Improved Online/Offline Attribute Based Encryption and More

Jindan Zhang[1,2,3(✉)], Baocang Wang[4], and Xu An Wang[2,5]

[1] Xianyang Vocational Technical College, Xianyang, China
69957106@qq.com
[2] Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, Guilin, China
wangxazjd@163.com
[3] State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, China
[4] School of Telecommunications Engineering, Xidian University, Xi'an, China
bcwang79@aliyun.com
[5] Key Laboratory of Cryptology and Information Security, Engineering University of CAPF, Xi'an, China

Abstract. Attribute based encryption is a very useful primitive for scalable access control on the ciphertexts and has found broad applications, such as secure cloud storage etc. When this primitive is used by mobile phones, the computation cost is too heavy. So Hohenberger and Waters introduced the concept of Online/offline attribute based encryption and give a such concrete construction. In this paper, we give an improved construction based on their proposal. Compared with their proposal, our proposal needs 5 pairings instead of $2|I| + 1$ pairings, which is much more efficient than the original scheme. Furthermore, we generalize this technique to speed up the computation of multi-modular exponentiation, and thus also get an interesting result.

1 Introduction

Attribute based encryption is a cryptographic primitive for flexible controlling on decryption ability for the ciphertexts, such as secure cloud storage [5–8]. However, most of the attribute based encryption use the bilinear pairing, which is a heavy computation task. Thus if we use mobile phone to implement this primitive, the mobile phone will run out the energy in a very short time. Thus we need some advanced techniques to help the mobile phones to implement the task of attribute based encryption and decryption. Usually there are two ways for doing this: the first one is outsourcing the decryption of attribute based encryption to the cloud, which has received great attention these years, and many wonderful results have been achieved, such as [1–4,9]; the second one is to implement the encryption in the online/offline way, that is, when the mobile phone is in charging, it can implement the offline part of the encryption, which is a heavy

L. Barolli et al. (eds.), *Advances in Internetworking, Data & Web Technologies*,
Lecture Notes on Data Engineering and Communications Technologies 6,
DOI 10.1007/978-3-319-59463-7_60

task, when the mobile phone is working without charging, it can implement the online part of encryption, which is a more easy part for encryption. Which can be implement in several seconds. These two ideas are very useful for widely application of attribute based encryption for our life.

In this paper, we concentrate on the second technique, online/offline attribute based encryption. We find that the HK proposal can even be improved again, we can reduce the number of bilinear pairings from linear with the attributes to constant ones, thus can further enlarge the time the mobile phone can live when doing such encryption.

We first review of HW's online/offline scheme and then we propose an improved one based on it. Then we generalize our technique to speed up the computation of multi-modular exponentiation, taking the vector commitments as an example, which is also an interesting work. Finally we give the conclusion.

2 Review of HW's Online/Offline ABE Scheme

Here we first review the concept and scheme of online/offline ABE. In PKC'14 [10], Hohenberger and Waters proposed an online/offline ABE scheme based on the unbounded KP-ABE scheme of Rouselakis and Waters [11].

1. $\mathsf{Setup}(\lambda, U)$. The setup algorithm takes as input a security parameter and a universe U of attributes. To cover the most general case, we let $U = \{0,1\}^*$. It then chooses a bilinear group $\mathbb{G}$ of prime order p, generators $g, h, u, w \in \mathbb{G}$. In addition, it chooses random exponents $\alpha \in Z_p$. The authority sets $MSK = (\alpha, PK)$ as the master secret key. It publishes the public parameters as:

$$PK = (\mathbb{G}, p, g, h, u, w, e(g,g)^{\alpha})$$

   We assume that the universe of attributes can be encoded as elements in Z_p.
2. $\mathsf{Extract}(MSK, (M, \rho))$. The extract algorithm takes as input the master secret key MSK and an LSSS access structure (M, ρ). Let M be an $l \times n$ matrix. The function ρ associates rows of M to attributes. The algorithm initially chooses random values $y_2, \cdots, y_n \in Z_p$. It then computes l shares of the master secret key as $(\lambda_1, \lambda_2, \cdots, \lambda_l) := M \cdot (\alpha, y_2, \cdots, y_n)^T$ (where T denotes the transpose). It then picks l random exponents $t_1, t_2, \cdots, t_l \in Z_p$. For $i = 1$ to l, it computes

$$K_{i,0} := g^{\lambda_i} w^{t_i}, K_{i,1} := (u^{\rho(i)} h)^{-t_i}, K_{i,2} = g^{t_i}$$

   and the private key is $SK = ((M, \rho), \{K_{i,0}, K_{i,1}, K_{i,2}\}_{i \in [1,l]})$.
3. $\mathsf{Offline.Encrypt}(PK))$. The offline encryption algorithm takes in the public parameters only. Here we describe the basic system which assumes a maximum bound of P attributes will be associated with any ciphertext. The algorithm first picks a random $s \in Z_p$ and computes

$$key = e(g,g)^{\alpha s}, C_0 = g^s$$

Next for $j = 1$ to P, it chooses random $r_j, x_j \in Z_p$ and computes

$$C_{j,1} = g^{r_j}, C_{j,2} = (u^{x_j} h)^{r_j} w^{-s}$$

One can view this as encrypting for a random attribute x_j, where this will be corrected in the online phase. We remark that the work done in the offline phase is roughly equivalent to the work of the regular encryption algorithm in [11].

The intermediate ciphertext is $IT = (Key, C_0, \{r_j, x_j, C_{j,1}, C_{j,2}\}_{j\in[1,P]})$.

4. Online.Encrypt(PK)). The online encryption KEM algorithm takes as input the public parameters, an intermediate ciphertext IT, and a set of attributes $S = (A_1, A_2, \cdots, A_{k\leq P})$. For $j = 1$ to k, it computes $C_{j,3} := (r_j(A_j - x_j)) \bmod p$. Intuitively, this will correct to the proper attributes. It sets the ciphertext as:

$$CT = (S, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j\in[1,k]})$$

The encapsulated key is key. The dominant cost is one multiplication in Z_p per attribute in S.

5. Decrypt(SK, CT). The decryption algorithm in the KEM setting recovers the encapsulated key. It takes as input a ciphertext $CT = (S, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j\in[1,k]})$ for attribute set S and a private key $SK = ((M, \rho), \{K_{i,0}, K_{i,1}, K_{i,2}\}_{i\in[1,l]}$ for access structure (M, ρ). If S does not satisfy this access structure, then the algorithm issues an error message. Otherwise, it sets $I := \{i : \rho(i) \in S\}$ and computes constants $\omega_i \in Z_p$ such that $\Sigma_{i\in I}\omega_i \cdot M_i = (1, 0, \cdots, 0)$, where M_i is the i-th row of the matrix M. Then it then recovers the encapsulated key by calculating $key :=$

$$\Pi_{i\in I}(e(C_0, K_{i,0})e(C_{j,1}, K_{i,1})e(C_{j,2} \cdot u^{C_{j,3}}, K_{i,2}))^{\omega_i} = e(g, g)^{\alpha s}$$

where j is the index of the attribute $\rho(i)$ in S(it depends on i). This does not increase the number of pairing operations over [[11], Appendix C], although it adds $|I|$ exponentiations.

Correctness. If the attribute set S of the ciphertext is authorized, we have that $\Sigma_{i\in I}\omega_i\lambda_i = \alpha$. Therefore, $Key =:$

$$\begin{aligned}
&\Pi_{i\in I}(e(C_0, K_{i,0})e(C_{j,1}, K_{i,1})e(C_{j,2} \cdot u^{C_{j,3}}, K_{i,2}))^{\omega_i} \\
&= \Pi_{i\in I}(e(g^s, g^{\lambda_i}\omega^{t_i})e(g^{r_j}, (u^{\rho(i)}h)^{-t_i})e((u^{x_j}h)^{r_j}\omega^{-s} \cdot u^{r_j(\rho(i)-x_j)}, g^{t_i}))^{\omega_i} \\
&= \Pi_{i\in I}(e(g^s, g^{\lambda_i}\omega^{t_i})e(g^{r_j}, (u^{\rho(i)}h)^{-t_i})e((u^{x_j}h)^{r_j}\omega^{-s} \cdot u^{r_j(\rho(i)-x_j)}, g^{t_i}))^{\omega_i} \\
&= \Pi_{i\in I}(e(g, g)e(g, \omega)^{st_i}e(g, u)^{-r_jt_i\rho(i)}e(g, h)^{-r_jt_i}e(g, u)^{\rho(i)r_jt_i}e(g, h)^{r_jt_i}e(g, \omega)^{-st_i})^{\omega_i} \\
&= \Pi_{i\in I}e(g, g)^{s\omega_i\lambda_i} = e(g, g)^{s\alpha}
\end{aligned}$$

2.1 Our Improved Online/Offline ABE Scheme

1. Setup(λ, U). The setup algorithm takes as input a security parameter and a universe U of attributes. To cover the most general case, we let $U = \{0, 1\}^*$.

It then chooses a bilinear group $\mathbb{G}$ of prime order p, generators $g, h, u, w \in \mathbb{G}$. In addition, it chooses random exponents $\alpha \in Z_p$. The authority sets $MSK = (\alpha, PK)$ as the master secret key. It publishes the public parameters as:

$$PK = (\mathbb{G}, p, g, h, u, w, e(g, g)^{\alpha})$$

We assume that the universe of attributes can be encoded as elements in Z_p.

2. Extract$(MSK, (M, \rho))$. The extract algorithm takes as input the master secret key MSK and an LSSS access structure (M, ρ). Let M be an $l \times n$ matrix. The function ρ associates rows of M to attributes. The algorithm initially chooses random values $y_2, \cdots, y_n \in Z_p$. It then computes l shares of the master secret key as $(\lambda_1, \lambda_2, \cdots, \lambda_l) := M \cdot (\alpha, y_2, \cdots, y_n)^T$ (where T denotes the transpose). It then picks l random exponents $t_1, t_2, \cdots, t_l \in Z_p$ and also a random exponent $T_1 \in Z_p$. It first computes $K_0 = g^{T_1}$, and for $i = 1$ to l, it computes

$$K_{i,0} := g^{\lambda_i} w^{t_i}, K_{i,1} := (u^{\rho(i)} h)^{-t_i}, K_{i,2} = (t_i - T_1) \bmod p$$

and the private key is $SK = ((M, \rho), K_0, \{K_{i,0}, K_{i,1}, K_{i,2}\}_{i \in [1,l]})$.

3. Offline.Encrypt(PK)). The offline encryption algorithm takes in the public parameters only. Here we describe the basic system which assumes a maximum bound of P attributes will be associated with any ciphertext. The algorithm first picks a random $s \in Z_p$ and computes

$$key = e(g, g)^{\alpha s}, C_0 = g^s$$

Next it first selects a random T_2 and computes $C_1 = g^{T_2}$, and for $j = 1$ to P, it chooses random $r_j, x_j \in Z_p$ and computes

$$C_{j,1} = (r_j - T_2) \bmod p, C_{j,2} = (u^{x_j} h)^{r_j} w^{-s}$$

One can view this as encrypting for a random attribute x_j, where this will be corrected in the online phase. We remark that the work done in the offline phase is roughly equivalent to the work of the regular encryption algorithm in [11].

The intermediate ciphertext is $IT = (Key, C_0, C_1, \{r_j, x_j, C_{j,1}, C_{j,2}\}_{j \in [1,P]})$.

4. Online.Encrypt(PK)). The online encryption KEM algorithm takes as input the public parameters, an intermediate ciphertext IT, and a set of attributes $S = (A_1, A_2, \cdots, A_{k \leq P})$. For $j = 1$ to k, it computes $C_{j,3} := (r_j(A_j - x_j)) \bmod p$. Intuitively, this will correct to the proper attributes. It sets the ciphertext as:

$$CT = (S, C_0, C_1, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,k]})$$

The encapsulated key is key. The dominant cost is one multiplication in Z_p per attribute in S.

5. Decrypt(SK, CT). The decryption algorithm in the KEM setting recovers the encapsulated key. It takes as input a ciphertext $CT = (S, C_0, C_1, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,k]})$ for attribute set S and a private key

$SK = ((M,\rho), K_0, \{K_{i,0}, K_{i,1}, K_{i,2}\}_{i\in[1,l]}$ for access structure (M,ρ). If S does not satisfy this access structure, then the algorithm issues an error message. Otherwise, it sets $I := \{i : \rho(i) \in S\}$ and computes constants $\omega_i \in Z_p$ such that $\Sigma_{i\in I}\omega_i \cdot M_i = (1, 0, \cdots, 0)$, where M_i is the i-th row of the matrix M. Then it then recovers the encapsulated key by calculating $key :=$

$$\Pi_{i\in I}(e(C_0, K_{i,0})e(C_1 \cdot g^{C_{j,1}}, K_{i,1})e(C_{j,2} \cdot u^{C_{j,3}}, K_0 \cdot g^{K_{i,2}}))^{\omega_i} = e(g,g)^{\alpha s}$$

where j is the index of the attribute $\rho(i)$ in S(it depends on i).

Correctness. If the attribute set S of the ciphertext is authorized, we have that $\Sigma_{i\in I}\omega_i\lambda_i = \alpha$. Therefore, $Key =$:

$$\begin{aligned}
&\Pi_{i\in I}(e(C_0, K_{i,0})e(C_1 \cdot g^{C_{j,1}}, K_{i,1})e(C_{j,2} \cdot u^{C_{j,3}}, K_0 \cdot g^{K_{i,2}}))^{\omega_i} \\
&= \Pi_{i\in I}(e(C_0, K_{i,0})e(g^{r_j}, K_{i,1})e(C_{j,2} \cdot u^{C_{j,3}}, g^{t_i}))^{\omega_i} \\
&= \Pi_{i\in I}(e(g^s, g^{\lambda_i}\omega^{t_i})e(g^{r_j}, (u^{\rho(i)}h)^{-t_i})e((u^{x_j}h)^{r_j}\omega^{-s} \cdot u^{r_j(\rho(i)-x_j)}, g^{t_i}))^{\omega_i} \\
&= \Pi_{i\in I}(e(g^s, g^{\lambda_i}\omega^{t_i})e(g^{r_j}, (u^{\rho(i)}h)^{-t_i})e((u^{x_j}h)^{r_j}\omega^{-s} \cdot u^{r_j(\rho(i)-x_j)}, g^{t_i}))^{\omega_i} \\
&= \Pi_{i\in I}(e(g,g)e(g,\omega)^{st_i}e(g,u)^{-r_jt_i\rho(i)}e(g,h)^{-r_jt_i}e(g,u)^{\rho(i)r_jt_i}e(g,h)^{r_jt_i}e(g,\omega)^{-st_i})^{\omega_i} \\
&= \Pi_{i\in I}e(g,g)^{s\omega_i\lambda_i} = e(g,g)^{s\alpha}
\end{aligned}$$

But note here

$$\begin{aligned}
&\Pi_{i\in I}(e(C_0, K_{i,0})e(C_1 \cdot g^{C_{j,1}}, K_{i,1})e(C_{j,2} \cdot u^{C_{j,3}}, K_0 \cdot g^{K_{i,2}}))^{\omega_i} \\
&= e(C_0, \Pi_{i\in I}K_{i,0}^{\omega_i})e(C_1, \Pi_{i\in I}K_{i,1}^{\omega_i})e(g, \Pi_{i\in I}K_{i,1}^{\omega_i C_{j,1}})e(\Pi_{i\in I}(C_{j,2} \cdot u^{C_{j,3}})^{\omega_i}, K_0) \\
&e(\Pi_{i\in I}(C_{j,2} \cdot u^{C_{j,3}})^{\omega_i K_{i,2}}, g)
\end{aligned}$$

which needs 5 pairings instead of $2|I| + 1$ pairings for the original scheme, and the original scheme needs $2|I|$ modular exponentiation while this scheme needs $5|I| + 2$ modular exponentiation, which is still more efficient than the original scheme.

3 Generalization

Here we generalize the above technique to the setting for modular exponentiation, which is a very usual operation in cryptographic primitives. We illustrate the new technique for speeding up multi-modular exponentiation via an improvement to [12]. First we review the CF vector commitment scheme.

3.1 CF's Vector Commitments

1. VC.KeyGen$(1^k, q)$. Let $\mathbb{G}, \mathbb{G}_T$ be two bilinear groups of prime order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $g \in \mathbb{G}$ be a random generator. Randomly choose $z_1, \cdots, z_q \leftarrow Z_p$. For all $i = 1, \cdots, q$ set $h_i = g^{z_i}$. For all $i, j = 1, \cdots, q$, $i \neq j$ set $h_{i,j} = g^{z_i z_j}$. Set $pp = (g, \{h_i\}_{i\in[q]}, \{h_{i,j}\}_{i,j\in[q],i\neq j})$. The message space is $\mathcal{M} = Z_p$.

2. $\mathsf{VC.Com}_{pp}(m_1, \cdots, m_q)$. Compute $C = h_1^{m_1} h_2^{m_2} \cdots h_q^{m_q}$ and output C and the auxiliary information $aux = (m_1, \cdots, m_q)$
3. $\mathsf{VC.Open}_{pp}(m_i, i, aux)$. Compute

$$\Lambda_i = \prod_{j=1, j\neq i}^{q} h_{i,j}^{m_j} = (\prod_{j=1, j\neq i}^{q} h_j^{m_j})^{z_i}$$

4. $\mathsf{VC.Ver}_{pp}(C, m_i, i, \Lambda_i)$. If the following equations hold,

$$e(C/h_i^{m_i}, h_i) = e(\Lambda_i, g)$$

then outputs 1, otherwise output 0.
5. $\mathsf{VC.Update}_{pp}(C, m, m', i)$. Compute the updated commitment $C' = C \cdot h_i^{m_i - m}$. Finally output C' and $U = (m, m', i)$.
6. $\mathsf{VC.ProofUpdate}_{pp}(C, \Lambda_j, m', U)$. A client who owns a proof Λ_j, that is valid w.r.t. to C for some message at position j, can use the update information $U = (m, m', i)$ to compute the updated commitment C' and produce a new proof Λ'_j which will be valid w.r.t C'. We distinguish two cases:
   a. $i \neq j$. Compute the updated commitment $C' = C \cdot h_i^{m'-m}$ while the updated proof is $\Lambda'_j = \Lambda_j (h_i^{m'-m})^{z_j} = \Lambda_j h_{j,i}^{m'-m}$
   b. $i = j$. Compute the updated commitment as $C' = C \cdot h_i^{m'-m}$ while the updated proof remains the same as Λ_i.

3.2 Our Improved Algorithm

1. $\mathsf{VC.KeyGen}(1^k, q)$. Let $\mathbb{G}, \mathbb{G}_T$ be two bilinear groups of prime order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $g \in \mathbb{G}$ be a random generator. Randomly choose $z_1, \cdots, z_q \leftarrow Z_p$. For all $i = 1, \cdots, q$ set $h_i = g^{z_i}$. Furthermore, choose a random $t \in Z_p$, computes $T = g^t, r_1 = z_1 - t \bmod p, r_2 = z_2 - t \bmod p, \cdots, r_q = z_q - t \bmod p$. Note here $h_i = g^t \cdot g^{r_i} = g^{t + z_i - t} = g^{z_i}$. For all $i, j = 1, \cdots, q$, $i \neq j$ set $h_{i,j} = g^{z_i z_j}$. Set $pp = (g, T, r_1, r_2, \cdots, r_q, \{h_{i,j}\}_{i,j \in [q], i \neq j})$. The message space is $\mathcal{M} = Z_p$.
2. $\mathsf{VC.Com}_{pp}(m_1, \cdots, m_q)$. Compute

$$\begin{aligned} C &= h_1^{m_1} h_2^{m_2} \cdots h_q^{m_q} = (Tg^{r_1})^{m_1} \cdots Tg^{r_q})^{m_q}) \\ &= T^{(m_1 + m_2 + \cdots + m_q)} g^{r_1 m_1 + r_2 m_2 + \cdots + r_q m_q} \end{aligned}$$

and output C and the auxiliary information $aux = (m_1, \cdots, m_q)$. Note here the committer needs only compute two modular exponentiations instead of q modular exponentiations.
3. $\mathsf{VC.Open}_{pp}(m_i, i, aux)$. Compute

$$\Lambda_i = \prod_{j=1, j\neq i}^{q} h_{i,j}^{m_j} = (\prod_{j=1, j\neq i}^{q} h_j^{m_j})^{z_i}$$

4. $\mathsf{VC.Ver}_{pp}(C, m_i, i, \Lambda_i)$. If the following equations hold,

$$e(C/h_i^{m_i}, h_i) = e(\Lambda_i, g)$$

then outputs 1, otherwise output 0.

5. $\mathsf{VC.Update}_{pp}(C, m, m', i)$. Compute the updated commitment $C' = C \cdot h_i^{m_i - m}$. Finally output C' and $U = (m, m', i)$.
6. $\mathsf{VC.ProofUpdate}_{pp}(C, \Lambda_j, m', U)$. A client who owns a proof Λ_j, that is valid w.r.t. to C for some message at position j, can use the update information $U = (m, m', i)$ to compute the updated commitment C' and produce a new proof Λ'_j which will be valid w.r.t C'. We distinguish two cases:
   a. $i \neq j$. Compute the updated commitment $C' = C \cdot h_i^{m'-m}$ while the updated proof is $\Lambda'_j = \Lambda_j (h_i^{m'-m})^{z_j} = \Lambda_j h_{j,i}^{m'-m}$
   b. $i = j$. Compute the updated commitment as $C' = C \cdot h_i^{m'-m}$ while the updated proof remains the same as Λ_i.

4 Conclusion

In this paper, we consider the issue of implementing of online/offline of ABE for mobile devices with energy efficiency. We give an improvement to the HW's proposal. And we also generalize our technique to the setting of multi modular-exponentiation. However, we also note our results are very basic, there are many work need to do in the future, such as proving the security of the proposals in the formal model, and extending this technique to other settings.

Acknowledgements. This work was supported by the National Natural Science Foundation of China (No. 61572390), the 111 Project (No. B08038), and Guangxi Key Laboratory of Cryptography and Information Security (Grant No. GCIS201610).

References

1. Green, M., Hohenberger, S., Waters, B.: Outsourcing the decryption of ABE ciphertexts. In: Proceedings of the USENIX Security Symposim, San Francisco, CA, USA (2013)
2. Lai, J., Deng, R., Guan, C., Weng, J.: Attribute-based encryption with verifiable outsourced decryption. IEEE Trans. Inf. Forensics Secur. **8**(8), 1343–1354 (2013)
3. Li, J., Huang, X., Li, J., Chen, X., Xiang, Y.: Securely outsourcing attribute-based encryption with checkability. IEEE Trans. Parallel Distrib. Syst. (2013, in Press). doi:10.1109/TPDS.2013.27
4. Qin, B., Deng, R.H., Liu, S., Ma, S.: Attribute-based encryption with efficient verifiable outsourced decryption. IEEE Trans. Inf. Forensics Secur. **10**(7), 1384–1393 (2015)
5. Puzar, M., Plagemann, T.: Data sharing in mobile ad-hoc networks-a study of replication and performance in the MIDAS data space. Int. J. Space-Based Situated Comput. **1**(2/3), 137–150 (2015)

6. Petrlic, R., Sekula, S., Sorge, C.: A privacy-friendly architecture for future cloud computing. Int. J. Grid Util. Comput. **4**(4), 265–277 (2013)
7. Wang, Y., Du, J., Cheng, X., Liu, Z., Lin, K.: Degradation and encryption for outsourced PNG images in cloud storage. In. Int. J. Grid Util. Comput. **7**(1), 22–28 (2016)
8. Ye, X., Khoussainov, B.: Fine-grained access control for cloud computing. Int. J. Grid Util. Comput. **4**(2/3), 160–168 (2013)
9. Wang, X.A., Ma, J., Xhafa, F.: Outsourcing decryption of attribute based encryption with energy efficiency. In: Proceeding of the International Conference on P2P, Parallel, Grid, Cloud and Internet Computing 3PGCIC 2015, pp. 444-448. IEEE (2015)
10. Hohenberger, S., Waters, B.: Online/offline attribute-based encryption. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 293–310. Springer, March 2014
11. Rouselakis, Y., Waters, B.: Practical constructions and new proof methods for large universe attribute-based encryption. In: Sadeghi, A.-R., Gligor, V.D., Yung, M. (eds.) ACM CCS 13, pp. 463–474. ACM Press, November 2013
12. Catalano, D., Fiore, D.: Vector commitments and their applications. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 55–72. Springer, February/March 2013