S1 Appendix

Proof of the Theorem 2

The construction of the simulator mentioned in theorem 1 is described as follows:

Setup. The simulator first sets an integer \( u = 4q \), chooses the integer \( k \in R \{0, 1, 2, ..., n\} (n = |A|) \), integer \( x', x_{ij} \in R \{0, 1, ..., u - 1\} \). Meanwhile, it chooses \( y', y_{ij} \in R \mathbb{Z}_p, 1 \leq i \leq n, 1 \leq j \leq m \). All these parameters are kept by the simulator itself. For an attribute list \( L = \{l_1, l_2, ..., l_q\} \), we define \( F(L) = (p - uk) + x' + \sum_{a_{ij} \in L} x_{ij} \), \( J(L) = y' + \sum_{a_{ij} \in L} y_{ij} \), and

\[
K(L) = \begin{cases} 
0, & x' + \sum_{a_{ij} \in L} x_{ij} \equiv 0 \pmod{u} \\
1, & \text{otherwise.}
\end{cases}
\]

For a given asymmetric DBDH tuple \((g, g^b, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, Z)\), the simulator sets \( \hat{g}_1 = \hat{g}^a, \hat{g}_2 = \hat{g}^b, g_2 = g^b \). Then the public parameter \( \hat{f} = \hat{g}_2^2 - uk + x' \hat{g}^{y'} \), \( f = g_2^2 - uk + x' g^{y'}, \Psi_{ij} = g_2^{a_{ij} x_{ij}} \). The \( \Psi_{ij} \) implies \( \psi_{ij} = bx_{ij} + y_{ij} \).

Phase 1. The adversary \( A \) will issue private key queries. Suppose \( A \) issues a query for an private key associated with attribute list \( L \). If \( K(L) = 0 \) simulator aborts and randomly chooses its guess \( \beta' \) of the challenger’s value \( \beta \). Otherwise, it chooses \( r \in R \mathbb{Z}_p^* \), and computes

\[
D_0 = g_1^{-1/F(L)} \hat{g}^r, D_1 = g_1^{-J(L)/F(L)} (\hat{f} \cdot g_2^{\sum_{a_{ij} \in L} x_{ij}} \sum_{\psi_{ij} \in L} y_{ij})^r.
\]

Let \( \tilde{r} = r - a/F(L) \), we can get

\[
D_0 = g_1^{-1/F(L)} \hat{g}^{\tilde{r}} = \hat{g}^{r - a/F(L)} = \hat{g}^{\tilde{r}}
\]

\[
D_1 = g_1^{-J(L)/F(L)} (\hat{f} \cdot g_2^{\sum_{a_{ij} \in L} x_{ij}} \sum_{\psi_{ij} \in L} y_{ij})^r
\]

\[
= g^{ab - aJ(L)/F(L)} (\hat{g}^{b(p - uk + x')} \hat{g}^{b \sum_{\psi_{ij} \in L} x_{ij}} + y' \sum_{\psi_{ij} \in L} bx_{ij} + y_{ij})^r
\]

\[
= g^{ab} (\hat{g}^{bF(L)} \hat{g}^{J(L)})^{-a/F(L)} (\hat{g}^{bF(L)} \hat{g}^{J(L)})^r
\]

\[
= g_2^a (g_2^{-F(L)} \hat{g}^{F(L)} \hat{g}^{J(L)})^{-a/F(L)}
\]

\[
= g_2^{\tilde{r}} (\hat{g}^{a_{ij} \in L} \psi_{ij})^r.
\]

This simulator will be able to perform this computation iff \( F(L) \neq 0 \pmod{p} \). For ease of analysis, the simulator will only continue (not abort) in the sufficient condition where \( K(L) \neq 0 \) (If we have \( K(L) = 0 \) this implies \( F(L) \neq 0 \pmod{p} \) since we can assume \( p > nm \) for any reasonable values of \( p, n, \) and \( m \).)
Challenge. The adversary $A$ next will submit two equal length messages $M_0, M_1$ and two access policies $W_0, W_1$. The simulator flips two fair binary coin $r$ and $e$. If $x' + \sum_{a_{ij} \in W_r} x_{ij} \neq uk$, the simulator will abort and give a random guess $\beta'$. Otherwise, $F(W_r) \equiv 0 \pmod p$, and the simulator outputs the ciphertext $T_c = \{ZM_r, C_0, C_0^{f(W_r)}\}$, in that $C_0 = g^r$,

\[
C_0^{f(W_r)} = g^{c(y' + \sum_{a_{ij} \in W_r} y_{ij})} = g^{c(y' + \sum_{a_{ij} \in W_r} y_{ij} + b(p - uk + x' + \sum_{a_{ij} \in W_r} x_{ij}))} = (g^{b(p - uk + x')}g^{y'}g^{\sum_{a_{ij} \in W_r} x_{ij}}g^{\sum_{a_{ij} \in W_r} y_{ij}})^c = (f \prod_{a_{ij} \in W_r} \psi_{ij})^c.
\]

Clearly, if $Z = e(g, \tilde{g})^{abc}$, the $T_c$ is a valid encryption of $M$ associated with $W_r$. Otherwise, $Z$ just is a random element from $G_T$, and the adversary is impossible to get any information about $r$ and $e$.

Phase 2. The adversary $A$ continues to issue private key queries with the requirement that the attribute list can not satisfy $W_0, W_1$. The simulator repeats the same method used in Phase 1.

Guess. The adversary $A$ outputs a guess $(r', e')$ of $(r, e)$

Artificial Abort. If the simulator has not aborted at this point it will check to see if the adversary’s guess $r = r'$ and $e = e'$. If $r' = r$ and $e = e'$ then the simulator outputs a guess $\beta' = 1$, otherwise it outputs $\beta = 0$.

Intuitively, under the condition of $Z = e(g, \tilde{g})^{abc}$, if $A$ can always successfully guess $r'$ within a probability polynomial time, it means the simulator can resolve the asymmetric DBDH problem within a probability polynomial time as well, which contradicts to the assumption that the asymmetric DBDH is NP-hard. This simulator is difficult to be analyzed directly since it might abort before all of the queries are made. The key point is that we should figure out an appropriate estimate of the simulator abort probability because

\[
Pr[\beta' = \beta] = Pr[\beta' = \beta | abort] Pr[abort] + Pr[\beta' = \beta | \overline{abort}] Pr[\overline{abort}].
\]

Hereafter, we adopt the same proof method as in literature [22] and give out the same conclusion mentioned in theorem 1. Since the complete proof process is longer, we omit it here due to space limitation and direct the interested reader to [22].