Article

Implementation of an Attribute-Based Encryption Scheme Based on SM9

Yang Shi ¹, Zhiyuan Ma ¹, Rufu Qin ², Xiaoping Wang ³, Wujing Wei ¹ and Hongfei Fan ^{1,*} ¹ School of Software Engineering, Tongji University, Shanghai 201804, China² State Key Laboratory of Marine Geology, Tongji University, Shanghai 200092, China³ College of Electronics and Information Engineering, Tongji University, Shanghai 201804, China

* Correspondence: fanhongfei@tongji.edu.cn

Received: 14 June 2019; Accepted: 22 July 2019; Published: 30 July 2019

Abstract: In recent years, attribute-based encryption (ABE) has been widely applied in mobile computing, cloud computing, and the Internet of things, for supporting flexible and fine-grained access control of sensitive data. In this paper, we present a novel attribute-based encryption scheme that is based on bilinear pairing over Barreto and Naehrig curves (BN-curves). The identity-based encryption scheme SM9, which is a Chinese commercial cryptographic standard and a forthcoming part of ISO/IEC11770-3, has been used as the fundamental building block, and thus we first introduce SM9 and present our SM9 implementation in details. Subsequently, we propose the design and implementation of the ABE scheme. Moreover, we also develop a hybrid ABE for achieving lower ciphertext expansion rate when the size of access structure or plaintext is large. The performance and energy consumption of the implementation of the proposed ABE and its hybrid version are evaluated with a workstation, a PC, a smart phone, and an embedded device. The experimental results indicated that our schemes work well on various computing platforms. Moreover, the proposed schemes and their implementations would benefit developers in building applications that fulfill the regulatory compliance with the Chinese commercial cryptographic standard since there is no existing ABE scheme compatible with any Chinese cryptographic standard.

Keywords: attribute-based encryption; identity-based encryption; SM9; BN-curves; bilinear pairing

1. Introduction

There is an increasing requirement for data sharing and processing in the distributed computing environment with fine-grained access control with the development of cloud computing, mobile computing, and the Internet of things. Public key encryption is a powerful approach towards protecting the confidentiality of sensitive data. However, there are two limitations as follows. Firstly, a public key cryptosystem relies on the public key infrastructure (PKI), which requires huge overhead on certificate management and verification. Secondly, the resource provider needs to encrypt data with the public key of each user in the receiving group and separately send the ciphertext to the corresponding user, which results in large processing overhead and bandwidth consumption.

Shamir presented the first identity-based cryptosystem [1] in 1984 to mitigate the first issue. Lately, Boneh and Franklin [2] proposed the most well-known identity-based encryption (IBE) scheme that was based on bilinear pairing, leading to a new development on identity-based cryptography. The main feature of identity-based cryptosystems is that they do not need certificates for public keys. The identity string of a user, such as the email address or phone number, can be used as the user's public key.

To mitigate the second issue, Sahai and Waters [3] proposed a new category of encryption scheme, called attribute-based encryption (ABE), where the provider of the data can decide the access policy

of the ciphertext, which implies that only the users who satisfy the specified attributes can decrypt the ciphertext. The resource providers only need to encrypt messages according to the decryptors' attributes, without taking care of their identities and the number of qualified decryptors. This feature significantly reduces the overhead of data encryption and enables resource providers to formulate flexible and scalable access control policies to manage the sharing range of data. The first ABE scheme [3] only supports threshold access control strategies. The community further proposed key-policy ABE [4] and ciphertext-policy ABE to support more flexible access control strategies [5]. In recent years, a number of novel ABE schemes have been proposed with distinct security features [6–9]. Moreover, specialized ABE schemes for various application scenarios such as cloud computing [10–12], social networks [13], Internet of things (IoT) [14–16], blockchains [17,18], and mobile computing [19,20], have been proposed.

In this paper, we propose an ABE scheme utilizing SM9 IBE as a building block. SM9 is a cryptography standard that defines a set of identity-based cryptographic schemes, including signature, encryption, and key agreement. It originates from a Chinese cryptographic standard [21], and then its signature scheme has been adopted by the International Organization for Standardization as ISO/IEC 14888-3:2018 [22]. Currently, its encryption scheme and key agreement scheme have been formally reviewed as proposals for ISO/IEC 18033-5 [23] and ISO/IEC11770-3 [24], respectively.

There is no ABE scheme that is compatible with any Chinese cryptographic standard, as well as the forthcoming ISO standard. Therefore, the proposed ABE scheme is fully compatible with the Chinese standard and the forthcoming international standard to fulfill the regulatory compliance. The SM9 IBE is efficient and bandwidth-saving; for example, it performs better than the ISO/IEC 18033-5 [23] in terms of both computational efficiency and ciphertext size [25]. Our ABE scheme inherits such features of SM9 IBE. The scheme is implemented in Java, and experimental results on PCs, smart phones, and embedded devices indicated that our scheme performs well on typical platforms.

As a building block, the SM9 IBE scheme, which is based on bilinear pairings over the prime order elliptic curves that were proposed in [26] (i.e., BN-curves), has been implemented at first. It is worth noting that our implementation is not trivial, because this is the first Java implementation of SM9 to the best of our knowledge, and we have implemented the fundamental mathematical structures from scratch, since there is no appropriate Java library that can be utilized. We have not found any Java library that implements the R-Ate pairing [27] and the extension field specified by the SM9 standard simultaneously. For example, the most widely-used Java library for pairing-based cryptography, i.e., JPBC [28], supports neither the R-Ate pairing nor the demanded extension field. Therefore, we have implemented the R-Ate pairing over a BN-curve and the specified extension finite field, and then integrated them with the interfaces that were provided by the JPBC library to support SM9 IBE and the proposed ABE schemes.

Our contributions are briefly summarized, as follows: (i) we have proposed an ABE scheme based on the SM9 IBE scheme; (ii) we have implemented the ABE scheme as well as the SM9 IBE; and, (iii) we have presented a hybrid ABE scheme as an optimization. The proposed schemes and implementations would benefit developers in building applications that fulfill the regulatory compliance with the Chinese commercial cryptographic standard and the forthcoming ISO standard. Additionally, the ABE schemes have advantages, such as optimized ciphertext expansion rate and anonymity of receivers. The experimental results regarding performance and energy consumption indicate that the proposed scheme works well on various platforms, such as PCs, smart phones, and embedded devices.

The remainder of this paper is organized, as follows. Section 2 briefly introduces the preliminaries and the overview of the software architecture. Section 3 introduces the implementation of SM9 IBE scheme and the experimental results on performance and energy consumption. Section 4 presents an ABE scheme based on the SM9 IBE scheme, as well as the experimental results on performance and energy consumption. Section 5 presents an optimization of the ABE scheme, i.e., a hybrid ABE scheme. Section 6 concludes the paper.

2. Prerequisites and Overview

BN-curves [26] are pairing-friendly curves with prime order and the embedding degree k of 12, which present great efficiency and security in the pairing process. The equation of a BN-curve is

$$E : y^2 = x^3 + b, b \neq 0 \quad (1)$$

The trace (of Frobenius) of the curve [29], the curve order, and the characteristic of F_q are parameterized as

$$\begin{aligned} t &= 6x^2 + 1 \\ n &= 36x^4 - 36x^3 + 18x^2 - 6x + 1 \\ p &= 36x^4 - 36x^3 + 24x^2 - 6x + 1 \end{aligned} \quad (2)$$

R-ate pairing [27] is a generalization of the Ate pairing. This pairing enables the loop length of Miller's algorithm [30] to be shorter than that of the Ate pairing. This makes the computation of pairing more efficient.

Let π_q be the Frobenius endomorphism, and t be the trace (of Frobenius) of the curve. For input Q, P , the R-ate pairing algorithm on the BN-curves is shown in Algorithm 1.

Algorithm 1 $R_a(Q, P)$

```

1   $s \leftarrow 6t + 2, s \leftarrow \sum_{i=0}^{L-1} s_i 2^i, s_i \in \{-1, 0, 1\}$ 
2   $T \leftarrow Q, f \leftarrow 1$ 
3  For  $i = L - 2$  to 0 Do
4       $f \leftarrow f^2 \cdot l_{T,T}(P); T \leftarrow 2T$ 
5      IF  $s_i = -1$ 
6           $f \leftarrow f \cdot l_{T,-Q}(P); T \leftarrow T - Q$ 
7      ELSE IF  $s_i = 1$ 
8           $f \leftarrow f \cdot l_{T,Q}(P); T \leftarrow T + Q$ 
9  EndFor
10  $Q_1 \leftarrow \pi_q(Q); Q_2 \leftarrow \pi_{q^2}(Q)$ 
11  $f \leftarrow f \cdot l_{T,Q_1}(P); T \leftarrow T + Q_1$ 
12  $f \leftarrow f \cdot l_{T,-Q_2}(P); T \leftarrow T - Q_2$ 
13  $f \leftarrow f^{(p^{12}-1)/r}$ 
14 output  $f$ 
```

We have implemented the proposed ABE scheme and the SM9 IBE scheme in Java. The program can be easily deployed on a variety of platforms because of the strong portability of Java. The structure of the software is shown in Figure 1.

The package `api`, package `util`, package `pairing` and package `field` are the packages provided by the Java Pairing-based Cryptography Library (JPBC). The package `api` provides interfaces that are related to pairing operations, such as the finite field, elliptic curves, and the pairing functions. The package `util` provides support for mathematical operations and so on. Package `pairing` and package `field` are the specific implementations of the interfaces that are exposed in package `api`. We construct extension of the finite field and the particular elliptic curve specified by the SM9 standard in the package `pairing`. We have also implemented the R-Ate pairing in this package.

`Sm9Util` is a static class, and it contains all of the supporting functions, such as KDF and `H2RFi` functions, which are detailed in Section 3.1. The supporting functions can be utilized by all other classes.

The `KeyGeneratorCenter` class is designed following the singleton pattern, where the system parameters are stored. Moreover, it generates the `Sm9DecryptPrivateKey` that corresponds to the user ID/attributes for decryption.

Figure 1. The structure of attribute-based encryption (ABE) and identity-based encryption (IBE) Java implementation.

The Sm9Engine implements the SM9 hybrid encryption scheme. It contains three interfaces for users: `initEncrypt`, `initDecrypt`, and `processBlock`. The `initEncrypt/initDecrypt` function sets the ID/key for encryption/decryption. The `processBlock` function encrypts/decrypts the message after the initialization.

Sm9ABEEEngine implements an ABE scheme supporting AND-gate-only access structure, which will be introduced in Section 4, being based on the Sm9Engine. The AND-gate-only access structure can be uniquely transformed to a user identity. The encryption and decryption processes of SM9 ABE include: (i) transforming the AND-gate-only access structure into a user Identity; and, (ii) invoking the functions that were provided by Sm9Engine.

Sm9IBBEABEngine implements the ABE scheme that is presented in Section 4. Different from Sm9ABEngine, Sm9IBBEABEngine supports the generic access structure. A generic access structure, which is introduced in Section 4, is a generalization of the AND-gate-only access structure. The Sm9IBBEABEngine has two functions: Encrypt and Decrypt. The input of the Encrypt function includes a generic access structure A and the message that is to be encrypted. Users holding the private key corresponding to the attributes satisfying the access structure A through the Decrypt function can only successfully decrypt the output of the Encrypt function.

Sm9ABEHybridEngine implements the hybrid ABE scheme that is proposed in Section 5, which optimizes the ciphertext size of the ABE scheme that is presented in Section 4.

3. Implementation and Evaluation of SM9 Identity-Based Encryption

3.1. Supporting Functions

Here, we describe the supporting functions that are used in the schemes, including the key derivation function, which works as KDF2 in ISO/IEC 18033-2 [31], the hash-to-range function in [25], and the block cipher and the system parameters.

3.1.1. Key Derivation Function KDF (H_v , Z , $klen$)

Given a hash function H_v with output bit length v , bit string Z , and an integer $klen$ (that denotes the required bit length of the secret keys, where $klen < (2^{32}-1)v$). The output is a bit string K of length $klen$. The pseudocode for this function is given in Algorithm 2.

Algorithm 2 $KDF(H_v, Z, klen)$

```

1   $ct \leftarrow 0x00000001$ 
2  For  $i = 1$  to  $\lceil klen/v \rceil$  Do
3       $Ha_i \leftarrow H_v(Z||ct)$ 
4       $ct \leftarrow ct + 1$ 
5  EndFor
6  output the first  $klen$  bits of  $Ha_1 || Ha_2 || \dots || Ha_{\lceil klen/v \rceil}$ 
```

3.1.2. Hash to Range Function H2RF_i (H_v , Z , n)

Given a hash function H_v with output bit length v , bit string Z , integer n , and integer index i . The output is an integer h_i . Algorithm 3 gives the pseudocode. The SM9 standard requires SM3 [32] to be used as the hash function.

Algorithm 3 $H2RF_i(H_v, Z, n)$

```

1   $hlen \leftarrow \lceil 8 \times (5 \times (\log_2 n)) / 32 \rceil$ 
2   $Ha \leftarrow KDF(H_v, i||Z, hlen)$ 
3  output  $h_i \leftarrow (Ha \bmod (n-1)) + 1$ 
```

3.1.3. Block Cipher

The block cipher includes the encryption algorithm $Enc(K_1, m)$ and decryption algorithm $Dec(K_1, c)$. $Enc(K_1, m)$ encrypts plaintext m with key K_1 and its output is a ciphertext bit string c . $Dec(K_1, c)$ decrypts ciphertext c using key K_1 and its output is either a plaintext bit string m or the message "error." The bit length of the key K_1 is denoted by $K1_len$. The SM9 standard requires SM4 [33] to be used as the block cipher.

3.2. Setup and Key Extraction

This section describes the algorithm that is used to set up the system and the extraction algorithm for the private decryption key.

3.2.1. Setup (1^k)

Given input k , the output of this algorithm is the master public key M_{pk} and master secret key M_{sk} . Algorithm 4 gives its pseudocode.

Algorithm 4 $Setup(1^k)$	
<hr/>	
1	Generate three groups G_1, G_2 and G_T of prime order r
2	Generate a bilinear pairing map $e : G_1 \times G_2 \rightarrow G_T$
3	$P_1 \xleftarrow{\$} G_1$
4	$P_2 \xleftarrow{\$} G_2$
5	$ke \xleftarrow{\$} \mathbb{Z}_r^*$
6	$P_{pub-e} \leftarrow [ke]P_1$
7	$g_e \leftarrow e(P_{pub-e}, P_2)$
8	Pick a cryptographic hash function H_v
9	$hid_e \leftarrow 3$
10	$M_{pk} \leftarrow \left(\begin{array}{c} G_1, G_2, G_T, e, P_1, P_2, P_{pub-e}, g_e, \\ H2RF_1(H_v, \cdot, \cdot), H2RF_2(H_v, \cdot, \cdot), hid_e \end{array} \right)$
11	$M_{sk} \leftarrow (ke)$
12	output (M_{pk}, M_{sk})

3.2.2. Private-Key-Extract (M_{pk}, M_{sk}, ID_A)

Given an identity string $ID_A \in \{0,1\}^*$ of entity A , M_{pk} and M_{sk} , the operation outputs “error” if

$$ke + H2RF_1(H_v, ID_A || hid_e, r) \bmod r = 0 \quad (3)$$

otherwise, it outputs decryption private key

$$DE_A = \left[\frac{ke}{ke + H2RF_1(H_v, ID_A || hid_e, r)} \right] P_2 \quad (4)$$

3.3. KEM-DEM Algorithms

The SM9 encryption is a hybrid encryption scheme [25] that is built from an identity-based key encapsulation mechanism (KEM) and a data encapsulation mechanism (DEM). The encryption and decryption schemes are described, as follows.

3.3.1. KEM-DEM-Encrypt (M_{pk}, ID_A, m)

Given an identity string ID_A , plain text m (of bit length $mlen$), and master public key M_{pk} , the operation runs, as shown in Algorithm 5.

Algorithm 5 KEM – DEM – Encrypt (M_{pk}, ID_A, m)

```

1   $h_1 \leftarrow \mathbf{H2RF}_1(H_v, ID_A || hid_e, r)$ 
2   $Q \leftarrow [h_1]P_1 + P_{pub-e}$ 
3   $x \xleftarrow{\$} \mathbb{Z}_r^*$ 
4   $C_1 \leftarrow [x]Q$ 
5   $t \leftarrow g_e^x$ 
6  IF DEM is Stream cipher based on KDF
7     $klen \leftarrow mlen + v$ 
8     $K_1 || K_2 \leftarrow \mathbf{KDF}(H_v, C_1 || t || ID_A, klen)$ 
9    IF  $K_1$  is full zero
10     go to line 3
11    ELSE
12      $C_2 \leftarrow m \oplus K_1$ 
13  ELSE IF DEM is Block cipher combined with KDF
14     $klen \leftarrow K_1\_len + v$ 
15     $K_1 || K_2 \leftarrow \mathbf{KDF}(H_v, C_1 || t || ID_A, klen)$ 
16    IF  $K_1$  is full zero
17     go to line 3
18    ELSE
19      $C_2 \leftarrow \text{Enc}(K_1, m)$ 
20     $C_3 \leftarrow H_v(C_2 || K_2)$ 
21  output ( $C_1, C_2, C_3$ )

```

The encryption algorithm is a combination of KEM and DEM. It first generates and encapsulates a random key, and then uses that key to encrypt the message while using different types of DEM. Finally, a message authentication code is created to ensure the integrity and authenticity of the ciphertext. The encapsulated key, encrypted message, and authentication code make up the ciphertext. Figure 2 shows the flow chart of the encryption algorithm.

Figure 2. Encryption algorithm flow.

3.3.2. KEM-DEM-Decrypt ($M_{pk}, ID_A, DE_A, (C_1, C_2, C_3)$)

Given the master public key M_{pk} , an identity string ID_A , the corresponding private key DE_A , and cipher text (C_1, C_2, C_3) , and the operation runs, as shown in Algorithm 6.

Algorithm 6 KEM – DEM – Decrypt ($M_{pk}, ID_A, DE_A, (C_1, C_2, C_3)$)

```

1  IF  $C_1 \notin G_1^*$ 
2    output  $\perp$  and terminate
3  ELSE
4     $t \leftarrow e(C_1, DE_A)$ 
5    IF DEM is Stream cipher based on KDF
6       $klen \leftarrow C_2\_len + v$ 
7       $K_1 \parallel K_2 \leftarrow \mathbf{KDF}(H_v, C_1 \parallel t \parallel ID_A, klen)$ 
8      IF  $K_1$  is full zero
9        output  $\perp$  and terminate
10     ELSE
11        $m \leftarrow C_2 \oplus K_1$ 
12     ELSE IF DEM is Block cipher combined with KDF
13        $klen \leftarrow K_1\_len + v$ 
14        $K_1 \parallel K_2 \leftarrow \mathbf{KDF}(H_v, C_1 \parallel t \parallel ID_A, klen)$ 
15       IF  $K_1$  is full zero
16         output  $\perp$  and terminate
17       ELSE
18          $m \leftarrow \text{Dec}(K_1, C_2)$ 
19          $C_3' \leftarrow H_v(C_2 \parallel K_2)$ 
20         IF  $C_3' \neq C_3$ 
21           output  $\perp$  and terminate
22         ELSE
23           output  $m$ 

```

Upon receiving the ciphertext, the decryption algorithm decapsulates the secret key, then decrypts the message using the same type of DEM, as used by the encryption. Finally, it verifies the authentication code. Only when the integrity and authenticity of the ciphertext are confirmed is the plaintext output. Based on the algorithm, the flow chart of this process is shown in Figure 3.

Figure 3. Decryption algorithm flow.

3.4. Performance Evaluation

We measured the runtime of the KEM-DEM-Encrypt and KEM-DEM-Decrypt algorithms on four devices, which ranged from a smart phone to a workstation. Table 1 lists the configurations of the test devices. We ran the algorithm 100 times on these devices, recorded the total time, and then calculated the average value as the result. Two types of DEM were tested. Figure 4 shows the result of using a block cipher. Figure 5 shows the result of using a stream cipher. Note that the “Time” axis is logarithmic. For the detailed parameters, please refer to Table 2.

Table 1. Device configurations.

Device	Operating System	Processor
Raspberry Pi 3	Raspbian 8	ARM Cortex-A53@1.2GHz
OnePlus A6000	Android 8.1	Qualcomm Snapdragon 845@2.8GHz
MacBook Pro	macOS High Sierra 10.13.4	Intel Core i5@2.9GHz
Workstation	Ubuntu 16.04 LTS	Intel Xeon E5-2640 v3@2.6GHz * 32

Table 2. Detailed parameters in the experiment.

Parameter	Value
Curve equation	$y^2 = x^3 + 5$
Curve trace	D8000000 019062ED 0000B98B 0CB27659
Curve order	B6400000 02A3A6F1 D603AB4F F58EC744 49F2934B 18EA8BEE E56EE19C D69ECF25
Characteristic of F_q	B6400000 02A3A6F1 D603AB4F F58EC745 21F2934B 1A7AEEDB E56F9B27 E351457D
ID length	20 bytes
Private key length	256 bits
Plaintext length	1024 bytes
DEM algorithm (block cipher)	SM4 [33], a block cipher whose block size and key size are both 128 bits.
Hash function (H_v)	SM3 [32], a cryptographic hash function which outputs 128-bit digestions.

Figure 4. DEM's encryption and decryption performance (using the block cipher).

Figure 5. Data encapsulation mechanism's (DEM's) encryption and decryption performance (using the stream cipher).

We can see that the performances of using two types of DEM are very close when comparing Figures 4 and 5. This indicates that the encryption and decryption processes of the SM4 block cipher have approximately the same speed as the stream cipher. The execution time of decryption is approximately three times as much as that of the encryption. Through further comparison and analysis, we found that the pairing operation leads to such difference in the execution time.

3.5. Energy Consumption Evaluation

The energy consumption is a major concern when executing the algorithms on low-power devices. We have evaluated the energy consumption of SM9 IBE scheme on the Raspberry Pi 3 and the OnePlus A6000 smartphone. The power consumption (mAh) is shown in Figure 6, where Enc/Dec (0) represents the KEM-DEM algorithms while using stream cipher, and Enc/Dec (1) represents the algorithms using block cipher.

Figure 6. Algorithms' energy consumption on Raspberry Pi 3 and OnePlus A6000 smartphone.

Figure 6 shows that the energy consumptions of using two types of DEM are very similar. The decryption algorithms consume more energy than the encryption algorithms. OnePlus A6000 consumes more energy than Raspberry Pi 3 when executing the same algorithm.

4. Design, Implementation and Evaluation of the Attribute-Based Encryption Scheme from Identity-Based Encryption

Based on the idea in [34,35], we can transform the SM9 IBE scheme into an ABE scheme. With slight modification of the original scheme, we can construct a generic ABE scheme. Roughly speaking, we uniquely convert an access structure into an identity, and then invoke the identity-based encryption scheme with the identity.

4.1. Transformation from Access Structures to Identities

AND-gate-only Access Structure Transformation Algorithm. Here, we introduce the transformation algorithm of AND-gate-only access structure. The definition of AND-gate-only Access Structure is: The universe of attributes is denoted by u and the size of the universe is $|u|$. We can use an AND-gate-only access structure A such as $(att_1 \text{ AND } \dots \text{ AND } att_n)$, where $1 \leq n \leq |u|$. It also can be written as a set of attributes, e.g., $A = \{att_1, att_2, \dots, att_n\}$. Let $S = \{X_1, \dots, X_n\}$, where $1 \leq n \leq |u|$, be an attribute set of a user. We say that S satisfies the access structure A if and only if $att_i = X_i$, for all $1 \leq i \leq n$, denoted as $S \models A$.

The input includes an AND-gate-only access structure $A = \{X_1, \dots, X_n\}$ and the universe u of the attribute. It outputs an ID , which is uniquely corresponding to the access structure A . Algorithm 7 presents the pseudocode.

Algorithm 7 $\varphi(A, u)$

```

1  Let  $ID_A[i]$  be the  $i$ -th bit of  $ID_A$ 
2  For  $i = 1$  to  $|u|$  Do
3      IF  $X_i \in A$ 
4           $ID_A[i] \leftarrow 1$ 
5      ELSE
6           $ID_A[i] \leftarrow 0$ 
7  END For
8  output  $ID_A$ 
```

Generic Access Structure Conversion Algorithm. Here, we present the transformation algorithm of generic access structure [36]. Let $P = \{P_1, P_2, \dots, P_n\}$ be a set of attributes. A generic access structure is a collection A of non-empty subsets of $\{P_1, P_2, \dots, P_n\}$, i.e., $A \subseteq 2^{\{P_1, P_2, \dots, P_n\}} \setminus \{\emptyset\}$. The sets in A are called the authorized sets and the sets not in A are called the unauthorized sets. We can also represent the generic access structure as a disjunction of conjunctive clauses, i.e., disjunctive normal form (DNF).

The input of the conversion algorithm includes a generic access structure $A = \{A_1, \dots, A_n\} \subseteq 2^u$, as described before and the universe u of the attribute. It outputs a set of identities $S = \{ID_1, ID_2, \dots, ID_n\}$, which is uniquely corresponding to the access structure A . Algorithm 8 presents the pseudocode.

Algorithm 8 $\xi(A, u)$

```

1  Let  $S$  be a null set,  $n$  be the size of  $A$ 
2  For  $i = 1$  to  $n$  Do
3       $ID_i \leftarrow \varphi(A_i)$ 
4       $S \leftarrow S \cup \{ID_i\}$ 
5  END For
6  output  $S$ 
```

4.2. ABE Scheme Based on SM9 IBBE

We slightly modify the SM9 IBE scheme into an IBBE (Identity-based Broadcast Encryption) [34,35] scheme to generalize our scheme, so that we can transform it into an ABE scheme that supports the generic access structure. The setup and key generation algorithms of IBBE are the same as the

original SM9 IBE scheme. We propose the IBBE encrypt and decrypt algorithms based on the SM9's KEM-DEM-Encrypt and KEM-DEM-Decrypt algorithms. Taking the master public key M_{pk} , a set of identities $S = \{ID_1, ID_2, \dots, ID_n\}$, and the message M as input, the IBBEEncrypt algorithm proceeds as presented in Algorithm 9, and it outputs a set of ciphertexts $C = \{CT_1, CT_2, \dots, CT_n\}$.

Algorithm 9 IBBEEncrypt(M_{pk}, S, M)

```

1  Let  $ID_i$  be the  $i$ -th  $ID$  of  $S$ , and  $C$  be a null set.
2  For  $i = 1$  to  $|S|$  Do
3     $CT_i \leftarrow \text{SM9.KEM-DEM-Encrypt}(M_{pk}, ID_i, M)$ 
4     $C \leftarrow C \cup \{CT_i\}$ 
5  END For
6  output  $C$ 
```

Taking the master public key M_{pk} , the user identity ID , the associated private key SK_{ID} , and a set of ciphertexts $C = \{CT_1, CT_2, \dots, CT_n\}$ as input, we present the IBBEDecrypt algorithm. The pseudocode is presented in Algorithm 10.

Algorithm 10 IBBEDecrypt(M_{pk}, ID, SK_{ID}, C)

```

1  Let  $CT_i$  be the  $i$ -th ciphertext of  $C$ 
2  For  $i = 1$  to  $|C|$  Do
3     $M \leftarrow \text{SM9.KEM-DEM-Decrypt}(M_{pk}, ID, SK_{ID}, CT_i)$ 
4    IF  $M \neq \perp$ 
5      output  $M$  and terminate
6    ELSE
7      continue
8  END For
9  output  $\perp$ 
```

If the user identity $ID \in S$ (of IBBEEncrypt algorithm) and $SK_{ID} \leftarrow \text{SM9.Private-Key-Extract}(M_{pk}, M_{sk}, ID)$, then we can correctly decrypt the ciphertext with the IBBEDecrypt algorithm.

We construct the ABE scheme that supports the generic access structure based on the IBBE scheme, with the following four parts:

4.2.1. Setup (1^k)

Given a security parameter k , this algorithm calls the Setup algorithm of SM9 scheme and it sets the ABE scheme's master public key M_{pk} and the master secret key M_{sk} .

$$(M_{pk}, M_{sk}) \leftarrow \text{SM9.Setup}(1^k) \quad (5)$$

4.2.2. KeyGen (M_{pk}, M_{sk}, U)

Given the master public key M_{pk} , the master secret key M_{sk} , and a set of attributes U , this algorithm converts the set of attributes U into an identity $ID_U \in \{0, 1\}^{|u|}$ by running the algorithm φ , and then calls the Private-Key-Extract algorithm of SM9. It outputs the ABE's private key SK_U .

$$SK_U \leftarrow \text{SM9.Private-Key-Extract}(M_{pk}, M_{sk}, \varphi(U, u)) \quad (6)$$

4.2.3. Encrypt (M_{pk}, A, M)

Given the master public key M_{pk} , an access structure A , and a message M , this algorithm converts the access structure $A = \{A_1, A_2, \dots, A_n\}$ into a set of identities by running the algorithm ξ , and then gets a set of ciphertexts $C = \{CT_1, CT_2, \dots, CT_n\}$ by calling the IBBEEncrypt algorithm.

$$C \leftarrow \text{IBBEEncrypt}(M_{pk}, \xi(A, u), M) \quad (7)$$

4.2.4. Decrypt (M_{pk} , U , SK_U , C)

Given the master public key M_{pk} , a set of attributes U , the private key SK_U , and the set of ciphertexts C , this algorithm converts the set of attributes U into an identity $ID_U \in \{0, 1\}^{|u|}$ by running the algorithm φ and then gets the plaintext M by running the IBBEDecrypt algorithm.

$$M \leftarrow \text{IBBEDecrypt}(M_{pk}, \varphi(U, u), SK_U, C) \quad (8)$$

4.3. Performance Evaluation

Here, we present the experimental results on different devices. The Figures 7 and 8 show how the size of the access structure produces an impact on the performance of the ABE encryption and decryption algorithms. The former experimental result in Section 4.3 shows that the performance of using two types of DEM is very similar, and thus we only present the experimental result of the ABE scheme using block cipher. The size of the universe of attributes is 20.

Figure 7. ABE Encryption Performance.

Figure 8. ABE Decryption Performance.

The experimental result shows that the execution time and the size of access structure have a linear correlation. In summary, the proposed ABE scheme's performance on MacBook Pro and workstation is reasonably well, and its performance on Raspberry Pi and OnePlus smart phone is acceptable.

4.4. Energy Consumption Evaluation

We have also evaluated the energy consumption of ABE scheme on a Raspberry Pi 3 and a OnePlus A6000 smartphone. Figure 9 shows the power consumption (mAh).

Figure 9. ABE algorithms' energy consumption on Raspberry Pi 3 and OnePlus A6000 smartphone.

The battery capacity of a OnePlus A6000 is 3300 mAh. The ABE Decryption process, which is the most energy-consuming process, only takes approximately 0.13% of the capacity.

4.5. Security Analysis

The security of the ABE scheme also relies on the security of the transformation technique because we have transformed the SM9 IBE scheme into the ABE scheme following the technique in [34,35]. Theorem 4.1 in [35] claims that the ABE scheme derived from an IBBE scheme is secure against chosen ciphertext attacks (CCA) if the underlying IBBE scheme is CCA-secure, which is defined, as follows:

An IBBE scheme is secure against CCA if there exists no probabilistic polynomial-time adversary who can win the following security game in a non-negligible advantage.

- (i) **Setup.** The challenger C takes a unary security parameter 1^k as input, and returns the master public key M_{pk} to the adversary $\mathcal{A}$ and keeps the master secret key M_{sk} privately.
- (ii) **Learn 1.** $\mathcal{A}$ submits a series of queries $q_1, \dots, q_n$, where $q_i (i \in 1, \dots, n)$ is either a private key query or a decryption query. Note that in a private key query, $\mathcal{A}$ sends an identity to C , and then C returns the corresponding private key to $\mathcal{A}$; in a decryption query, $\mathcal{A}$ sends a ciphertext and an identity to C , and then C decrypts the ciphertext for $\mathcal{A}$.
- (iii) **Challenge.** $\mathcal{A}$ submits two equal-length messages M_0, M_1 and a challenge set of identities $\{ID_1^*, \dots, ID_n^*\}$ to C . Afterwards, C uniformly randomly selects $b' \in \{0, 1\}$, encrypts $M_{b'}$ under $\{ID_1^*, \dots, ID_n^*\}$, and finally sends the ciphertext C^* to $\mathcal{A}$.
- (iv) **Learn 2.** $\mathcal{A}$ repeats the steps in the *Learn 1* phase except for querying the identities and the ciphertext involved in the challenge.
- (v) **Guess.** $\mathcal{A}$ outputs a guess $b'' \in \{0, 1\}$ of b' and wins the game if and only if $b'' = b'$.

$\mathcal{A}$'s advantage in the above game is defined as $|\Pr[b'' = b'] - \frac{1}{2}|$.

Obviously, the proposed ABE scheme is derived from an underlying IBBE scheme that straightforwardly repeats the SM9 IBE for a limited number of times, and thus the IBBE scheme's security can be reduced to the security of SM9 IBE. As a cryptographic standard, we assume that the SM9 IBE is secure against CCA.

5. Optimization on the ABE Scheme

For the ABE scheme that is presented in Section 4, a plaintext has to be encrypted n times, where n is the size of the access structure. Therefore, it may be inefficient when the length of the plaintext or the size of the access structure is large. We further develop a hybrid encryption scheme, which uses the proposed ABE scheme for key encapsulation and the SM4 block cipher for data encapsulation to optimize the scheme.

5.1. Hybrid ABE Scheme

The hybrid scheme consists of four algorithms. The Setup algorithm and the KeyGen algorithm are the same as their counterparts in the ABE scheme. The encryption and decryption algorithms work, as follows.

Encrypt(M_{pk}, A, M): Given the master public key M_{pk} , an access structure A and a message M , this algorithm proceeds, as presented in Algorithm 11.

Algorithm 11 Encrypt(M_{pk}, A, M)

- 1 Randomly generated a SM4 secret key and an IV
 - 2 $C_1 \leftarrow \text{SM4.Encrypt}(M, \text{key}, IV)$ /* Encrypting in CBC mode. */
 - 3 $C_2 \leftarrow \text{ABE.Encrypt}(M_{pk}, A, \text{key} \parallel IV)$
 - 4 Output (C_1, C_2)
-

Decrypt(M_{pk}, U, SK_U, C): Given the master public key M_{pk} , a set of attributes U , the private key SK_U , and the ciphertext C , the decrypt algorithm proceeds as presented in Algorithm 12.

Algorithm 12 Decrypt(M_{pk}, U, SK_U, C)

- 1 (C_1, C_2) $\leftarrow C$
 - 2 $\text{key} \parallel IV \leftarrow \text{ABE.Decrypt}(M_{pk}, U, SK_U, C_2)$
 - 3 $M \leftarrow \text{SM4.Decrypt}(C_1, \text{key}, IV)$ /* Decrypting in CBC mode. */
 - 4 Output M
-

The hybrid scheme is more space-economic than the scheme in the previous section. Let m be the length of plaintext and n be the access structure's size. The size of the ciphertext of this hybrid ABE scheme is $m + n * (32 + 96)$ bytes, which is much smaller than before. For example, when the plaintext's length is 1 GB and the access structure's size is 8, the ciphertext's length is only 1 GB + 1 KB.

5.2. Performance Evaluation

Here, we present the experimental results on different devices. We set the default size of the universe of attributes as 20 and the plaintext's default length as 1 KB. The other parameters are inherited from the SM9 scheme. Figures 10 and 11 show the performance of the encryption and decryption. Figure 12 shows the relation between the ciphertext's size and the access structure's size. Figure 13 shows the relation between the ciphertext's size and the plaintext's length under the condition that the access structure's size is 10.

Figure 10. Performance of hybrid ABE encryption (1 KB plaintext).

Figure 11. Performance of hybrid ABE decryption (1 KB plaintext).

Figure 12. Hybrid ABE's ciphertext's size with respect to the access structure's size (1 KB plaintext).

Figure 13. Hybrid ABE's ciphertext's size with respect to plaintext's length (the size of access structure is 10).

5.3. Energy Consumption Evaluation

We have also evaluated the energy consumption of the hybrid ABE scheme on a Raspberry Pi 3 and a OnePlus A6000 smartphone. Figure 14 shows the power consumption (mAh).

Figure 14. Hybrid ABE algorithms' energy consumption on Raspberry Pi 3 and OnePlus A6000 smartphone (1 KB plaintext).

6. Conclusions

We present an ABE scheme that is based on the SM9 IBE scheme, which supports the generalized access structure. The proposed scheme complies with the Chinese commercial cryptographic standard and the forthcoming ISO standard. In terms of implementation, we first implement and embed the R-ate pairing on BN-curves, which is required by SM9, into the JPBC library; and, then implement SM9 as well as the proposed ABE. Moreover, we also develop a hybrid ABE for achieving lower ciphertext expansion rate when the size of access structure or plaintext is large. In general, the proposed schemes' performance is reasonably well on PCs, and is also acceptable on smart phones and embedded devices, according to our experimental results.

Author Contributions: Conceptualization, Y.S. and X.W.; methodology, Y.S.; software, Z.M., Y.S. and W.W.; validation, R.Q.; writing—original draft preparation, Z.M.; writing—review and editing, H.F.

Funding: The National Natural Science Foundation of China (grant numbers 61772371, 61702374, and 61672128), the National Key Research and Development Program of China (grant number 2018YFC0830406), the National Critical Science and Technology Infrastructure Program (China National Seafloor Scientific Observatory, Tongji University), the Shanghai Sailing Program (grant number 17YF1420500), and the Fundamental Research Funds for the Central Universities.

Acknowledgments: The authors are grateful to the editors and anonymous reviewers for their suggestions and comments for improvement of the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Shamir, A. Identity-based cryptosystems and signature schemes. In Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques, Santa Barbara, CA, USA, 19–22 August 1984; Springer: Berlin/Heidelberg, Germany, 1984.
- Boneh, D.; Franklin, M. Identity-based encryption from the Weil pairing. *Siam J. Comput.* **2003**, *32*, 586–615. [\[CrossRef\]](#)
- Sahai, A.; Waters, B. Fuzzy identity-based encryption. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005.
- 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, Alexandria, VA, USA, 30 October–3 November 2006.
- Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP'07), Oakland, CA, USA, 20–23 May 2007.
- Zhang, J.; Chen, J.; Gong, J.; Ge, A.; Ma, C. Leakage-resilient attribute based encryption in prime-order groups via predicate encodings. *Des. Codes Cryptogr.* **2018**, *86*, 1339–1366. [\[CrossRef\]](#)
- Nguyen, K.T.; Oualha, N.; Laurent, M.J.W.W. Securely outsourcing the ciphertext-policy attribute-based encryption. *World Wide Web* **2018**, *21*, 169–183. [\[CrossRef\]](#)
- Ma, H.; Zhang, R.; Yang, G.; Song, Z.; Sun, S.; Xiao, Y. Concessive Online/Offline Attribute Based Encryption with Cryptographic Reverse Firewalls—Secure and Efficient Fine-Grained Access Control on Corrupted Machines. In *European Symposium on Research in Computer Security*; Springer International Publishing: Berlin/Heidelberg, Germany, 2018.
- Lai, J.; Tang, A.Q. *Making Any Attribute-Based Encryption Accountable, Efficiently*; Springer International Publishing: Berlin/Heidelberg, Germany, 2018.
- Teng, W.; Yang, G.; Xiang, Y.; Zhang, T.; Wang, D. Attribute-Based Access Control with Constant-Size Ciphertext in Cloud Computing. *IEEE Trans. Cloud Comput.* **2017**, *5*, 617–627. [\[CrossRef\]](#)
- Li, J.; Lin, X.; Zhang, Y.; Han, J. KSF-OABE: Outsourced Attribute-Based Encryption with Keyword Search Function for Cloud Storage. *IEEE Trans. Serv. Comput.* **2017**, *10*, 715–725. [\[CrossRef\]](#)
- Chi, P.; Lei, C. Audit-Free Cloud Storage via Deniable Attribute-Based Encryption. *IEEE Trans. Cloud Comput.* **2018**, *6*, 414–427. [\[CrossRef\]](#)
- Wei, W.; Liu, S.; Li, W.; Du, D. Fractal Intelligent Privacy Protection in Online Social Network Using Attribute-Based Encryption Schemes. *IEEE Trans. Comput. Soc. Syst.* **2018**, *5*, 736–747. [\[CrossRef\]](#)
- Chaudhary, R.; Aujla, G.S.; Garg, S.; Kumar, N.; Rodrigues, J.J. SDN-Enabled Multi-Attribute-Based Secure Communication for Smart Grid in IIoT Environment. *IEEE Trans. Ind. Inform.* **2018**, *14*, 2629–2640. [\[CrossRef\]](#)
- Belguith, S.; Kaaniche, N.; Laurent, M.; Jemai, A.; Attia, R. PHOABE: Securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted IoT. *Comput. Netw.* **2018**, *133*, 141–156. [\[CrossRef\]](#)
- Ambrosin, M.; Anzanpour, A.; Conti, M.; Dargahi, T.; Moosavi, S.R.; Rahmani, A.M.; Liljeberg, P. On the Feasibility of Attribute-Based Encryption on Internet of Things Devices. *IEEE Micro* **2016**, *36*, 25–35. [\[CrossRef\]](#)
- Wu, A.; Zhang, Y.; Zheng, X.; Guo, R.; Zhao, Q.; Zheng, D. Efficient and privacy-preserving traceable attribute-based encryption in blockchain. *Ann. Telecommun.* **2019**, *74*, 1–11. [\[CrossRef\]](#)

18. Rahulamathavan, Y.; Phan, R.C.W.; Rajarajan, M.; Misra, S.; Kondo, A. Privacy-preserving blockchain based IoT ecosystem using attribute-based encryption. In Proceedings of the 2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Bhubaneswar, Odisha, India, 17–20 December 2017.
19. Miao, Y.; Ma, J.; Liu, X.; Li, X.; Liu, Z.; Li, H. Practical Attribute-Based Multi-Keyword Search Scheme in Mobile Crowdsourcing. *IEEE Internet Things J.* **2018**, *5*, 3008–3018. [CrossRef]
20. Kwon, H.; Kim, D.; Hahn, C.; Hur, J. Secure authentication using ciphertext policy attribute-based encryption in mobile multi-hop networks. *Multimed. Tools Appl.* **2017**, *76*, 19507–19521. [CrossRef]
21. GM/T0044-2016, SM9 identity-based cryptographic algorithms. 2016. Available online: <http://www.gmbz.org.cn/main/postDetail.html?id=20180322410400> (accessed on 24 July 2019).
22. ISO/IEC. *IT Security Techniques Digital Signatures with Appendix; Part 3: Discrete Logarithm Based Mechanisms*; ISO/IEC: Geneva, Switzerland, 2018.
23. ISO/IEC. *Information technology Security Techniques Encryption Algorithms; Part 5: Identity-Based Ciphers*; ISO/IEC: Geneva, Switzerland, 2015.
24. ISO/IEC. *Information technology Security Techniques—Key Management—Part 3: Mechanisms Using Asymmetric Techniques*; ISO/IEC: Geneva, Switzerland, 2015.
25. Cheng, Z. The SM9 Cryptographic Schemes. IACR Cryptology ePrint Archive. 2017. Available online: <https://eprint.iacr.org/2017/117.pdf> (accessed on 15 June 2019).
26. Barreto, P.S.L.M.; Naehrig, M. Pairing-friendly elliptic curves of prime order. In *International Workshop on Selected Areas in Cryptography*; Springer: Berlin/Heidelberg, Germany, 2005.
27. Lee, E.; Lee, H.S.; Park, C.M. Efficient and generalized pairing computation on abelian varieties. *IEEE Trans. Inf. Theory* **2009**, *55*, 1793–1803. [CrossRef]
28. The Java Pairing Based Cryptography Library (JPBC). Available online: <http://gas.dia.unisa.it/projects/jpbc/index.html#.XLCaUOsZQI> (accessed on 1 May 2019).
29. McCarthy, D.J.P.J.O.M. The trace of Frobenius of elliptic curves and the p-adic gamma function. *Pac. J. Math.* **2013**, *261*, 219–236. [CrossRef]
30. Miller, V.S. The Weil pairing, and its efficient calculation. *J. Cryptol.* **2004**, *17*, 235–261. [CrossRef]
31. ISO/IEC. *Information Technology Security Techniques-Encryption Algorithms-Part 2: Asymmetric Ciphers*; ISO/IEC: Geneva, Switzerland, 2006.
32. GM/T0004-2012, SM3 cryptographic hash algorithm. 2012. Available online: <http://www.gmbz.org.cn/upload/2018-07-24/1532401392982079739.pdf> (accessed on 24 July 2019).
33. GM/T0002-2012, SM4 block cipher algorithm. 2012. Available online: <http://www.gmbz.org.cn/upload/2018-04-04/1522788048733065051.pdf> (accessed on 24 July 2019).
34. Fan, C.I.; Tseng, Y.F.; Lin, C.W. Attribute-Based Encryption from Identity-Based Encryption. 2017 [cited 2017; 219]. Available online: <https://eprint.iacr.org/2017/219.pdf> (accessed on 10 June 2019).
35. Fan, C.I.; Tseng, Y.F.; Lin, C.W. Attribute-Based Encryption from Identity-Based Encryption. Available online: <https://pdfs.semanticscholar.org/7be4/f934560fc490d55968cc9752047b5917d669.pdf> (accessed on 10 July 2019).
36. Gentry, C. Practical identity-based encryption without random oracles. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, 28 May–1 June 2006.