C_3) \cdot e(K_2, C_2)}{e(\prod_{k=0}^{\tau} K_{3,k}^{a_k}, C_1)} \cdot C_0.$$

Correctness

$$\begin{aligned}
e(K_1, C_3) &= e(g^r, ((\prod_{i=1}^{L} VH_i^{v_i})^{\prod_{k=1}^{\tau}(i-j_k)})^{\frac{s}{a_0}}) \\
&= \prod_{i=1}^{L} e(g, V)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, H_i)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)v_i}{a_0}}. \\
e(K_2, C_2) &= e(g^{r_1}, f^s) = e(g, f)^{r_1 s} \\
e(\prod_{k=0}^{\tau} K_{3,k}^{a_k}, C_1) &= e(w^{a_0}(\prod_{k=0}^{\tau} \prod_{i=1}^{L} H_i^{z_i i^k a_k} V^{i^k a_k})^r f^{r_1 a_0}, g^{\frac{s}{a_0}}) \\
&= e(g, w)^{\frac{sa_0}{a_0}} \cdot e(g, f)^{\frac{sr_1 a_0}{a_0}} \cdot \prod_{i=1}^{L} e(g, V)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)}{a_0}} e(g, H_i)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}} \\
&= e(g, w)^s \cdot e(g, f)^{sr_1} \cdot \prod_{i=1}^{L} e(g, V)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g, H_i)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}}.
\end{aligned}$$

Then we have:

$$\frac{e(K_1,C_3) \cdot e(K_2,C_2) \cdot C_0}{e(\prod_{k=0}^{\tau} K_{3,k}^{a_k}, C_1)} = \frac{M \cdot e(g,w)^s \cdot \prod_{i=1}^{L} e(g,V)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g,H_i)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)v_i}{a_0}} \cdot e(g,f)^{r_1 s}}{e(g,w)^s \cdot e(g,f)^{sr_1} \cdot \prod_{i=1}^{L} e(g,V)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)}{a_0}} \cdot e(g,H_i)^{\frac{sr \prod_{k=1}^{\tau}(i-j_k)z_i}{a_0}}} = M.$$

6 Security Proof of CCP-HVE2 Scheme

Theorem 2. *Assume decision L-BDHE assumption holds in $\mathbb{G}$, then our CP-HVE Scheme 2 is secure.*

Proof. Suppose that there exists an adversary A which can attack our scheme with non-negligible advantage ϵ, we construct another algorithm B which uses A to solve the decision L-BDHE problem. On input $(g, h, \overrightarrow{y}_{g,\alpha,L} = (g_1, g_2, \ldots, g_L,$
$g_{L+2}, \ldots, g_{2L}), T)$, where $g_i = g^{\alpha^i}$ and for some unknown $\alpha \in \mathbb{Z}_p^*$. The goal of B is to determine whether $T = e(g_{L+1}, h)$ or not.
In the rest of the proof, we denote $W(\overrightarrow{v}) = \{1 \leq i \leq L | v_i = *\}$ and $\overline{W}(\overrightarrow{v}) = \{1 \leq i \leq L | v_i \neq *\}$, and $W(\overrightarrow{v})|_j^k$ as $\{i \in W(\overrightarrow{v}) | j \leq i \leq k\}$.
B simulates the game for A as follows:

- **Init**: A declares two challenge alphabet vectors $\overrightarrow{v_0^*} \in \Sigma_L^*$ and $\overrightarrow{v_1^*} \in \Sigma_L^*$ under the restriction that $W(\overrightarrow{v_0^*}) = W(\overrightarrow{v_1^*})$. B flips a coin $\mu \in \{0, 1\}$. For simplicity we denote $\overrightarrow{v_\mu^*} = (v_1^*, v_2^*, \cdots, v_L^*)$.
- **Setup**: B chooses $N << L$, and random values $\gamma, y, \psi, u_1, \ldots, u_L \in_R \mathbb{Z}_p$ and sets

$$\begin{aligned}
&Y = e(g^\alpha, g^{\alpha^L} g^\gamma), f = g^\psi, \\
&V = g^y \prod_{i \in \overline{W}(\overrightarrow{v_\mu^*})} g^{\alpha^{L+1-i} v_{\mu,i}^*} \\
&\{H_i = g^{u_i - \alpha^{L+1-i}}\}_{i \in \overline{W}(\overrightarrow{v_\mu^*})}, \{H_i = g^{u_i}\}_{i \in W(\overrightarrow{v_\mu^*})}.
\end{aligned}$$

The master key component w is $g^{\alpha^{L+1} + \alpha\gamma}$. Since B does not have $g^{\alpha^{L+1}}$, B cannot compute w directly.

- **Query Phase 1**: A queries the user secret key for $\overrightarrow{\sigma_u} = (\sigma_1, \sigma_2, \ldots, \sigma_u)$ that does not match the challenge patterns. Let $k \in \overline{W}(\overrightarrow{v_\mu^*})$ be the smallest integer such that $\sigma_k \neq v_{\mu,k}^*$.
  B needs to simulate the user key generation process. We start from $K_{3,i}$.

  $$\begin{aligned} K_{3,0} &= w(\prod_{i=1}^{L} H_i^{\sigma_i} V)^r f^{r_1} \\ &= g^{\alpha^{L+1}+\alpha\gamma}(\prod_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} g^{u_i-\alpha^{L+1-i}} \cdot \prod_{W(\overrightarrow{v_\mu^*})|_1^k} (g^{u_i}))^{\sigma_i} \cdot g^{y+\sum\limits_{\overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^*})^r f^{r_1}. \\ &\overset{def}{=} g^{\alpha^{L+1}+\alpha\gamma}(g^X)^r f^{r_1} \end{aligned}$$

  where

  $$X = \sum_{\overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^* + y + \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i \sigma_i.$$

  Since

  $$\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (u_i - \alpha^{L+1-i})\sigma_i + \sum_{W(\overrightarrow{v_\mu^*})|_1^k} u_i\sigma_i = \sum_{\overline{W}(\overrightarrow{v_\mu^*})|_1^k} (-\alpha^{L+1-i}\sigma_i) + \sum_{i=1}^{k} u_i\sigma_i$$

  and recall $\sigma_i = v_{\mu,i}^*$ for $i \in \overline{W}(\overrightarrow{v_\mu^*})|_1^{k-1}$ and $\sigma_k \neq v_{\mu,k}^*$. Hence, we have

  $$X = \alpha^{L+1-k}\Delta_k + \textstyle\sum_{\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \alpha^{L+1-i} v_{\mu,i}^* + \sum_{i=1}^k x_i\sigma_i + y$$

  where $\Delta_k = v_{\mu,k}^* - \sigma_k$. Then we choose $\hat{r}, r_1$ randomly in $\mathbb{Z}_n$, and set $r = \frac{-\alpha^k}{\Delta_k} + \hat{r}$. $K_{3,0}$ can be represented as

  $$\begin{aligned} &K_{3,0} \\ &= g^{\alpha^{L+1}+\alpha\gamma} \cdot g^{-\alpha^{L+1}} \cdot g^{\sum\limits_{i\in\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \frac{-\alpha^{L+1-i+k} v_{\mu,i}^*}{\Delta_k}} \cdot g^{a^k(-\frac{\sum_{i=1}^k x_i\sigma_i + y.}{\Delta_k})} \cdot (V\prod_{i=1}^{k} h_i^{\sigma_i})^{\hat{r}} \cdot f^{r_1} \\ &= g^{\alpha\gamma} \cdot g^{\sum\limits_{i\in\overline{W}(\overrightarrow{v_\mu^*})|_{k+1}^L} \frac{-\alpha^{L+1-i+k} v_{\mu,i}^*}{\Delta_k}} \cdot g^{a^k(-\frac{\sum_{i=1}^k x_i\sigma_i + y.}{\Delta_k})} \cdot (V\prod_{i=1}^{k} H_i^{\sigma_i})^{\hat{r}} \cdot f^{r_1}. \end{aligned}$$

  For $\hat{k} = 1$ to N, we compute

  $$K_{3,\hat{k}} = (g^{y+\sum\limits_{\overline{W}(\overrightarrow{v_\mu^*})} \alpha^{L+1-i} v_{\mu,i}^*} \cdot (\prod_{\overline{W}(\overrightarrow{v_\mu^*})|_1^{k-1}} g^{u_i-\alpha^{L+1-i}} \cdot \prod_{W(\overrightarrow{v_\mu^*})|_1^{k-1}} (g^{u_i})^{\sigma_i})^{\frac{-\alpha^k i^{\hat{k}}}{\Delta_k} + \hat{r} i^{\hat{k}}}.$$

  Other elements in the key can also be simulated:

  $$K_1 = g^r = (g^{\alpha_k})^{-1/\Delta_k} \cdot g^{\hat{r}}, K_2 = g^{r_1}.$$

- **Challenge**: A sends to message M_0, M_1 to B, then sets using Viete formulas

  $$a_{\tau-k} = (-1)^k \sum_{i\leq i_1<i_2<\ldots<i_k\leq\tau} j_{i_1} j_{i_2} \ldots j_{i_k}, 0 \leq k \leq \tau.$$

  Let $t = a_0$. It creates ciphertext as:

  $$C_0 = M_b \cdot T \cdot e(g^\alpha, h)^\gamma, C_1 = h^{1/t}, C_2 = h^\psi, C_3 = ((h^{y+\sum\limits_{i=1}^{L} u_i v_{\mu,i}^*})^{\prod\limits_{k=1}^{\tau}(i-j_k)})^{\frac{1}{t}}$$

  If $T = e(g,h)^{\alpha^{L+1}}$, the challenge ciphertext is a valid encryption of M_b. On the other hand, when T is uniformly distributed in $\mathbb{G}_T$, the challenge ciphertext is independent of b.

- **Query Phase 2**: Same Phase 1.
- **Guess**: A output $b' \in \{0,1\}$. If $b' = b$ then B outputs 1, otherwise outputs 0.

If $b' = 0$, then the simulation is the same as in the real game. Hence, A will have the probability $\frac{1}{2} + \epsilon$ to guess b correctly. If $b' = 1$, then T is random in $\mathbb{G}$, then A will have probability $\frac{1}{2}$ to guess b correctly. Therefore, B can solve the decision L-BDHE assumption also with advantage ϵ. □

7 Performance Comparison

We give a detailed comparison among all the HVE schemes in Table 2. The schemes are compared in terms of the order of the underlying group, ciphertext size, decryption cost, and security assumption. In the table, p denotes the pairing operation, L the length of the vector, and N denotes the maximum number of wildcards.

Table 2. Performance Comparison

Scheme	Group Order	Ciphertext Size	Decryption Cost	Assumption
Katz et al. [6]	pqr	$(2L+1)\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	$(2L+1)$p	c3DH
Shi–Waters [20]	pqr	$(L+3)\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	$(L+3)$p	c3DH
Ivovino–Persiano[21]	p	$(2L+1)\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	$(2L+1)$p	DBDH + DLIN
Sedghi et al. [8]	p	$(N+3)\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	3p	DLIN
Lee–Dong [25]	pqr	$(L+2)\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	4p	cBDH BSD c3DH
Park [23]	p	$(2L+3)\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	5p	DBDH+DLIN
Hattori et al. [9]	pq	$(2L+3)\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	3p	$L-w$DBDHI BSD $L-c$DDH
CP-HVE1	pq	$4\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	4p	$L-$cBDHE BSD $L-c$DDH
CP-HVE2	p	$3\lvert\mathbb{G}\rvert + 1\lvert\mathbb{G}_T\rvert$	3p	L-BDHE

Remark: In Table 2, we do not count the wildcard positions when measuring the ciphertext size. To indicate those wildcard positions, a naive way is to use an L-bit string, which has the same size as several group elements when L is linear in the security parameter. When $N \ll L$, then a more efficient way is to use the index for the first wildcard position and the offsets for the remaining wildcard positions.

8 Conclusion

We proposed two efficient ciphertext policy Hidden Vector Encryption schemes in this paper. Both of our encryption schemes can achieve constant ciphertext size, which forms the major contribution of this work. We proved the security of our schemes in a selective security model which captures both plaintext and attribute hiding properties. One of our future work is to extend our schemes so that they can achieve adaptive security.

References

1. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM conference on Computer and communications security. CCS ’06, New York, NY, USA, ACM (2006) 89–98
2. Lewko, A.B., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: Advances in Cryptology – EUROCRYPT. (2010) 62–91

3. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: Public Key Cryptography. (2011) 53–70
4. Lewko, A.B., Waters, B.: New proof methods for attribute-based encryption: Achieving full security through selective techniques. In: CRYPTO. (2012) 180–198
5. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Proceedings of the 4th conference on Theory of cryptography. TCC'07, Berlin, Heidelberg, Springer-Verlag (2007) 535–554
6. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Proceedings of the theory and applications of cryptographic techniques 27th annual international conference on Advances in cryptology. EUROCRYPT'08, Berlin, Heidelberg, Springer-Verlag (2008) 146–162
7. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: TCC. (2011) 253–273
8. Sedghi, S., Liesdonk, P., Nikova, S., Hartel, P., Jonker, W.: Searching keywords with wildcards on encrypted data. In Garay, J., Prisco, R., eds.: Security and Cryptography for Networks. Volume 6280 of Lecture Notes in Computer Science., Springer Berlin Heidelberg (2010) 138–153
9. Hattori, M., Hirano, T., Ito, T., Matsuda, N., Mori, T., Sakai, Y., Ohta, K.: Ciphertext-policy delegatable hidden vector encryption and its application to searchable encryption in multi-user setting. In Chen, L., ed.: Cryptography and Coding. Volume 7089 of Lecture Notes in Computer Science., Springer Berlin Heidelberg (2011) 190–209
10. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Advances in Cryptology – CRYPTO. (1984) 47–53
11. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Advances in Cryptology – CRYPTO. (2001) 213–229
12. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: IMA Int. Conf. (2001) 360–363
13. Boneh, D., Boyen, X., Goh, E.J.: Hierarchical identity based encryption with constant size ciphertext. In: EUROCRYPT05. (2005) 440–456
14. Boneh, D., Crescenzo, G.D., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Advances in Cryptology – EUROCRYPT. (2004) 506–522
15. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. J. Cryptology **21**(3) (2008) 350–391
16. Abdalla, M., Catalano, D., Dent, A., Malone-Lee, J., Neven, G., Smart, N.: Identity-based encryption gone wild. In Bugliesi, M., Preneel, B., Sassone, V., Wegener, I., eds.: Automata, Languages and Programming. Volume 4052 of Lecture Notes in Computer Science., Springer Berlin Heidelberg (2006) 300–311
17. Waters, B.: Efficient identity-based encryption without random oracles. In: Advances in Cryptology – EUROCRYPT. (2005) 114–127
18. Boneh, D., Boyen, X.: Efficient selective-id secure identity based encryption without random oracles. In: Proceedings of Eurocrypt 2004, volume 3027 of LNCS, Springer-Verlag (2004) 223–238
19. Abdalla, M., De Caro, A., Phan, D.H.: Generalized key delegation for wildcarded identity-based and inner-product encryption. Information Forensics and Security, IEEE Transactions on **7**(6) (2012) 1695–1706
20. Shi, E., Waters, B.: Delegating capabilities in predicate encryption systems. In: Proceedings of the 35th international colloquium on Automata, Languages and Programming, Part II. ICALP '08, Berlin, Heidelberg, Springer-Verlag (2008) 560–578
21. Iovino, V., Persiano, G.: Hidden-vector encryption with groups of prime order. In: Proceedings of the 2nd international conference on Pairing-Based Cryptography. Pairing '08, Berlin, Heidelberg, Springer-Verlag (2008) 75–88
22. Blundo, C., Iovino, V., Persiano, G.: Private-key hidden vector encryption with key confidentiality. In Garay, J., Miyaji, A., Otsuka, A., eds.: Cryptology and Network Security. Volume 5888 of Lecture Notes in Computer Science., Springer Berlin Heidelberg (2009) 259–277
23. Park, J.H.: Efficient hidden vector encryption for conjunctive queries on encrypted data. IEEE Trans. on Knowl. and Data Eng. **23**(10) (October 2011) 1483–1497
24. Seo, J., Kobayashi, T., Ohkubo, M., Suzuki, K.: Anonymous hierarchical identity-based encryption with constant size ciphertexts. In: Public Key Cryptography – PKC 2009. Volume 5443 of Lecture Notes in Computer Science., Springer Berlin Heidelberg (2009) 215–234
25. Lee, K., Lee, D.H.: Improved hidden vector encryption with short ciphertexts and tokens. Des. Codes Cryptography **58**(3) (March 2011) 297–319
26. Lewko, A.B.: Tools for simulating features of composite order bilinear groups in the prime order setting. In: Advances in Cryptology – EUROCRYPT. (2012) 318–335