Non-monotonic Practical ABE with Direct Revocation, Blackbox Tracability, and a Large Attribute Universe

Dirk Thatmann$^{0000-0002-6646-281X}$,
dirk.thatmann@alumni.tu-berlin.de
Technische Universität Berlin,
Telekom Innovation Laboratories

September 20, 2019

1 Abstract

This work shows all necessary calculations to extend the "Practical Attribute Based Encryption: Traitor Tracing, Revocation, and Large Universe" scheme of Liu and Wong with non-monotonic access structures. We ensure that the blackbox tracability property is preserved.

2 Introduction

We selected Liu et al.'s "Augmented R-CP-ABE" scheme [1, 2] as the technical foundation for a data encryption service. In the course of the requirements analysis it turned out that the property of the non-monotonic access structures is needed, but is not provided by Liu's original scheme. This work uses the techniques introduced by Yamada et al. in [3] which build on top of Ostrovsky et al. [4] to retrofit the property of the non-monotonic access structure, as it was already outlined by Thatmann in [5].

The remainder is structured as follows: we start with an overview of terms and symboles in Section 2.1. New required computations to archive the non-monotonic property and blackbox tracability property are subject of Section 3 and 4. An evaluation is given in Section 5.

2.1 Overview of terms and symbols

The terms, and symbols listed in Table 1 are intended to help get an easier access to the mathematical description of the Attribute-based Encryption (ABE) scheme.

Symbols:	Description
$\mathbb{A}$	$\mathbb{A} = (A, \rho)$ is an LSSS matrix. A is an $l \times n$ matrix. ρ maps each row A_k of A to an attrib. $\rho(k) \in \mathbb{U} = \mathbb{Z}_p$
R	$R = \subseteq [m, m]$ is a revocation list.
$\mathcal{U}$	Attribute Universe, $\mathcal{U} \in \mathbb{Z}_p$
S	Attribute $S \in \mathbb{Z}_p$
PP	Public Parameters, can be seen as equivalent to a public key
ω_k	$\omega_k \in \mathbb{Z}_l$ is a set of reconstruction constants. $k \in [l]$ with l being the row index number of the LSSS matrix (see $\mathbb{A}$ description)
e	e is an bilinear map
M; ct	M is a plaintext message. ct means ciphertext
T_i	T_i is a variable and linked to the user-index matrix at row i. T_i is also an indicator for the position of the plaintext in the ciphertext.
D_p	D_P is calculated from many mappings e and ensures that the attributes S of a private key SK can solve the access structure $\mathbb{A}$. D_P is part of the decryption. (D=Decryption and P=Policy)
D_l	D_I is calculated from many mapping e ensures that the hidden user-index (i, j) and the revocation list $\mathbb{R}$ are taken into account during decryption. (D=Decryption and I=Index)
$\vec{\chi}$	$\vec{\chi}$ is a vector. This vector is the core element of the traceability function.
N	total number of users in the ABE system
m^2	m is the user-index matrix size. It depends of the given amound of users N.
λ	the security parameter, curve parameters

Table 1: terms amd symbols

3 Extended computations for non-monotonic access structures

We decided to use the notation applied by Liu et al. in [2]. Next, we recapture the Practical Attribute-based Encryption (PABE)'s Augmented R-CP-ABE construction and emphasize the modifications required for achieving the non-monotonic and unbounded access structures property. The blue colored parts indicate the new additional elements and computations required to achieve the non-monotonic property. Black colored formulas indicate the original PABE construction. Color highlighting is only used at the beginning of each method or at selected locations.

3.1 Setup method

$$Setup\ (\lambda, N = m^2) \rightarrow (PP, MSK)$$

The Setup method uses the group generator $G(\lambda)$ and gets further (e, p, G, G_T) as parameters. e is a bilinear map and p the prime order of $\mathbb{G}$ and $\mathbb{G}_T$. $\mathbb{G}$ represents the source group and $\mathbb{G}_T$ the target group of the mapping. The attribute universe is $\mathcal{U} = \mathbb{Z}_p$ and λ is the security parameter. N defines the total number of users in the system. From a technical point of view, a matrix must now be created in which all users can be accommodated. As an example we choose $90 \leq N \leq 100$ then m is set to 10 which leads to a 10×10 matrix for the user index.

The algorithm randomly chose

$$g, h, f, f_1, ..., f_m, G, H \in \mathbb{G}; \{a_i, r_i, z_i \in \mathbb{Z}_p\}_{i\in[m]}, \{c_j \in \mathbb{Z}_p\}_{j\in[m]}$$

and outputs the Master Secret Key (MSK) and the system's Public Parameter (PP):

$$MSK_n = MSK \bigcup \{b \in \mathbb{Z}_p\}$$
$$PP_n = PP \bigcup \{G' = H^b\}$$

3.1.1 KeyGen method

$$Keygen\ (PP, MSK, S \subseteq \mathbb{Z}_p) \rightarrow SK_{(i,j),S}$$

The KeyGen method creates a secret key SK by using the Public Parameters PP, a set of attributes S and the MSK a secret key SK. The $\left\{\delta'_{i,j,x} \in \mathbb{Z}_p\right\}_{\forall x\in S}$ should be chosen in a way that $\delta'_{i,j,x_1} + \cdots + \delta'_{i,j,x_k} = \sigma_{i,j}$ with $k = |S|$ applies. Hereby, every delta δ represents an attribute and the $+$ character expresses the group operator. In our case the scalar product. The KeyGen method sets a counter $c = 0$ an calculates the corresponding index (i, j) with $1 \leq i, j \leq m$ and $(i - 1) * m + j = c$. By this all created Secret Key (SK) contain the index (i, j).

We use Yamada et al.'s approach to key generation, which involves the random generation of a set of variables $\{\delta'_{i,j,x}\}_{\forall x\in S} \in \mathbb{Z}_p$. These variables have the following property:

$$\forall x \in S: \quad \delta'_{i,j,1} + ... + \delta'_{i,j,x} = \sigma_{i,j} \tag{1}$$

The KeyGen method outputs the secret key as follows:

Choose $\left\{\delta'_{i,j,x} \in \mathbb{Z}_p\right\}_{\forall x\in S}$ such that $\delta'_{i,j,x_1} + \cdots + \delta'_{i,j,x_k} = \sigma_{i,j}$ with $k = |S|$.

$$SK_n = SK \bigcup \left\{\widetilde{K}_{i,j,x}, \widetilde{K}'_{i,j,x}\right\}_{\forall x\in S}$$

with $\widetilde{K}_{i,j,x} = g^{b\delta'_{i,j,x}}$ and $\widetilde{K}'_{i,j,x} = \left(H^{bx}h^b\right)^{\delta'_{i,j,x}}$.

The calculation of the private keys are identical to those of Liu and Wong.

$$\begin{aligned}
&K_{i,j} = g^{\alpha_i} g^{r_i c_j} (f f_j)^{\sigma_{i,j}}, \quad K'_{i,j} = g^{\sigma_{i,j}}, \quad K''_{i,j} = Z_i^{\sigma_{i,j}}, \\
&\{\overline{K}_{i,j,j'} = f_j^{\sigma_{i,j}}\}_{j'\in[m]\setminus\{j\}} \\
&\{K_{i,j,x} = g^{\delta_{i,j,x}}, \quad K'_{i,j,x} = (H^x h)^{\delta_{i,j,x}} G^{-\sigma_{i,j}}\}_{x\in S},
\end{aligned}$$

The additional variables $\widetilde{K}_{i,j,x}$ and $\widetilde{K}'_{i,j,x}$ are calculated as follows:

$$\{\widetilde{K}_{i,j,x} = g^{b\delta'_{i,j,x}}, \quad \widetilde{K}'_{i,j,x} = (G'^x h^b)^{\delta'_{i,j,x}}\}_{x\in S}$$

Both variables $\widetilde{K}_{i,j,x}$ and $\widetilde{K}'_{i,j,x}$ are now added (union) to the private key $SK_{(i,j),S}$ which looks now like this:

$$SK_{(i,j),S} = \big((i,j), S, K_{i,j}, K'_{i,j}, K''_{i,j}, \{\overline{K}_{i,j,j'}\}_{j'\in[m]\setminus\{j\}}, \{K_{i,j,x}, K'_{i,j,x}, \widetilde{K}_{i,j,x}, \widetilde{K}'_{i,j,x}\}_{x\in S}\big)$$

3.1.2 Encrypt method

$$Encrypt(PP, M, R, \mathbb{A} = (A,\rho), (\bar{i},\bar{j})) \rightarrow CT_{R,(A,\rho)}$$

The encrypt method encrypts a plaintext message M with the help of the Public Parameter PP under consideration of an attribute revocation list R. The access structure $\mathbb{A}$ must be defined beforehand. This boolean formula secures the encrypted data in Ciphertext-policy Attribute-based Encryption (CP-ABE) schemes. For all attributes $x \in S$ it has to be checked whether x is prime (negated) and then set P_k accordingly with $\rho(k) = x$:

$$P_k = \begin{cases} f^{A_k \cdot \mathrm{u}} G^{\xi_k} & \text{if } \rho(k) = x \\ f^{A_k \cdot \mathrm{u}} (G')^{\xi_k} & \text{if } \rho(k) = x' \end{cases}$$

The following calculations are carried out in the preparatory phase of the encryption. They are identical to [2]:

$$\begin{aligned} \kappa, \quad \tau, \quad s_1, ..., s_m, \quad t_1, ..., t_m &\in \mathbb{Z}_p, \\ v_c, \quad w_1, ..., w_m &\in \mathbb{Z}_p^3, \\ \varepsilon_1, ..., \varepsilon_l &\in \mathbb{Z}_p, \\ u = (\pi, u_2, ..., u_n) &\in \mathbb{Z}_p^n, \\ r_x, r_y, r_z &\in \mathbb{Z}_p. \end{aligned}$$

With the three prime numbers r_x, r_y, and r_z the vectors $\vec{\chi_1}, \vec{\chi_2}, \vec{\chi_3}$ can be calculated, which are needed for the Blackbox Traceability functionality.

$$\begin{aligned} &\vec{\chi_1} = (r_x, 0, r_z) \\ &\vec{\chi_2} = (0, r_y, r_z) \\ &\vec{\chi_3} = \vec{\chi_1} \times \vec{\chi_2} = (-r_y r_z, \quad -r_x r_z, \quad r_x r_y) \end{aligned}$$

The user-index $(\bar{i},\bar{j})$ can be used to calculate v_i, the set of all finite linear combinations or linear span.

$$\begin{gathered} \forall i \in \{1, ..., \bar{i}\} : v_i \in \mathbb{Z}_p^3, \\ \forall i \in \{\bar{i}+1, ..., m\} : v_i \in span\{\vec{\chi_1}, \vec{\chi_2}\} \end{gathered}$$

3.2 Ciphertext Construction

Given v_i we can create a ciphertext. The construction is a two-step process because we have to perform a row and a column calculation on the user-index matrix and a second calculation on the LSSS matrix.

1. Calculation on user-index matrix
   For the rows and columns calculations we have to consider all cases for $i \geq \bar{i}$ and $i < \bar{i}$. For each row i with $(1 \leq i \leq m)$ of the user-index matrix with size m we calculate R_i, R'_i, Q_i, Q'_i, Q''_i, and T_i as follows:

   - if $i < \bar{i}$ choose $\tilde{s}_i \in \mathbb{Z}_p$ randomly and calculate

   $$\begin{aligned} &R_i = g^{v_i}, \quad R'_i = g^{\kappa v_i}, \\ &Q_i = g^{s_i}, \quad Q'_i = (f \prod_{j' \in \overline{R}_i} f'_j)^{s_i} Z_i^{t_i} f^{\pi}, \quad Q''_i = g^{t_i}, \\ &T_i = E_i^{\tilde{s}_i} \end{aligned}$$

   - if $i \geq \bar{i}$ calculate

   $$\begin{aligned} &R_i = G_i^{s_i v_i}, \quad R'_i = G_i^{\kappa s_i v_i}, \\ &Q_i = g^{\tau s_i (v_i \cdot v_c)}, \quad Q'_i = (f \prod_{j' \in \overline{R}_i} f'_j)^{\tau s_i (v_i \cdot v_c)} Z_i^{t_i} f^{\pi}, \quad Q''_i = g^{t_i}, \\ &T_i = M \cdot E_i^{\tau s_i (v_i \cdot v_c)} \end{aligned}$$

   For each column j of the user-index matrix $(1 \leq j \leq m)$, calculate C_j and C'_j as follows:

   - if $j < \bar{j}$ choose $\mu_j \in \mathbb{Z}_p$ randomly and calculate

   $$\begin{aligned} &C_j = H^{\tau(v_c + \mu_j \vec{\chi_3})} \cdot g^{\kappa \omega_j}, \\ &C'_j = g^{\omega_j} \end{aligned}$$

   - if $j \geq \bar{j}$ calculate:

   $$\begin{aligned} &C_j = H^{\tau v_c} \cdot g^{\kappa \omega_j}, \\ &C'_j = g^{\omega_j} \end{aligned}$$

2. Calculation on the LSSS matrix
   For each row k of the LSSS matrix with size l $(1 \leq k \leq l)$ we calculate: The introduction of the non-monotonic access rule has an effect on the calculations of the LSSS matrix. There must be a case distinction. As described above, $(p(k) = x)$ with $f^{A_k \cdot u} G^{\epsilon_k}$ must be used for all monotonic access rules and $(p(k) = x')$ with $f^{A_k \cdot u} G'^{\epsilon_k}$ must be used for all non-monotonic access rules. It follows:

(a)

$$P_k = \begin{cases} f^{A_k \cdot \mathrm{u}} G^{\xi_k} & \text{if } \rho(k) = x \\ f^{A_k \cdot \mathrm{u}} (G')^{\xi_k} & \text{if } \rho(k) = x' \end{cases}$$

(b)

$$P'_k = (H^{\rho(k)} h)^{-\epsilon_k}$$

(c)

$$P''_k = g^{\epsilon_k}$$

The ciphertext now contains one more element: P_k

$$CT_{R,\mathbb{A}=(A,\rho)} = \left(R, (A,\rho), (R_i, R'_i, Q_i, Q'_i, Q''_i, T_i)_{i=1}^{m}, (C_j, C'_j)_{j=1}^{m}, (P_k, P'_k, P''_k)_{k=1}^{l} \right)$$

3.3 Decrypt method

The decryption method gets as arguments the PP, the ciphertext, the secret attribute key(s) $SK_{(i,j),S}$. If the attributes S can solve the access structure $\mathbb{A} = (A_{(l \times n)}$ (evaluation to true) the decryption works. Otherwise $\perp$ follows.

$$Decrypt_A(PP, CT_{R,(\mathbb{A}=(A,p))}, SK_{i,j}, S) \rightarrow M or \perp$$

To determine whether the attributes can solve the access structure, reconstruction constants $\{\omega_k \in \mathbb{Z}_p\}_{k \in [l]}$ must be included in the calculation. These constants have the following property:

$$\sum_{p(k) \in S} \omega_k A_k = (1, 0, ..., 0) \tag{2}$$

The constants can not be calculated if the set of attributes S does not satisfy the access policy $\mathbb{A}$. In essence, this decryption procedure lies upon the equation

It is only possible to calculate the constants if the private key with its attributes can solve the access structure. We use the following formula to reconstruct the plaintext, the message M, just like Liu and Wong do (compare [2, p.21]).

$$M = \frac{T_I}{D_P \cdot D_I} \tag{3}$$

As with Liu et al., D_p is the part of the equation that ensures that the attributes S of the private key can solve the access structure $\mathbb{A} = (A, p)$ of the ciphertext. D_I is responsible for the fact that the excluded subsets of user indexes can no longer decrypt. Both D_p and D_I are results of many calculated pairings in the context of bilinear maps calculations. The variable T_I is connected to the user-index at row I. It also indicates the position at which the message M, i.e. the plain text, is embedded in the ciphertext.

In order to get a holistic understanding of the decryption and to prove its correctness, we next present the reconstruction of the plaintext M in detail.

In his master thesis [6], Alwin Alwin documented in detail the calculation steps with all mathematical transformations for the calculation of D_P and D_I. We now repeat his very detailed description.

3.4 Calculating D_p

The calculation of D_P differs from the original Liu scheme, because the non-monotonic attributes have been added. This can be seen in the new calculation of $D_{P_{part-1}}$, where we have to process negated and non-negated attributes separately.

$$D_P = \prod_{p(k)\in S} \left(e(K'_{i,j}, P_k) \quad D_{P_{\text{part-1}}} \right)^{\omega_k}$$

The negated attributes x' and normal attributes x must be considered. The calculation of $D_{P_{\text{part-1}}}$ therefore differs:

$$\text{if} \quad p(k) = x' \quad \Rightarrow \quad D_{P_{\text{part-1}}} = \prod_{p(z)\in S} \left(e(\widetilde{K}_{i,j,p(z)}, P'_k) \quad e(\widetilde{K}'_{i,j,p(z)}, P''_k) \right)^{\frac{1}{p(k)-p(z)}}$$

$$\text{if} \quad p(k) = x \quad \Rightarrow \quad D_{P_{\text{part-1}}} = e(K_{i,j,p(k)}, P'_k) \quad e(K'_{i,j,p(k)}, P''_k)$$

We must consider two cases when calculating $D_{P_{\text{part-1}}}$. We start by looking at the first case, which always occurs when a negated attribute $P(k) = x'$ is present.

$$
\begin{aligned}
D_{P_{\text{part-1}}} &= \prod_{p(z)\in S} \left(e(\widetilde{K}_{i,j,p(z)}, P'_k) \quad e(\widetilde{K}'_{i,j,p(z)}, P''_k) \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g^{b\delta'_{i,j,p(z)}}, (H^{p(k)}h)^{-\varepsilon_k}) \quad e((G'^{p(z)}h^b)^{\delta'_{i,j,p(z)}}, g^{\varepsilon_k}) \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g^{b\delta'_{i,j,p(z)}}, H^{-p(k)\varepsilon_k}) \quad e(g^{b\delta'_{i,j,x}}, h^{-\varepsilon_k}) \quad e((H^{bp(z)}h^b)^{\delta'_{i,j,p(z)}}, g^{\varepsilon_k}) \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g,H)^{-p(k)\varepsilon_k b\delta'_{i,j,p(z)}} \quad e(g,h)^{-\varepsilon_k b\delta'_{i,j,x}} \quad e(H^{bp(z)\delta'_{i,j,p(z)}}, g^{\varepsilon_k}) \quad e(h^{b\delta'_{i,j,p(z)}}, g^{\varepsilon_k}) \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g,H)^{-p(k)\varepsilon_k b\delta'_{i,j,p(z)}} \quad e(g,h)^{-\varepsilon_k b\delta'_{i,j,x}} \quad e(H,g)^{p(z)\varepsilon_k b\delta'_{i,j,p(z)}} \quad e(h,g)^{\varepsilon_k b\delta'_{i,j,p(z)}} \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g,H)^{-p(k)\varepsilon_k b\delta'_{i,j,p(z)}} \quad \cancel{e(g,h)^{-\varepsilon_k b\delta'_{i,j,x}}} \quad e(H,g)^{p(z)\varepsilon_k b\delta'_{i,j,p(z)}} \quad \cancel{e(h,g)^{\varepsilon_k b\delta'_{i,j,p(z)}}} \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g,H)^{(-p(k)+p(z))\varepsilon_k b\delta'_{i,j,p(z)}} \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g,H)^{-\varepsilon_k b\delta'_{i,j,p(z)}(p(k)-p(z))} \right)^{\frac{1}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} \left(e(g,H)^{-\varepsilon_k b\delta'_{i,j,p(z)}} \right)^{\frac{p(k)-p(z)}{p(k)-p(z)}} \\
&= \prod_{p(z)\in S} e(g,H)^{-\varepsilon_k b\delta'_{i,j,p(z)}} \\
&= e(g,H)^{-\varepsilon_k b(\sum_{z\in x} \delta'_{i,j,p(z)})} \qquad | \quad \text{with equation 1} \\
&= e(g,H)^{-\varepsilon_k b\sigma_{i,j}}
\end{aligned}
$$

After the calculation of $D_{P_{part-1}}$ we can now continue with the calculation of D_P.

$$
\begin{aligned}
D_P &= \prod_{p(k)\in S}\Big(e(K'_{i,j},P_k) \quad DP_{\text{part-1}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}G'^{\varepsilon_k}) \quad e(g,H)^{-\varepsilon_k b\sigma_{i,j}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}) \quad e(g^{\sigma_{i,j}},H^{\varepsilon_k b}) \quad e(g,H)^{-\varepsilon_k b\sigma_{i,j}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}) \quad e(g,H)^{\varepsilon_k b\sigma_{i,j}} \quad e(g,H)^{-\varepsilon_k b\sigma_{i,j}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}) \quad \cancel{e(g,H)^{\varepsilon_k b\sigma_{i,j}}} \quad \cancel{e(g,H)^{-\varepsilon_k b\sigma_{i,j}}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u})\Big)^{\omega_k}\\
&= e(g^{\sigma_{i,j}},f)^{\sum_{p(k)\in S}\omega_k(A_k\cdot u)} \quad | \quad \text{with equation 2}\\
&= e(g^{\sigma_{i,j}},f)^{\pi}
\end{aligned}
$$

If we have a normal, non-negated attribute $(p(k)=x)$, the calculation by $P_k = f^{A_k\cdot u}(G)^{\varepsilon_k}$ of $D_{P_{\text{part-1}}}$ looks like this:

$$
\begin{aligned}
D_{P\text{part-1}} &= e\Big(K_{i,j,p(k)},P'_k\Big) \quad e\Big(K'_{i,j,p(k)},P''_k\Big)\\
&= e\Big(g^{\delta_{i,j,p(k)}},(H^{p(k)}h)^{-\varepsilon_k}\Big) \quad e\Big((H^{p(k)}h)^{\delta_{i,j,p(k)}}G^{-\sigma_{i,j}},g^{\varepsilon_k}\Big)\\
&= e\Big(g^{\delta_{i,j,p(k)}},H^{-p(k)\varepsilon_k}\Big) \quad e\Big(g^{\delta_{i,j,p(k)}},h^{-\varepsilon_k}\Big) \quad e\Big((H^{p(k)}h)^{\delta_{i,j,p(k)}},g^{\varepsilon_k}\Big) \quad e\Big(G^{-\sigma_{i,j}},g^{\varepsilon_k}\Big)\\
&= e\Big(g^{\delta_{i,j,p(k)}},H^{-p(k)\varepsilon_k}\Big) \quad e\Big(g^{\delta_{i,j,p(k)}},h^{-\varepsilon_k}\Big) \quad e\Big(H^{p(k)\delta_{i,j,p(k)}},g^{\varepsilon_k}\Big) \quad e\Big(h^{\delta_{i,j,p(k)}},g^{\varepsilon_k}\Big) \quad e\Big(G^{-\sigma_{i,j}},g^{\varepsilon_k}\Big)\\
&= e\Big(g,H\Big)^{-p(k)\varepsilon_k\delta_{i,j,p(k)}} \quad e\Big(g,h\Big)^{-\varepsilon_k\delta_{i,j,p(k)}} \quad e\Big(H,g\Big)^{p(k)\varepsilon_k\delta_{i,j,p(k)}} \quad e\Big(h,g\Big)^{\varepsilon_k\delta_{i,j,p(k)}} \quad e\Big(G,g\Big)^{\varepsilon_k-\sigma_{i,j}}\\
&= \cancel{e\Big(g,H\Big)^{-p(k)\varepsilon_k\delta_{i,j,p(k)}}} \quad \cancel{e\Big(g,h\Big)^{-\varepsilon_k\delta_{i,j,p(k)}}} \quad \cancel{e\Big(H,g\Big)^{p(k)\varepsilon_k\delta_{i,j,p(k)}}} \quad \cancel{e\Big(h,g\Big)^{\varepsilon_k\delta_{i,j,p(k)}}} \quad e\Big(G,g\Big)^{\varepsilon_k-\sigma_{i,j}}\\
&= e(G,g)^{\varepsilon_k-\sigma_{i,j}}
\end{aligned}
$$

This now leads to the following calculation of D_P:

$$
\begin{aligned}
D_P &= \prod_{p(k)\in S}\Big(e(K'_{i,j},P_k) \quad DP_{\text{part-1}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}G^{\varepsilon_k}) \quad e(G,g)^{\varepsilon_k-\sigma_{i,j}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}) \quad e(g^{\sigma_{i,j}},G^{\varepsilon_k}) \quad e(G,g)^{\varepsilon_k-\sigma_{i,j}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}) \quad e(g,G)^{\varepsilon_k\sigma_{i,j}} \quad e(G,g)^{\varepsilon_k-\sigma_{i,j}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u}) \quad \cancel{e(g,G)^{\varepsilon_k\sigma_{i,j}}} \quad \cancel{e(G,g)^{\varepsilon_k-\sigma_{i,j}}}\Big)^{\omega_k}\\
&= \prod_{p(k)\in S}\Big(e(g^{\sigma_{i,j}},f^{A_k\cdot u})\Big)^{\omega_k}\\
&= e(g^{\sigma_{i,j}},f)^{\sum_{p(k)\in S}\omega_k(A_k\cdot u)} \quad | \quad \text{with equation 2}\\
&= e(g^{\sigma_{i,j}},f)^{\pi}
\end{aligned}
$$

By this case distinction can be distinguished between negated and non-negated attributes, where the calculation of D_p will result in $e(g^{\sigma_{i,j}},f)^{pi}$. Because if this is not the case, the reconstruction of the plaintext will always fail. Compare section 3.5.1.

3.5 Calculating D_I

We divide the calculation of D_I into two parts $D_{I part-1}$ and $D_{I part-2}$ to get a better overview.

$$D_I = D_{I_{\text{part-1}}} \cdot D_{I_{\text{part-2}}}$$

The two parts can be calculated as follows:

$$
\begin{aligned}
D_{I_{\text{part-1}}} &= \frac{e(\overline{K}_{i,j},Q_i)\cdot e(K''_{i,j},Q''_i)}{e(K'_{i,j},Q'_i)}\\
D_{I_{\text{part-2}}} &= \frac{e_3(R'_i,C'_j)}{e_3(R_i,C_j)}
\end{aligned}
$$

We start with the consideration of part 1, where we first calculate $\overline{K}_{i,j}$ as follows:

$$
\begin{aligned}
\overline{K}_{i,j} &= K_{i,j}\cdot\Big(\prod_{j'\in\overline{R}'_i\setminus\{j\}}\overline{K}_{i,j,j'}\Big)\\
&= g^{\alpha_i}g^{r_ic_j}(ff_j)^{\sigma_{i,j}}\cdot\Big(\prod_{j'\in\overline{R}'_i\setminus\{j\}}f_{j'}^{\sigma_{i,j}}\Big)\\
&= g^{\alpha_i}g^{r_ic_j}\cdot\Big(f\prod_{j'\in\overline{R}'_i}f_{j'}\Big)^{\sigma_{i,j}}
\end{aligned}
$$

With the calculated $\overline{K}_{i,j}$ we can calculate $D_{I part-1}$:

$$
\begin{aligned}
D_{I_{\text{part-1}}} &= \frac{e\Big(\overline{K}_{i,j}, Q_i\Big) \cdot e\Big(K''_{i,j}, Q''_i\Big)}{e\Big(K'_{i,j}, Q'_i\Big)} \\
&= \frac{e\Big(g^{\alpha_i} g^{r_i c_j} \cdot (f \prod_{j' \in \overline{R}'_i} f_{j'})^{\sigma_{i,j}}, g^{\tau s_i (v_i \cdot v_c)}\Big) \cdot e\Big(Z_i^{\sigma_{i,j}}, g^{t_i}\Big)}{e\Big(g^{\sigma_{i,j}}, (f \prod_{j' \in \overline{R}'_i} f_{j'})^{\tau s_i (v_i \cdot v_c)} Z_i^{t_i} f^{\pi}\Big)} \\
&= \frac{e\Big(g^{\alpha_i} g^{r_i c_j}, g^{\tau s_i (v_i \cdot v_c)}\Big) \cdot e\Big((f \prod_{j' \in \overline{R}'_i} f_{j'})^{\sigma_{i,j}}, g^{\tau s_i (v_i \cdot v_c)}\Big) \cdot e\Big(Z_i^{\sigma_{i,j}}, g^{t_i}\Big)}{e\Big(g^{\sigma_{i,j}}, (f \prod_{j' \in \overline{R}'_i} f_{j'})^{\tau s_i (v_i \cdot v_c)}\Big) \cdot e\Big(g^{\sigma_{i,j}}, Z_i^{t_i} f^{\pi}\Big)} \\
&= \frac{e\Big(g^{\alpha_i} g^{r_i c_j}, g^{\tau s_i (v_i \cdot v_c)}\Big) \cdot \cancel{e\Big((f \prod_{j' \in \overline{R}'_i} f_{j'})^{\sigma_{i,j}}, g^{\tau s_i (v_i \cdot v_c)}\Big)} \cdot e\Big(Z_i^{\sigma_{i,j}}, g^{t_i}\Big)}{\cancel{e\Big(g^{\sigma_{i,j}}, (f \prod_{j' \in \overline{R}'_i} f_{j'})^{\tau s_i (v_i \cdot v_c)}\Big)} \cdot e\Big(g^{\sigma_{i,j}}, Z_i^{t_i}\Big) \cdot e\Big(g^{\sigma_{i,j}}, f^{\pi}\Big)} \\
&= \frac{e\Big(g^{\alpha_i} g^{r_i c_j}, g^{\tau s_i (v_i \cdot v_c)}\Big) \cdot \cancel{e\Big(Z_i^{\sigma_{i,j}}, g^{t_i}\Big)}}{\cancel{e\Big(g^{\sigma_{i,j}}, Z_i^{t_i}\Big)} \cdot e\Big(g^{\sigma_{i,j}}, f^{\pi}\Big)} \\
&= \frac{e\Big(g^{\alpha_i} g^{r_i c_j}, g^{\tau s_i (v_i \cdot v_c)}\Big)}{e\Big(g^{\sigma_{i,j}}, f^{\pi}\Big)}
\end{aligned}
$$

Now we can turn to the calculation of $part-2$. This calculation depends on two indexes, the user-index $(\bar{i}, \bar{j})$, which is used for encryption and the user-index (i, j), hidden in the private key. Since the decryption process only works if $\big((i = \bar{i}) \wedge (j \geq \bar{j})\big)$ or $(i > \bar{i})$ applies. This fact leads to a case distinction in which six cases have to be considered which are now being dealt with.

1. $i < \bar{i} \quad \wedge \quad j < \bar{j}$

$$
\begin{aligned}
D_{I_{\text{part-2}}} &= \frac{e_3(R'_i, C'_j)}{e_3(R_i, C_j)} \\
&= \frac{e_3(g^{\kappa v_i}, g^{\omega_j})}{e_3(g^{v_i}, g^{c_j \tau (v_c + \mu_j \chi_3)} \cdot g^{\kappa \omega_j})} \\
&= \frac{e_3(g, g)^{\kappa v_i \omega_j}}{e_3(g, g)^{v_i c_j \tau (v_c + \mu_j \chi_3) + v_i \kappa \omega_j}} \\
&= \frac{1}{e_3(g, g)^{v_i c_j \tau (v_c + \mu_j \chi_3) + v_i \kappa \omega_j - v_i \kappa \omega_j}} \\
&= \frac{1}{e_3(g, g)^{v_i c_j \tau (v_c + \mu_j \chi_3)}} \\
&= \frac{1}{e(g, g)^{c_j \tau (v_i v_c + v_i \chi_3 \mu_j)}}
\end{aligned}
$$

Since we randomly select $v_i \in \mathbb{Z}_p$ and $v_i \cdot \chi_3 \neq 0$, the calculation of $D_{I_{\text{part-2}}}$ leads to an additional exponent that destroys M. M cannot be reconstructed. Please compare section 3.5.1.

2. $i < \bar{i} \quad \wedge \quad j \geq \bar{j}$

$$\begin{aligned} D_{I_{\text{part-2}}} &= \frac{e_3(R'_i, C'_j)}{e_3(R_i, C_j)} \\ &= \frac{e_3(g^{\kappa v_i}, g^{\omega_j})}{e_3(g^{v_i}, g^{c_j \tau v_c} \cdot g^{\kappa \omega_j})} \\ &= \frac{e_3(g,g)^{\kappa v_i \omega_j}}{e_3(g,g)^{v_i c_j \tau v_c + v_i \kappa \omega_j}} \\ &= \frac{1}{e_3(g,g)^{v_i c_j \tau v_c + v_i \kappa \omega_j - v_i \kappa \omega_j}} \\ &= \frac{1}{e(g,g)^{c_j \tau v_i v_c}} \end{aligned}$$

Also here the reconstruction of M is made impossible, because one exponent is missing, so that the exponents cannot cancel each other out. Please compare section 3.5.1.

3. $i = \bar{i} \quad \wedge \quad j < \bar{j}$

$$\begin{aligned} D_{I_{\text{part-2}}} &= \frac{e_3(R'_i, C'_j)}{e_3(R_i, C_j)} \\ &= \frac{e_3(g^{r_i \kappa s_i v_i}, g^{\omega_j})}{e_3(g^{r_i s_i v_i}, g^{c_j \tau (v_c + \mu_j \chi_3)} \cdot g^{\kappa \omega_j})} \\ &= \frac{e_3(g,g)^{r_i \kappa s_i v_i \omega_j}}{e_3(g,g)^{r_i s_i v_i c_j \tau (v_c + \mu_j \chi_3) + r_i s_i v_i \kappa \omega_j}} \\ &= \frac{1}{e_3(g,g)^{r_i s_i v_i c_j \tau (v_c + \mu_j \chi_3) + r_i s_i v_i \kappa \omega_j - r_i s_i v_i \kappa \omega_j}} \\ &= \frac{1}{e_3(g,g)^{r_i s_i v_i c_j \tau (v_c + \mu_j \chi_3)}} \\ &= \frac{1}{e(g,g)^{r_i s_i c_j \tau (v_i v_c + v_i \chi_3 \mu_j)}} \end{aligned}$$

An unwanted additional exponent prevents the calculation here as well (see section 3.5.1) . M cannot be reconstructed.

4. $i = \bar{i} \quad \wedge \quad j \geq \bar{j}$

$$
\begin{aligned}
D_{I_{\text{part-2}}} &= \frac{e_3(R'_i, C'_j)}{e_3(R_i, C_j)} \\
&= \frac{e_3(g^{r_i \kappa s_i v_i}, g^{\omega_j})}{e_3(g^{r_i s_i v_i}, g^{c_j \tau v_c} \cdot g^{\kappa \omega_j})} \\
&= \frac{e_3(g,g)^{r_i \kappa s_i v_i \omega_j}}{e_3(g,g)^{r_i s_i v_i c_j \tau v_c + r_i s_i v_i \kappa \omega_j}} \\
&= \frac{1}{e_3(g,g)^{r_i s_i v_i c_j \tau v_c + r_i s_i v_i \kappa \omega_j - r_i s_i v_i \kappa \omega_j}} \\
&= \frac{1}{e(g,g)^{r_i s_i c_j \tau v_i v_c}}
\end{aligned}
$$

The first case in which the result can be easily shortened with other terms when reconstructing M. The reconstruction of M succeeds (see section 3.5.1).

5. $i > \bar{i} \quad \wedge \quad j < \bar{j}$

$$
\begin{aligned}
D_{I_{\text{part-2}}} &= \frac{e_3(R'_i, C'_j)}{e_3(R_i, C_j)} \\
&= \frac{e_3(g^{r_i \kappa s_i v_i}, g^{\omega_j})}{e_3(g^{r_i s_i v_i}, g^{c_j \tau (v_c + \mu_j \chi_3)} \cdot g^{\kappa \omega_j})} \\
&= \frac{e_3(g,g)^{r_i \kappa s_i v_i \omega_j}}{e_3(g,g)^{r_i s_i v_i c_j \tau (v_c + \mu_j \chi_3) + r_i s_i v_i \kappa \omega_j}} \\
&= \frac{1}{e_3(g,g)^{r_i s_i v_i c_j \tau (v_c + \mu_j \chi_3) + r_i s_i v_i \kappa \omega_j - r_i s_i v_i \kappa \omega_j}} \\
&= \frac{1}{e_3(g,g)^{r_i s_i v_i c_j \tau (v_c + \mu_j \chi_3)}} \\
&= \frac{1}{e_3(g,g)^{r_i s_i c_j \tau (v_i v_c + v_i \chi_3 \mu_j)}} \\
&= \frac{1}{e(g,g)^{r_i s_i c_j \tau v_i v_c}}
\end{aligned}
$$

Again, the result is shaped in such a way that it can be cancelled when reconstructing M. The reconstruction of M succeeds (see section 3.5.1).

6. $i > \overline{i} \quad \wedge \quad j \geq \overline{j}$

$$\begin{aligned} D_{I_{\text{part-2}}} &= \frac{e_3(R'_i, C'_j)}{e_3(R_i, C_j)} \\ &= \frac{e_3(g^{r_i \kappa s_i v_i}, g^{\omega_j})}{e_3(g^{r_i s_i v_i}, g^{c_j \tau v_c} \cdot g^{\kappa \omega_j})} \\ &= \frac{e_3(g,g)^{r_i \kappa s_i v_i \omega_j}}{e_3(g,g)^{r_i s_i v_i c_j \tau v_c + r_i s_i v_i \kappa \omega_j}} \\ &= \frac{1}{e_3(g,g)^{r_i s_i v_i c_j \tau v_c + r_i s_i v_i \kappa \omega_j - r_i s_i v_i \kappa \omega_j}} \\ &= \frac{1}{e(g,g)^{r_i s_i c_j \tau v_i v_c}} \end{aligned}$$

As with cases 5 and 6, the result during the reconstruction of M can easily be erased against changing terms. The reconstruction of M succeeds (see section 3.5.1).

Now we can merge $part - 1$ and $part - 2$ and determine D_I:

$$\begin{aligned} D_I &= D_{I\text{part-1}} \cdot D_{I\text{part-2}} \\ &= \frac{e\left(g^{\alpha_i} g^{r_i c_j}, g^{\tau s_i (v_i \cdot v_c)}\right)}{e\left(g^{\sigma_{i,j}}, f^{\pi}\right)} \cdot \frac{1}{e\left(g,g\right)^{r_i s_i c_j \tau v_i v_c}} \\ &= \frac{e\left(g^{\alpha_i}, g^{\tau s_i (v_i \cdot v_c)}\right) e\left(g^{r_i c_j}, g^{\tau s_i (v_i \cdot v_c)}\right)}{e\left(g^{\sigma_{i,j}}, f^{\pi}\right)} \cdot \frac{1}{e\left(g,g\right)^{r_i s_i c_j \tau v_i v_c}} \\ &= \frac{e\left(g^{\alpha_i}, g^{\tau s_i (v_i \cdot v_c)}\right) e\left(g,g\right)^{r_i s_i c_j \tau (v_i \cdot v_c)}}{e\left(g^{\sigma_{i,j}}, f^{\pi}\right)} \cdot \frac{1}{e\left(g,g\right)^{r_i s_i c_j \tau v_i v_c}} \\ &= \frac{e\left(g^{\alpha_i}, g^{\tau s_i (v_i \cdot v_c)}\right) \cancel{e\left(g,g\right)^{r_i s_i c_j \tau (v_i \cdot v_c)}}}{e\left(g^{\sigma_{i,j}}, f^{\pi}\right)} \cdot \cancel{\frac{1}{e\left(g,g\right)^{r_i s_i c_j \tau v_i v_c}}} \\ &= \frac{e\left(g^{\alpha_i}, g^{\tau s_i (v_i \cdot v_c)}\right)}{e\left(g^{\sigma_{i,j}}, f^{\pi}\right)} \end{aligned}$$

We just calculated D_P and got $e\left(g^{\sigma_{i,j}}, f^{\pi}\right)$. If we look at the result of D_I we see that the same term occurs in the denominator of D_I and cancel each other out (see section 3.5.1).

3.5.1 Reconstructing M

We can reconstruct M like this:

$$
\begin{aligned}
M' &= \frac{T_i}{D_P \cdot D_I} \\
&= \frac{M \cdot E_i^{\tau s_i (v_i \cdot v_c)}}{e(g^{\sigma_{i,j}}, f)^{\pi} \cdot \frac{e\left(g^{\alpha_i}, g^{\tau s_i (v_i \cdot v_c)}\right)}{e\left(g^{\sigma_{i,j}}, f^{\pi}\right)}} \\
&= \frac{M \cdot e(g,g)^{\alpha_i \tau s_i (v_i \cdot v_c)}}{\cancel{e(g^{\sigma_{i,j}}, f)^{\pi}} \cdot \frac{e\left(g^{\alpha_i}, g^{\tau s_i (v_i \cdot v_c)}\right)}{\cancel{e\left(g^{\sigma_{i,j}}, f^{\pi}\right)}}} \\
&= \frac{M \cdot e(g,g)^{\alpha_i \tau s_i (v_i \cdot v_c)}}{e(g,g)^{\alpha_i \tau s_i (v_i \cdot v_c)}} \\
&= \frac{M \cdot \cancel{e(g,g)^{\alpha_i \tau s_i (v_i \cdot v_c)}}}{\cancel{e(g,g)^{\alpha_i \tau s_i (v_i \cdot v_c)}}} \\
&= M
\end{aligned}
$$

M can only be reconstructed if the attributes in the key can solve the access structure that D_P takes care of. Non-monotonic attributes can be used as shown. Finally, D_I can only be valid if the user-index (i, j) of the private key is equal to or greater than the embedded user-index $(\bar{i}, \bar{j})$ used for encryption.

4 Blackbox Tracing

The black box calculation is taken unchanged from [2]. It is based on D_I and the exclusion of user indexes. The decryption box D and the PP public parameters are needed to decrypt the potentially non-monotonic access structures under the probability parameter ϵ. The result is a set of user indexes marked as traitors.

5 Evaluation

The ABE scheme is compared with selected other ABE schemas regarding its functionalities and component sizes. The result is shown in the Figures 2 and 3. Furthermore, the schema was implemented in Java, so that its functionality was not only shown mathematically.

Table 2: Feature Comparison

Scheme	Traceability	Revocation	Large Attribute Universe	Non-Monotonic
[2013] Liu et al. [7]	blackbox	×	×	×
[2014] Yamada et al. [3]	×	×	✓	✓
[2014] Ning et al. [8]	whitebox	×	✓	×
[2014] Deng et al. [9]	blackbox	×	✓	×
[2016] Liu et al. [10]	blackbox	×	×	×
[2016] Li et al. [11]	blackbox	direct	×	×
[2016] Liu and Wong [1]	blackbox	direct	✓	×
[2017] Li et al. [12]	×	×	×	✓
this work	blackbox	direct	✓	✓

Table 3: Component Size Comparison

Scheme	Master Secret Key	Public Key	Cipher-Text	Private Key
Liu et al. [7]	$1+\sqrt{N}$	$3+4\sqrt{N}+\lvert U\rvert$	$9\sqrt{N}+2l$	$4+\lvert S\rvert$
Yamada et al. [3]	2	7	$2+3l$	$2+4\lvert S\rvert$
Ning et al. [8]	4	7	$3+3l$	$4+2\lvert S\rvert$
Deng et al. [9]	1	3	$4+2l$	$2+(\lvert S\rvert\cdot\lvert L\rvert)$
Liu et al. [10]	$6+6\sqrt{N}+4\lvert U\rvert$	$10+6\sqrt{N}+4\lvert U\rvert$	$8\sqrt{N}+l$	$4+\lvert S\rvert$
Li et al. [11]	$6+6\sqrt{N}+4\lvert U\rvert$	$10+11\sqrt{N}+4\lvert U\rvert$	$8\sqrt{N}+l$	$4+\sqrt{N}+\lvert S\rvert$
Liu and Wong [1]	$3\sqrt{N}$	$5+5\sqrt{N}$	$8\sqrt{N}+3l$	$3+\sqrt{N}+2\lvert S\rvert$
Li et al. [12]	$1+2\lvert U\rvert$	$2+\lvert U\rvert$	$2+\lvert R\rvert$	2
this work	$1+3\sqrt{N}$	$6+5\sqrt{N}$	$8\sqrt{N}+3l$	$3+\sqrt{N}+4\lvert S\rvert$

N= total number of users, $|L|$= Scheme specific value, denoting the length of a codeword
$|U|$= The number of all possible attributes in the attribute universe, l= Number of rows in LSSS matrix
$|S|$= Number of Attributes asigned to the privat key, $|R|$ Valid path size of the Ordered Binary Decision Diagram

6 Acknowledgement

A thank-you goes to Alwin Alwin, who, building on the work of Thatmann [5], wrote down the calculations and programmed a first Java prototype by extending "Traceable and Revocable Attribute-based Encryption" (jTR-ABE) [13] in his master thesis [6].

References

[1] Z. Liu and D. S. Wong. "Practical Attribute-Based Encryption: Traitor Tracing, Revocation and Large Universe". In: *The Computer Journal* 59.7 (2016), pp. 983–1004. ISSN: 0010-4620. DOI: 10.1093/comjnl/bxv101.

[2] Z. Liu and D. S. Wong. "Practical Attribute Based Encryption: Traitor Tracing, Revocation, and Large Universe". In: *IACR Cryptology ePrint Archive* (2014). Available at https://eprint.iacr.org/2014/616.

[3] S. Yamada, N. Attrapadung, G. Hanaoka, and N. Kunihiro. “A Framework and Compact Constructions for Non-monotonic Attribute-Based Encryption”. In: *Public-Key Cryptography – PKC 2014: 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings*. Ed. by H. Krawczyk. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 275–292. ISBN: 978-3-642-54631-0. DOI: `10.1007/978-3-642-54631-0_16`. URL: `http://dx.doi.org/10.1007/978-3-642-54631-0_16`.

[4] R. Ostrovsky, A. Sahai, and B. Waters. “Attribute-based Encryption with Non-monotonic Access Structures”. In: *Proceedings of the 14th ACM Conference on Computer and Communications Security*. CCS ’07. Alexandria, Virginia, USA: ACM, 2007, pp. 195–203. ISBN: 978-1-59593-703-2. DOI: `10.1145/1315245.1315270`. URL: `http://doi.acm.org/10.1145/1315245.1315270`.

[5] D. Thatmann. *Non-monotonic Practical Attribute Based Encryption*. 2016. URL: `https://entrance.snet.tu-berlin.de/wp-content/uploads/sites/6/2018/09/NonMonotonic-PABE.pdf`.

[6] A. Alwin. “On Tracability of Attribute-Based Encryption”. MA thesis. Technische Universität Berlin, 2019.

[7] Z. Liu, Z. Cao, and D. S. Wong. “Blackbox Traceable CP-ABE: How to Catch People Leaking Their Keys by Selling Decryption Devices on Ebay”. In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. CCS ’13. Berlin, Germany: ACM, 2013, pp. 475–486. ISBN: 978-1-4503-2477-9. DOI: `10.1145/2508859.2516683`. URL: `http://doi.acm.org/10.1145/2508859.2516683`.

[8] J. Ning, Z. Cao, X. Dong, L. Wei, and X. Lin. “Large Universe Ciphertext-Policy Attribute-Based Encryption with White-Box Traceability”. English. In: *Computer Security - ESORICS 2014*. Vol. 8713. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 55–72. ISBN: 978-3-319-11211-4. DOI: `10.1007/978-3-319-11212-1_4`. URL: `http://dx.doi.org/10.1007/978-3-319-11212-1_4`.

[9] H. Deng, Q. Wu, B. Qin, J. Mao, X. Liu, L. Zhang, and W. Shi. “Who Is Touching My Cloud”. In: *Computer Security - ESORICS 2014*. Ed. by M. Kutyłowski and J. Vaidya. Cham: Springer International Publishing, 2014, pp. 362–379. ISBN: 978-3-319-11203-9.

[10] Z. Liu and D. S. Wong. “Traceable CP-ABE on Prime Order Groups: Fully Secure and Fully Collusion-Resistant Blackbox Traceable”. In: *Information and Communications Security*. Ed. by S. Qing, E. Okamoto, K. Kim, and D. Liu. Cham: Springer International Publishing, 2016, pp. 109–124. ISBN: 978-3-319-29814-6. DOI: `10.1007/978-3-319-29814-6_10`. URL: `https://doi.org/10.1007/978-3-319-29814-6_10`.

[11] X. Li, K. Liang, Z. Liu, and D. S. Wong. *Attribute Based Encryption: Traitor Tracing, Revocation and Fully Security on Prime Order Groups*. Cryptology ePrint Archive, Report 2016/1140. 2016. URL: `https://eprint.iacr.org/2016/1140`.

[12] L. Li, T. Gu, L. Chang, Z. Xu, Y. Liu, and J. Qian. “A Ciphertext-Policy Attribute-Based Encryption Based on an Ordered Binary Decision Diagram”. In: *IEEE Access* 5 (2017), pp. 1137–1145. ISSN: 2169-3536. DOI: `10.1109/ACCESS.2017.2651904`.

[13] *Traceable and Revocable Attribute-based Encryption in Java (jTR-ABE)*. URL: `https://github.com/TU-Berlin-SNET/jTR-ABE`.