1 IND-CPA Security Model

The IND-CPA security model for the AD-SS-VAHN scheme can be described as a game between an attacker $\mathcal{A}$ and a challenger $\mathcal{C}$. In the model, we assume that the $\mathcal{A}$ can corrupt attribute authorities statically, but the key query is adaptive. The security game proceeds as follows.

Initialization: $\mathcal{A}$ announces the challenge access policy $W^* = (M^*, \rho^*, \mathcal{S}^*)$ and a revocation list R^*, where $\mathcal{S}^* = (I_S^*, S^*)$ and I_S^* is the set of user attribute names, $S^* = \{\beta_i^*\}_{i \in I_S^*}$ is the set of user attribute values.

Setup: $\mathcal{C}$ performs the $CASetup$ algorithm and provides $\mathcal{A}$ with the public parameters GP. The $\mathcal{A}$ designates a partially corrupted attribute authorities $I'_A \subset I_A$. For uncorrupted authorities in $I_A - I'_A$, the $\mathcal{C}$ sends only their public keys to $\mathcal{A}$. For corrupted authorities in I'_A, the $\mathcal{C}$ sends both their public and private keys to $\mathcal{A}$.

Query Phase 1: In this phase, the $\mathcal{A}$ can adaptively make the attribute key queries by submitting the attribute sets $(uid_1, S_1), (uid_2, S_2), \ldots, (uid_{Q_1}, S_{Q_1})$, where $S_i = \{S_k\}_{k \in I_A - I'_A}$ is a group of attributes that belongs to several uncorrupted attribute authorities.

- If $S_i \in M^*$ and $uid \notin R^*$, then termination.
- If $S_i \notin M^*$ or $uid \in R^*$, the $\mathcal{C}$ performs $KeyGen$ algorithm and sends SK_i to $\mathcal{A}$.

Challenge: The $\mathcal{A}$ chooses two messages m_0 and m_1 with equal length, and then submits them to $\mathcal{C}$. The $\mathcal{C}$ flips a coin $b \in \{0,1\}$ at random and runs $Offline.Encrypt$ and $Online.Encrypt$ to encrypt the message m_b under the challenge access policy W^* and revocation list R^*. Then $\mathcal{C}$ sends CT to $\mathcal{A}$.

Query Phase 2: This stage is the same as query stage 1.

Guess: The $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ of b. The advantage of the $\mathcal{A}$ is defined as $Adv = \Pr[b' = b] - 1/2$ in this game.

Definition 4: *AD-SS-VAHN is selectively secure if all probabilistic polynomial-time (PPT) attackers have at most a negligible advantage in the above security game.*

2 Traceability Security Model

Similar to the selective security model, the traceability security game proceeds as follows.

Setup: The $\mathcal{C}$ executes the $CASetup$ algorithm and offers the public parameters GP to $\mathcal{A}$.

Key Query: The $\mathcal{A}$ makes the key query by submitting the attribute sets $(uid_1, S_1), \ldots, (uid_Q, S_Q)$, where $S_i = \{S_k\}_{k \in I_A - I'_A}$ is a group of attributes that belongs to several uncorrupted attribute authorities. For each query, the $\mathcal{C}$ performs $KeyGen$ and sends SK_i to $\mathcal{A}$.

Key Forgery: The $\mathcal{A}$ will output a decryption key SK^*_{uid}. If $Trace(GP, SK^*_{uid}) \neq \top$ and $Trace(GP, SK^*_{uid}) \notin \{uid_1, \ldots, uid_Q\}$, then the $\mathcal{A}$ wins the game. The advantage of the $\mathcal{A}$ in this game is defined as $Trace(GP, SK^*_{uid}) \notin \{\top, uid_1, \ldots, uid_Q\}$.

Definition 5: *AD-SS-VAHN is fully traceable if there has no polynomial time attackers have a non-negligible advantage in the above security game.*

3 Security Proof

In accordance with the security model described above, we analyze the security of our AD-SS-VAHN scheme in this section, which can be summed up as the following theorems.

Theorem 1. *if the q-BDHE assumption holds, no polynomial time adversary can selectively break AD-SS-VAHN with a challenge matrix of size $l^* \times n^*$, where $n^* < q$.*

Proof. We assume that there is an adversary $\mathcal{A}$ to break the AD-SS-VAHN scheme with a non-negligible advantage in the game. $\mathcal{A}$ can query any attribute keys and proxy keys that cannot be combined with any keys obtained from corrupted AAs to decrypt the challenge ciphertext. On these conditions, the security game in multi-authority system can be treated in the same way as the security game in single authority system. Therefore, we can build a simulator $\mathcal{B}$ to break the security of the AD-SS-VAHN scheme with the advantage $\varepsilon/2$.

Let G_0 and G_1 be two multiplicative cyclic groups with prime order p and g be a generator of G_0. Let $e : G_0 \times G_0 \rightarrow G_1$ be a bilinear map. Give $Y = \left(g, g^s, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^q}, g^{\alpha^{q+2}}, \ldots, g^{\alpha^{2q}}\right)$. $\mathcal{B}$ flips a coin $\mu = \{0,1\}$. If $\mu = 0$, B calculates $Z = e(g,g)^{\alpha^{\bar{q}+1}}$; if $\mu = 1$, B chooses $Z \in G_T$ randomly.

Init. $\mathcal{A}$ selects a challenge access structure $W^* = (M^*, \rho^*, \mathcal{S}^*)$ and a user revocation list R^*, where M^* is the $l^* \times n^*$ challenge access matrix, $\mathcal{S}^* = (I_S^*, S^*)$, I_S^* is the user attribute names set, $S^* = \left\{\beta^*_{\rho^*(i)}\right\}_{i \in [1, l^*]}$ is the user attribute values set, ρ^* maps a row in M^* into an attribute name in I_S^* that only appears once.

Setup. The $\mathcal{A}$ selects a set of $I'_A \in I_A$ of compromised authorities and sends it to $\mathcal{B}$. For each uncompromised authorities AA_k $(k \in I_A - I'_A)$, the $\mathcal{B}$ executes the following process:

1. Randomly chooses $\alpha'_k \in Z_p$ $(k \in I_A - I'_A)$ and set $e(g,g)^{\alpha_k} = e(g,g)^{\alpha'_k} e(g^d, g^{d^q})$, then $\alpha_k = \alpha'_k + d^{q+1}$.
2. Randomly selects $a_k \in Z_p$ $(k \in I_A - I'_A)$ and computes g^{a_k}, $w = g^d, v = g^{d^q}$.
3. For the user revocation list R^*, let $I_{R^*} = \{i \in \mathrm{path}(u) \mid u \in R^*\}$, randomly selects $v_i \in Z_p$ where $i = 0, 1, \ldots, (3N-1)/2$. If $i \in I_{R^*}$, set $y_i = g^{v_i} g^{d^i}$, then $sk_i = v_i + d^i$; otherwise, set $y_i = g^{v_i} g^{d^q}$, then $sk_i = v_i + d^q$.

Then $\mathcal{B}$ publishes the public parameter $GP = \left\langle G_0, G_1, e, g, w, v, e(g,g)^{\alpha_k}, g^{a_k}, \{y_i\}_{i=0}^{(3N-1)/2} \right\rangle$, where $k \in I_A - I'_A$.

Phase 1. $\mathcal{B}$ answers the key queries from $\mathcal{A}$ with the attribute sets $(uid_1, \mathcal{S}_1), (uid_2, \mathcal{S}_2), \ldots, (uid_{Q_1}, \mathcal{S}_{Q_1})$, where $\mathcal{S}_i = (I_S, S), i \in I_A - I'_A$ is a group of attributes that belongs to several uncorrupted attribute authorities and $S = \{s_i\}_{i \in I_S}$ is the attribute value set. For each $s_i \in S$, if $s_i = \beta^*_{p^*(i)}$, then set $u_i = s_i + \sum_{n=1}^{n^*} d^n M^*_{k,n}$, where $\forall i \in \{1, 2, \ldots, l^*\}$; otherwise set $u_i = s_i$. There are four cases below, where $\mathcal{S} \models W^*$ represents that the $\mathcal{S}$ meets the access policy W^*, and the $\mathcal{S} \not\models W^*$ represents that the $\mathcal{S}$ does not meet the access policy W^*.

Case 1: If $\mathcal{S} \models W^*$ and $uid \notin R^*$, then terminate.

Case 2: If $\mathcal{S} \models W^*$ and $uid \in R^*$, then $\mathcal{B}$ performs the following steps:

1. Randomly chooses $c \in Z_p$. Let $r = -\frac{d^q}{a_k+c} + \frac{d^{q-1}}{a_k+c} \cdot \frac{M_{i,1}}{M^*_{i,2}}$ and computes $K_0, K_1, L_0, L_1, \{K_i\}_{i \in I_S}$:

$$
\begin{aligned}
&K_1 = c,\\
&K_0 = g^{\frac{\alpha'}{a_k+c}} \left(g^{\frac{d^q}{a_k+c}}\right)^{\frac{M^*_{i,1}}{M^*_{i,2}}} = g^{\frac{\alpha_k}{a_k+c}} w^r,\\
&L_0 = \left[\left(g^{d^q}\right)^{\frac{1}{a_k+c}}\right]^{-1} \left[\left(g^{d^{q-1}}\right)^{\frac{1}{a_k+c}}\right]^{\frac{M^*_{i,1}}{M^*_{i,2}}} = g^r,\\
&L_1 = (L_0)^{a_k} = g^{a_k r},\\
&K_i = \left[\left(g^{d^q}\right)^{\frac{s_i}{a_k+c}}\right]^{-1} \cdot \left[\left(g^{d^{q-1}}\right)^{\frac{s_i M^*_{i,1}}{a_k+c M^*_{i,2}}}\right]^{\frac{M^*_{i,1}}{M^*_{i,2}}} \cdot g^{d^{2q}}\\
&\left[\left(\prod_{k=1}^{n^*} g^{d^{q+k} \cdot M^*_{i,k}}\right)^{-1} \cdot \left(\prod_{k=1}^{n^*} g^{d^{q+k-1} \cdot M^*_{i,k}}\right)^{\frac{M^*_{i,1}}{M^*_{i,2}}}\right]^{\frac{1}{a_k+c}}\\
&\cdot \left[\left(g^{d^{2q-1}}\right)^{\frac{M^*_{i,1}}{M^*_{i,2}}}\right]^{-1}\\
&= g^{u_i \cdot r} v^{-(a_k+c)r}
\end{aligned}
$$

2. Suppose that $path(i_d) = \{i_0, \ldots, i_d\}$, where $i_0 = root$ and i_d is the leaf node value in the ternary tree that is related to the user uid. Since $uid \in R^*$, then $i_d \in I_{R^*}, sk_{i_d} = v_{i_d} + d^{i_d}$ is concluded. $\mathcal{B}$ calculates

$$
K_u = \left[\left(g^{d^q}\right)^{-1} \cdot \left(g^{d^{q-1}}\right)^{\frac{M^*_{i,1}}{M^*_{i,2}}}\right]^{\frac{1}{(v_{id}+d^{id}) \cdot (a_k+c)}} = g^{r/sk_{id}}.
$$

Case 3: If $\mathcal{S} \models W^*$ and $uid \in R^*$, $\mathcal{B}$ executes as follows:

1. According to the definition of LSSS, randomly choose a vector $\vec{\omega} = (\omega_1, \omega_2, \ldots, \omega_{n^*})^{\mathsf{T}} \in Z_p^{n^*}$, where $\omega_1 = -1$ and $M_i^* \cdot \vec{\omega} = 0$ for $i \in [2, l^*]$.

2. Select $c \in Z_p$ and set $K_1 = c$,

3. Randomly chooses $h \in Z_p$ and implicitly define $r = \frac{1}{a_k+c}\left(h + \omega_1 d^q + \omega_2 d^{q-1} \cdots + \omega_{n^*} d^{q-n^*+1}\right)$,

4. Calculate K_0, L_0 and L_1 as follows:

$$
\begin{aligned}
&L_0 = g^{\frac{h}{a_k+c}} \prod_{i=1}^{n^*} \left(g^{\omega_i d^{q+1-i}}\right)^{\frac{1}{a_k+c}} = g^r,\\
&L_1 = g^{\frac{a_k h}{a_k+c}} \prod_{i=1}^{n^*} \left(g^{\omega_i d^{q+1-i}}\right)^{\frac{a_k}{a_k+c}} = g^{a_k r},\\
&K_0 = \left(g^{\alpha'_k + dh} \prod_{i=2}^{n^*} g^{\omega_i d^{q+2-i}}\right)^{\frac{1}{a_k+c}} = g^{\frac{\alpha_k}{a_k+c}} w^r,
\end{aligned}
$$

5. For $\forall \tau \in I_S$, if there exists i such that $\rho^*(i) = \tau$ and $s_\tau = \beta^*_{\rho^*(i)}$, then $\mathcal{B}$ computes

$$
\begin{aligned}
K_\tau = L_0^{s_\tau} &\left[\prod_{j=1}^{n^*} \left(g^{t \cdot d^j} \cdot \prod_{k=1}^{n^*} g^{\omega_k d^{q+1+j-k}}\right)^{M^*_{i,j}}\right]^{\frac{1}{a_k+c}}\\
&\cdot \left(g^{t \cdot d^q} \prod_{i=1}^{n^*} g^{\omega_i d^{2q+1-i}}\right)^{-1}.
\end{aligned}
$$

Otherwise, the $K_\tau = L_0^{s_\tau}\left(g^{t \cdot d^q} \prod_{i=1}^{n^*} g^{\omega_i d^{2q+1-i}}\right)^{-1}$.

6. Suppose that $path(i_d) = \{i_0, \ldots, i_d\}$, where $i_0 = root$ and i_d is the leaf node value in the ternary tree that is related to the user uid. Since $uid \in R^*$, then $i_d \in I_{R^*}, sk_{i_d} = v_{i_d} + d^{i_d}$ is obtained. $\mathcal{B}$ computes

$$
K_u = \left(g^t \prod_{i=1}^{n^*} g^{\omega_i d^{q+1-i}}\right)^{1/\left(v_{i_d}+d^{i_d}\right) \cdot (a_k+c)} = g^{r/sk_{i_d}}.
$$

Case 4: If $\mathcal{S} \not\models W^*$ and $uid \notin R^*$, then the calculation process of L_0, L_1, K_0 and K_τ is the same as case 3. Since $uid \notin R^*$, then $i_d \notin I_{R^*}$, $sk_{i_d} = v_{i_d} + d^q$ is obtained. Next, $\mathcal{B}$ calculates $K_u = \left(g^t \prod_{i=1}^{n^*} g^{\omega_i d^{g+1-i}}\right)^{\frac{1}{\left(v_{i_d}+dq\right) \cdot (a_k+c)}} = g^{r/sk_{i_d}}$.

Challenge. $\mathcal{A}$ sends two equal-length keys k_0, k_1 to $\mathcal{B}$. $\mathcal{B}$ performs the following processes:

1. $\mathcal{B}$ tosses a fair coin $b \in \{0, 1\}$ and perform the $Online.Encrypt$ algorithm, and computes $C_2 = k_b \cdot e(g, g)^{\alpha_k s}$.

2. $\mathcal{B}$ performs the $Offline.Encrypt$ algorithm and computes $C_0 = g^s, C_1 = g^{a_k s}$.

3. $\mathcal{B}$ randomly chooses $r_2, \ldots, r_n \in Z_p^*$ and sets $\vec{v} = (s, sd + r_2, sd^2 + r_3, \ldots, sd^{n^*-1} + r_{n^*})^{\mathsf{T}} \in Z_{p^*}^*$, then calculate: $C_{i,1} = \prod_{j=2}^{n^*} \left(g^{dr_j}\right)^{M^*_{i,j}} \prod_{j=1}^{n^*} \left(g^{sd^j}\right)^{M^*_{i,j}} g^{-a_k d^{q+i}}$, $\left(g^{sd^{j-1}}\right)^{M^*_{i,j}}$, $C_{i,2} = \left(g^{t_{\rho^*(i)}}\right)^{-a_k d^i} \prod_{j=2}^{n^*} \left(g^{d^j M^*_{i,j}}\right)^{-a_k d^i} \cdot \prod_{j=2}^{n^*} \left(g^{r_j}\right)^{M^*_{i,j}} \prod_{j=1}^{n^*} \left(g^{sd^{j-1}}\right)^{M^*_{i,j}}$, $C_{i,3} = g^{-a_k d^i}$.

4. For $\forall j \in \text{cover}(R^*)$, since $sk_j = v_j + d^q$ and $y_j = g^{v_j + d^q}$, then $\mathcal{B}$ sets $T_j = (g^s)^{v_j + d^q} = y_j^s$.

Finally, $\mathcal{B}$ sends the ciphertext $CT = \langle C_0, C_1, C_2, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i \in [1, l^*]}, \{T_j\}_{j \in \text{cover}(R^*)} \rangle$ to $\mathcal{A}$.

Phase 2. This stage is the same as stage 1.

Guess. $\mathcal{A}$ eventually output a guess b' of b. If $b = b', \mathcal{B}$ outputs a guess $\mu' = 0$ of μ. If $b \neq b'$, $\mathcal{B}$ outputs a guess $\mu' = 1$ of μ.

1. If $\mu = 0$ then $Z = e(g, g)^{\alpha^{q+1} s}$. The $\mathcal{A}$ can obtain a valid ciphertext. Suppose the advantage of $\mathcal{A}$ is $\varepsilon = \Pr[b = b' \mid \mu = 0] - \frac{1}{2}$, then we can obtain $\Pr[b = b' \mid \mu = 0] = \Pr[\mu = \mu' \mid \mu = 0]$. Thus, the advantage of $\mathcal{B}$ in winning the game is $\Pr[\mu = \mu' \mid \mu = 0] = \varepsilon + \frac{1}{2}$.

2. If $\mu = 1$ then Z is a randomly chosen number from G_1. In this case, $\mathcal{A}$ cannot get any information about b. In such case, The advantage of $\mathcal{A}$ is $\Pr[b \neq b' \mid \mu = 1] = \frac{1}{2}$, then we can obtain $\Pr[b \neq b' \mid \mu = 1] =$

$\Pr[\mu=\mu' \mid \mu=1]$. Therefore, the advantage of B in winning the game is $\Pr[\mu=\mu' \mid \mu=1]=\frac{1}{2}$.

Finally, the advantage of $\mathcal{B}$ in solving q-BDHE hardness assumption is

$$\begin{aligned}\Pr[\mu=\mu'] =& \Pr[\mu=\mu' \mid \mu=0]\cdot\Pr[\mu=0] \\ &+\Pr[\mu=\mu' \mid \mu=1]\cdot\Pr[\mu=1]-\frac{1}{2} \\ =&\left(\varepsilon+\frac{1}{2}\right)\cdot\frac{1}{2}+\frac{1}{2}\cdot\frac{1}{2}-\frac{1}{2}=\frac{1}{2}\varepsilon\end{aligned}$$

□

Theorem 2. *If the l-SDH hardness assumption holds, the AD-SS-VAHN scheme is fully traceable based on $q<l$ in Definition 5, where q is the number of times the adversary queries the key.*

Proof. Assume that there is an adversary $\mathcal{A}$ with a non-negligible advantage ε in the traceability game. We can build a simulator $\mathcal{B}$ to break the security of the AD-SS-VAHN scheme with the same advantage. *Setup.* The $\mathcal{A}$ selects a set of $I'_A \in I_A$ of compromised authorities and sends it to $\mathcal{B}$. Then $\mathcal{B}$ performs the following processes:

1. For each uncorrupted authorities $AA_k\,(k\in I_A-I'_A)$, the $\mathcal{B}$ randomly chooses $\alpha_k, a_k \in Z_p\,(k\in I_A-I'_A)$.

2. $\mathcal{B}$ randomly selects $c_1,c_2,\ldots,c_q \in Z_p^*$ and sets $f(y)=\prod_{i=1}^{q}(y+c_i)=\sum_{i=0}^{q}\beta_i y^i$, where $\beta_i\in Z_p, i=0,1,\ldots,q$.

3. Let $A_i=g_1^{a_k^i}$ and $g=\prod_{i=0}^{q}(A_i)^{\beta_i}=g_1^{f(a_k)}$, then $g^{a_k}=\prod_{i=1}^{q+1}(A_i)^{\beta_{i-1}}=g_1^{f(a_k)\cdot a_k}$,

4. Randomly selects $w,v\in G$, $\theta\in Z_p$. For each leaf node in the ternary tree T, randomly chooses $\{sk_i\}_{i=1}^{(3N-1)/2}\in Z_p$ and compute $\left\{y_i=g^{sk_i}\right\}_{i=1}^{(3N-1)/2}$.

Finally, the $\mathcal{B}$ publish the public parameter $GP=\left\langle G_0,G_1,e,g,w=g^{\theta},v,e(g,g)^{\alpha_k},g^{a_k},\{y_i\}_{i=0}^{(3N-1)/2}\right\rangle$, where $k\in I_A-I'_A$.

Key query: $\mathcal{B}$ answers the key queries from $\mathcal{A}$ with the attribute sets $(uid_1,\mathcal{S}_1),(uid_2,\mathcal{S}_2),\ldots,(uid_{Q_1},\mathcal{S}_{Q_1})$, where $\mathcal{S}_i=(I_S,S), i\in I_A-I'_A$ is a group of attributes that belongs to several uncorrupted attribute authorities and $S=\{s_i\}_{i\in I_S}$ is the attribute value set. For the i-th query, $i<q$, the $\mathcal{B}$ sets $f_i(y)=\frac{f(y)}{y+c_i}=\prod_{j=1,j\neq i}^{q}(y+c_j)=\sum_{j=0,j\neq i}^{q-1}\lambda_j y^j$, where $\lambda_j\in Z_p, j=0,1,\ldots,q-1$ are the coefficients of $f_i(y)$. $\mathcal{B}$ calculates $\sigma_i=\prod_{j=0}^{q-1}(A_j)^{\lambda_j}=g_1^{f_i(a_k)}=g_1^{\frac{f(a_k)}{a_k+c_i}}=g^{\frac{1}{a_k+c_i}}$. $\mathcal{B}$ randomly chooses $r\in Z_p$ and computes the attribute key component that is related to the $(uid_i,\mathcal{S}_i)$

$$\left\langle K_0=(\sigma_i)^{\alpha_k}w^r=g^{\frac{\alpha_k}{a_k+c_i}}w^r, L_1=g^{a_k r}, L_0=g^r, \left\{K_i=g^{s_i\cdot r}(v^{a_k}\cdot v^{c_i})^{-r}=g^{s_i\cdot r}v^{-(a_k+c_i)r}\right\}_{i\in I_S}\right\rangle.$$

Suppose that $path(i_d)=\{i_0,\ldots,i_d\}$, where $i_0=root$ and i_d is the leaf node value in the ternary tree that is related to the user uid. The attribute key component that is related to the user uid is computed as $K_{uid_i}=g^{r/sk_{i_d}}$. At last, the $\mathcal{B}$ sends the finally decryption key $SK_{uid}=\left\langle K_0,K_1,L_0,L_1,K_u,\{K_i\}_{i\in I_S},\{sk_i\}_{i\in path(id)}\right\rangle$ to $\mathcal{A}$.

Key Forgery. The $\mathcal{A}$ outputs a attribute key SK^*_{uid}. If $Trace(GP,SK^*_{uid})\notin\{uid_1,\ldots,uid_Q\}$ and $Trace(GP,SK^*_{uid})\neq\top$, then the $\mathcal{A}$ wins the game with the advantage ε. There are two cases:

1. If ε does not occurs, $\mathcal{B}$ chooses $(c_r,w_r)\in Z_p\times G$ randomly as the solution of l-SDH problem.

2. If ε occurs, $\mathcal{B}$ sets a polynomial $f(y)=\varphi(y)(y+K_1)+\varphi-1$, where $\varphi(y)=\sum_{i=0}^{q-1}\varphi_i y^i$ and $\varphi-1\in Z_p^*$. Since $f(y)=\prod_{i=1}^{q}(y+c_i), c_i\in Z_p^*$ and $K_1\notin\{c_1,c_2,\ldots,c_q\}$, the $(y+K_1)$ cannot divide $f(y)$. Assume that $L_0=g^r$ and $r\in Z_p$ is unknown, we can obtain $L_1=g^{a_k r}$ with $e(g,L_1)=e(g^{a_k},L_0)$ and then get $K_0=g^{\frac{\alpha_k}{a_k+K_1}}w^r$ with $e\left(K_0,g^{a_k}g^{K_1}\right)=e(g,g)^{\alpha_k}e\left(L_0^{K_1}\cdot L_1,w\right)$.

Let $\eta=\left(K_0/L_0^{\theta}\right)^{\alpha_k^{-1}}=g^{\frac{1}{a_k+K_1}}=g_1^{\frac{f(a_k)}{a_k+K_1}}=g_1^{\varphi(a_k)}g_1^{\frac{\varphi-1}{a_k+K_1}}$. Then $\mathcal{B}$ can compute (c_r,w_r) by $c_r=K_1 \bmod \mathrm{p}\in \mathrm{Z}_p$ and $w_r=\left(\eta\cdot\prod_{i=0}^{q-1}A_i^{-\varphi_i}\right)^{\frac{1}{\psi-1}}=g_1^{\frac{1}{a_k+K_1}}$. Since $e(g_1^{a_k}\cdot g_1^{c_r},w_r)=e\left(g_1^{a_k}\cdot g_1^{K_1},g_1^{\frac{1}{a_k+K_1}}\right)=e(g_1,g_1)$, the (c_r,w_r) is the solution to l-SDH problem.

Denote ε as the advantage of $\mathcal{A}$ in winning the traceability game. If $\mathcal{B}$ chooses $(c_r,w_r)\in Z_p\times G$ randomly as the solution of l-SDH problem, then $\varepsilon=0$. If $\mathcal{B}$ outputs the solution of the l-SDH problem, the probability of (c_r,w_r) that satisfies the equality $e(g_1^a\cdot g_1^{c_r},w_r)=e(g_1,g_1)$ is 1 in the case of (Adv wins $\wedge\gcd(\gamma-1,p)=1$). The probability of $\mathcal{B}$ playing the the l-SDH problem is

$$\begin{aligned}\Pr[\varepsilon_{SDH}] =& \Pr\left[\varepsilon_{SDH}\mid\overline{Adv\text{ wins}}\right]\cdot\Pr[\overline{Adv\text{ wins}}] \\ &+\Pr[\varepsilon_{SDH}\mid Adv\text{ wins}\ \wedge\gcd(\gamma-1,p)\neq 1] \\ &\cdot\Pr[\text{Adv wins}\ \wedge\gcd(\gamma-1,p)\neq 1] \\ &+\Pr[\varepsilon_{SDH}\mid Adv\text{ wins}\ \wedge\gcd(\gamma-1,p)=1] \\ &\cdot\Pr[\text{Adv wins}\ \wedge\gcd(\gamma-1,p)=1] \\ =&0+0+1\cdot\Pr[\text{Adv wins}\wedge\gcd(\gamma-1,p)=1] \\ =&\Pr[\text{Adv wins}]\cdot\Pr[\gcd(\gamma-1,p)=1] \\ =&\varepsilon\end{aligned}$$

Therefore, if $\mathcal{A}$ can win the traceability game with a non-negligible advantage, $\mathcal{B}$ can break the security of the AD-SS-VAHN scheme with the same advantage.□

Theorem 3. *The AD-SS-VAHN scheme is secure against the collusion attack of users.*

Proof. The collusion attacks of users include the collusion between existing users and existing users and the collusion between existing users and revoked users. For the first case, each user in the system is distributed a globally unique identifier uid. The attribute keys generated by each AA will be bound together according to the user identity uid. In addition, the attribute keys issued by each AA are associated with its private key. Therefore, it is impossible for two or more existing users to collude with each other to decrypt the ciphertext. For the second case, if an existing user divulges his/her attribute key to the revoked user uid, since $uid\in R'$, the revoked user will not pass the

verification of RSUs, and RSUs will not perform the outsourced decryption for him/her. Even if the revoked user decrypts by himself, there is no node j such that $j \in cover(R') \cap path(uid)$. Therefore, the revoked user cannot perform the decryption operation and obtain the message m. Besides, the key leaker can be traced by white-box traceability and held accountable. Therefore, it is impossible for existing users and revoked users to launch the collusion attack. □