Protecting Data in the Cloud: An Assessment of Practical Digital Envelopes from Attribute based Encryption

Victor J. Sosa-Sosa, Miguel Morales-Sandoval, Oscar Telles-Hurtado
and Jose Luis Gonzalez-Compean

Center of Research and Advanced Studies of the National Polytechnic Institute (CINVESTAV),
Parque Científico y Tecnológico TECNOTAM, Km. 5.5 carretera Cd. Victoria-Soto La Marina, Ciudad Victoria, Mexico

Keywords: Attribute-based Encryption, Cloud Storage, Security, Access Control, Confidentiality, Authentication.

Abstract: Cloud storage services provide users with an effective and inexpensive mechanism to store and manage big data with anytime and anywhere availability. However, data owners face the risk of losing control over their data, which could be accessed by third non-authorized parties including the provider itself. Although conventional encryption could avoid data snooping, an access control problem arises and the data owner must implement the security mechanisms to store, manage and distribute the decryption keys. This paper presents a qualitative and quantitative evaluation of two Java implementations of security schemes called DET-ABE and AES4SeC. Both are based on the digital envelope technique and attribute based encryption, a non-conventional cryptography that ensures confidentiality and access control security services. The experimental evaluation was performed in a private cloud infrastructure where experiments for both implementations ran using the same platform, settings, underlying libraries, thus providing a more fair comparison. The quantitative evaluation revealed DET-ABE and AES4SeC have similar performance when applying low security levels (128-bit keys), whereas DET-ABE surpasses AES4SeC performance when medium (192-bit keys) and high (256-bit keys) security levels are required. Qualitative evaluation shows that AES4SeC also ensures authentication and integrity services, which are not supported by DET-ABE.

1 INTRODUCTION

Cloud storage is one of the most demanded services in the cloud computing model. That service is very attractive for users because it allows the storage of a high volume of data at a relative low cost and with the guarantee of availability and reliability.

One of the main aspects delaying the acceptance of cloud storage services is security (Theoharidou et al., 2013). Data owners are often reluctant to outsource their data to distant cloud servers because non-authorized parties including the service provider can learn from their data in plaintext form.

The two main security services demanded in a cloud storage service are (Theoharidou et al., 2013; Chow et al., 2009):

- Confidentiality: Non-authorized entities can learn anything from owners' data.
- Fine grained access control: Data owners can selectively decide who can access and use their data.

Conventional cryptography (Menezes et al., 1996) can be used to implement some schemes that ensure the confidentiality of data (i.e. using symmetric cryptography) and the sharing of the encryption key to consumers by means of cryptographic digital envelopes (asymmetric encryption). However, under this solution approach the key management is responsibility of the data owner which must implement the secure mechanisms to store and share the keys. In addition, the data owner must know the identity of consumers in advance (i.e. their public key) before creating the digital envelope and sending the encrypted key to them.

Attribute-based encryption (ABE) (Sahai and Waters, 2005) is non conventional cryptography. Instead of using traditional keys, ABE encrypts using a policy over a set of attributes, which defines an access control mechanism and many-to-many encryption. In ABE, the encryptor does not need to know the identity of data consumers in advance, it encrypts data to all those whose attributes satisfy the encryption policy. Consumers are provided with a set of attributes. A key is derived from users' attributes and used for

Sosa-Sosa, V., Morales-Sandoval, M., Telles-Hurtado, O. and González-Compeán, J.
Protecting Data in the Cloud: An Assessment of Practical Digital Envelopes from Attribute based Encryption.
DOI: 10.5220/0006484603820390
In *Proceedings of the 6th International Conference on Data Science, Technology and Applications (DATA 2017)*, pages 382-390
ISBN: 978-989-758-255-4

data decryption. Only those that have the attributes that satisfy the encryption policy will be able to decrypt.

ABE is considered an important enabler technology to provide the security services demanded in cloud storage. Nevertheless, there are few available implementation of ABE schemes for practical use in end user applications (Zickau et al., 2016). ABE are complex systems relying in the theory of elliptic curves and bilinear pairings, with several aspects to cover for a secure and efficient implementation. Most of the reported works on ABE are theoretical constructions, for example (Koo et al., 2013; Saikeerthana and Umamakeswari, 2015; Fu et al., 2014). Those works do not present experimental results that prove the schemes are well suited for use in practice.

There are two kinds of ABE schemes: KP-ABE and CP-ABE. In KP-ABE, the attributes are associated to the ciphertext and the access structure is associated to the keys. In CP-ABE the situation is the opposite, the attributes are associated to decryption keys and the access structure is associated to the ciphertext. This way, CP-ABE is conceptually closer to the role based access control (RBAC) technology and preferred for providing the access control mechanism for data stored in the cloud.

In the same way, there are two common constructions of CP-ABE. The first one is based on the Bethencourt et al (Bethencourt et al., 2007) work, where the access structure is implemented as a tree (leaf nodes are attributes and internal nodes are logic gates defining the encryption policy). Some representative ABE schemes using this type of technique are (Bobba et al., 2009; Wan et al., 2012). DET-ABE (Morales-Sandoval and Diaz-Perez, 2015) is a Java application that implements this technique. The second approach is based on the constructions given by Waters (Waters, 2011), where the access structure is implemented as a matrix and the encryption policies are represented as formatted boolean formulas. Some representative constructions of ABE schemes with matrix implementations of the access structure are (Liu et al., 2015; Liu and Wong, 2016). AES4SeC (Morales-Sandoval et al., 2017) is a Java application that implements these ABE schemes. Additional to offer confidentiality and access control, AES4SeC also provides authentication and integrity services.

This work presents a quantitative and qualitative comparison of two realizations of CP-ABE for providing confidentiality and access control mechanism over the data stored in untrusted cloud servers. By itself, CP-ABE is not able to encrypt large amounts of data. For that reason, DET-ABE and AES4SeC implement the digital envelope technique, that uses a symmetric cipher to encrypt data of any size, and the decryption key is encrypted with CP-ABE, thus enforcing access control and confidentiality at the same time. Both schemes use the Advanced Encryption Standard (AES) as symmetric cipher, with support of the three security levels of 128-, 192-, and 256-bit.

An study as the one presented here allows us to highlight the advantages and disadvantages of practical implementations of the two most representative ABE constructions. Our results show how execution time varies for different security level requirements and sizes of files that are shared through a file sharing system. These results can be of interest to those users that plan to implement this kind of cryptography in real scenarios for data storage and sharing in the cloud.

The remainder of this paper is organized as follows: Section 2 presents the generalities of an ABE scheme. Section 3 describes the concept of digital envelopes and how DET-ABE and AES4SeC implements that cryptographic concept to guarantee confidentiality and access control mechanisms for large amount of data. Section 4 describes the set of experiments and the settings to evaluate DET-ABE and AES4SeC under the same conditions. This section discusses the results achieved and provides a quantitative comparison. Finally, Section 5 presents the conclusion of this work and points out the future work.

2 ATTRIBUTE BASED ENCRYPTION

Attribute based encryption is a relative new cryptography that has a main distinctive that no keys but policies are used for the data encryption process. A policy is generally a boolean formula over a set of attributes. An attribute is a property of an entity, for example *"to be a doctor"*, *"to have an academic degree"*, etc. For simplicity, attributes can be viewed as text-strings. As an example consider the policy:

$$P = [\text{“}doctor\text{”}\ \textbf{and}\ \text{“}cardiologist\text{”}]\ \textbf{or}\ [\text{“}nurse\text{”}\ \textbf{and}\ \text{“}hospital\ number\ 25\text{”}]$$

If P were used in ABE, the encrypted data (ciphertext) can be only decrypted by entities having attributes $S_1 = \{a_1, a_2, \cdots, \text{doctor}, \cdots, \text{cardiologist}, \cdots\}$ or $S_2 = \{a_1, a_2, \cdots, \text{nurse}, \cdots, \text{hospital number } 25, \cdots\}$.

In ABE, a user having the sets S_1 or S_2 will be given a decryption key completely dependent on its attributes. When decrypting, the decryption key will match the policy "mathematically". CP-ABE consists

of four main algorithms:

1. *setup* (1^n)- initializes the scheme creating a public key *PK* and a master private key *MK*. Both keys are derived from arithmetic over an elliptic curve recommended for use in cryptography. The security level is expressed in terms of n.
2. *encrypt*(*PK*, *data*, *policy*) - encrypts *data* using *PK* and the given *policy*. The result is the ciphertext *CT*.
3. *decrypt*(SK_u, *ciphertext*) - decrypts the given *ciphertext* using the decryption key SK_u of user u. The decryption key is generated using the attribute set assigned to u. The result is the data in plain form.
4. *keygen*(*S*, *MK*) - generates a decryption key using a set *S* of attributes and the master key *MK*.

In CP-ABE schemes, a trusted authority is in charge of executing the algorithms *setup* and *keygen*, and to safeguard *MK*. In simple terms, encryption and decryption occur as follows (for formalisms see (Bethencourt et al., 2007; Waters, 2011)): data D to be encrypted is mapped to a number M in a multiplicative group ($D \mapsto M$). Each attribute i in the policy is also mapped to a number C_i in the form of exponentiations in a group ($i \mapsto C_i$). The ciphertext is a set of numbers produced from a secret value s, generated internally. The SK_u key is also a set of numbers that result from mapping each user attribute i to an abstract number and also hiding them in the form of exponentiations ($i \mapsto D_i$). The only way to reveal the user attributes or the attributes in the encryption policy is by solving the discrete logarithm problem, which is believed to be hard for groups of large order (key size). During decryption, the mathematical processing of each component $\{C_i, D_i\}$ associated to attribute i allows to recover the secret s and get back D after a series of mathematical operations.

3 SECURITY SCHEMES BASED ON DIGITAL ENVELOPES

Digital envelopes are cryptographic objects constructed using public key encryption. The main purpose of a digital envelope is the secure distribution of symmetric keys (Menezes et al., 1996). Each participant u has a pair of related keys $\{K^u_{priv}, K^u_{pub}\}$, with K^u_{priv} private and K^u_{pub} public. Anyone can encrypt and send encrypted data to u using K^u_{pub} but only u can decrypt using K^u_{priv}. The digital envelope has the main distinctive that the data being encrypted is a symmetric key k, which has been used to encrypt the real content to be shared in secret with u.

3.1 DET-ABE

Authors in (Morales-Sandoval and Diaz-Perez, 2015) present DET-ABE (Digital Envelope Technique - Attribute Based Encryption), an implementation of the digital envelope that uses the Advanced Encryption Standard (AES) as the symmetric cipher and CP-ABE to selectively distribute the AES key to decryptors. DET-ABE encrypts for the security levels recommended by the National Institute of Standards and Technology (NIST): 128-bit (minimum security level), 192-bit (medium security level), and 256-bit (maximum security level).

In DET-ABE the user is responsible for specifying the level of security to be applied.
The process securing data in DET-ABE is as follows:

1. Internally, an AES-key (k) is generated from a security level λ given by the user.
2. AES encrypts *data* using k, producing the ciphertext CT_{AES}.
3. Then, CP-ABE is used to encrypt k, given a policy P over a set of valid attributes S obtaining as a result the ciphertext CT_{ABE}.
4. Finally, $\{CT_{AES}, CT_{ABE}\}$ is stored in a binary file, which can be uploaded to the cloud.

The general process of DET-ABE encryption is shown in the Figure 1.

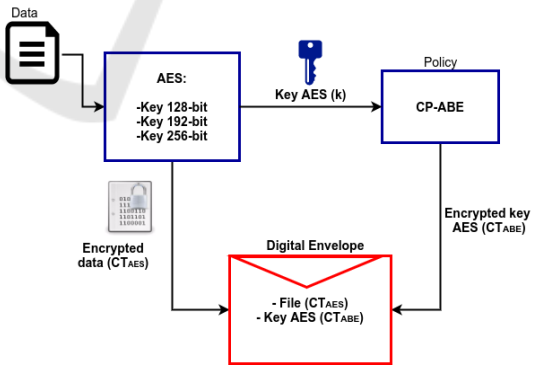

Figure 1: DET-ABE encryption process.

DET-ABE implement CP-ABE according to the construction provided by Betancourt et al. (Bethencourt et al., 2007), which proposed as access structure a tree structure where its nodes represents threshold gates (AND, OR) and the leaves describe attributes. The AND and OR gates are constructed as n-of-n and 1-of-n threshold gates, respectively. Generalizing, threshold gates are of the form m-of-n where $m \leq n$.

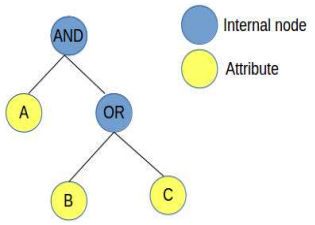

Figure 2: Tree access structure of policy $(B \vee C) \wedge A$.

In DET-ABE each policy is represented in postfix notation, for example:

- Attributes $S = \{A, B, C, D\}$
- Policy P as a boolean formula over S: $(B \vee C) \wedge A$
- P in postfix notation = $B\ C$ 1-of-2 A 2-of-2

In the previous example, the representation of P as a tree structure is shown in Figure 2.

3.2 AES4SeC

AES4SeC (Attribute based Encryption and Signing for Security in the Cloud) is a Java implementation of CP-ABE (Morales-Sandoval et al., 2017) according to the construction provided by Waters in (Waters, 2011), which uses a matrix as access structure (for an access policy that includes n attributes). This scheme was designed to provide confidentiality, access control and authentication in the process of storage, retrieval and sharing content in the cloud. AES4SeC uses the elliptic curve a bilinear pairing setting of CP-ABE to support digital signing using short signatures, which only require pairing-based cryptography (attributes are not used nor policies). Data is secured in the same way than with DET-ABE but additionally the data is digitally signed. This case, the security level of signatures must be compliant with the one for data encryption. Internally, digital signing hashes the data using the Secure Hash Algorithm SHA-2 and by a series of arithmetic operations produces the digital signature σ. Now, what is stored in a binary file and uploaded to the cloud is the triplet $\{CT_{AES}, CT_{ABE}, \sigma\}$. The general process of AES4SeC signing and encryption is shown in Figure 3.

AES4SeC uses Formatted Boolean Formulas (FBF) to represent access policies. A Formatted Boolean Formula can be expressed as $(F_1, F_2, ..., F_n, t)$, where n and t defines a (n,t)-gate of the FBFs $F_1, F_2, \cdots, F_n$. The value n is implicitly decided by the number of the children, and only the threshold value t is explicit in the formula. F_i is either an attribute or another FBF. For example,
$P_1 : (A \wedge B)$
P_1 as a FBF: $(A, B, 2)$

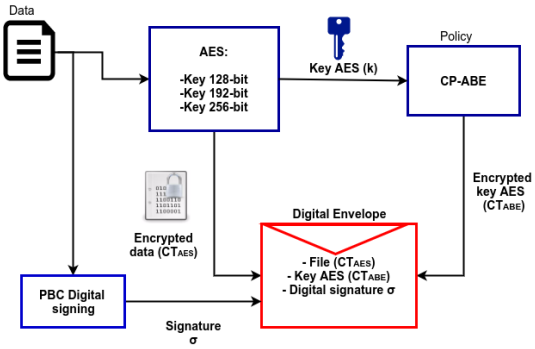

Figure 3: AES4SeC encryption process.

P_2: $(C \vee D)$
P_2 as a FBF: $(C, D, 1)$
P: $(A \wedge B) \wedge (C \vee D)$
P as a FBF: $(P_1, P_2, 2)$

3.3 Qualitative Comparison

The main differences between DET-ABE and AES4SeC include: the security assumptions in each construction, the implementation of access structure, the security services provided and the performance of main operation. The first three are summarized in Table 1. The last one is discussed with more details in the next section.

DET-ABE follows the construction provided by (Bethencourt et al., 2007), which is proven to be secure in the random oracle model. Contrary, AES4SeC uses the construction given by (Waters, 2011), which is proven to be secure in the standard model. The security assumptions in the later are more solid, foundationally sound and practical.

The access structure in DET-ABE is realized as a tree, which can exhibit a better timing when processed. Policies in FBF format can be more expressive but the realization of the access structure as a matrix could have penalties in the timing when processing policies that produce a matrix of high dimension. Finally, although both DET-ABE and AES4SeC guarantee confidentiality and access control, only AES4SeC can also guarantee integrity, authentication, and non-repudiation by incorporating the digital signature of the data at low cost, because the setting for deploying ABE is reused for providing the pairing-based cryptography that requires the digital signature algorithm.

Table 1: Comparison between DET-ABE and AES4SeC.

	DET-ABE	AES4SeC
Security model	Random oracle	Standard
Access structure	Tree	Matrix
Confidentiality?	✓	✓
Access control?	✓	✓
Authentication?	✗	✓
Integrity?	✗	✓
Non-repudiation?	✗	✓

4 EXPERIMENTAL EVALUATION

The main aim of this experimentation was to evaluate general features and performance of two applications that provide confidentiality and access control services to digital content. This content will be shared among users of 4 different organizations using a file sharing system running on a private cloud environment. Each application implements DET-ABE and AES4SeC, both running on a Java 8.0 virtual machine.

4.1 Prototype and Setup

Our interest was focused mainly on evaluating performance of DET-ABE and AES4SeC when providing different levels of security, such as minimum (128-bit keys), medium (192-bit keys) and high (256-bit keys), where one to four attributes can be involved in the encryption policies. The evaluation was motivated by the need for exchanging digital content in a secure way among users that belong to four different organizations by using a file sharing system that runs on a private cloud environment. The sizes of the files being shared in the cloud are 1KB, 1MB, 10MB and 100MB. In this scenario, we assume that the level of sensitivity of digital content has been determined by the content owner, as well as the level of coverage, i.e., define the users that will be able to access the shared content and that belong to an particular organization. For this evaluations, for simplicity we defined four organizations, each one represented by the attributes A, B, C and D, respectively.

The infrastructure used in our experimentation represents a prototype of SkyCDS (Gonzalez et al., 2015), which is a resilient delivery service based on diversified cloud storage. For testing purposes, we use a program that emulates a client generating files that are submitted in a secure way into the cloud storage. This client node is a 64 bits Intel core i7, 2.5Ghz with 8GB in RAM and 1TB of HD running Ubuntu 16.04 LTS as operating system.

Table 2 shows the details of the instances used in SkyCDS. Nevertheless, this article only reports the impact on the file generator node.

Table 2: Characteristics of the cloud instance.

Type	# Instances	Cores	RAM	HD
Metadata	1	2	2GB	100GB
Storage	5	4	6GB	80GB

4.2 Cryptographic Algorithms Settings

Table 3 shows the key sizes used in experimentation for both DET-ABE and AES4SeC. NIST has recommended three security levels expressed in terms of the key size of AES. While 128-bit is intended for protecting sensible data in general purpose applications, 192-bit are for more restrictive domains, as in military or government. The security level of 256-bit is recommended for protecting top secret data and for national security interests.

Digital envelopes in DET-ABE and AES4SeC use AES together with CP-ABE. Thus, these two cryptosystems must offer equivalent security strenght in terms of keys sizes. While brute force is the best attack over AES to obtain the encryption key, for CP-ABE the attack comes by solving the discrete logarithm problem over three sets: G_1, G_2, and G_T. Key sizes in CP-ABE are strongly related with the size of elements in these sets. G_1 and G_2 are sets of elliptic curve points defined over a finite field F_q. G_T is an extension field of F_q with degree k. In this experimentation we use the setting for the key sizes of CP-ABE provided in (Morales-Sandoval et al., 2017), for type F-curves, using an asymetric setting for the pairing defined by the mapping $G_1 \times G_2 \mapsto G_T$.
As a comparison and putting this in perspective, a content that is secured with AES using a 128bit key would require a key of 3072 bits to provide a similar level of security using RSA.

4.3 Metrics and Testbech

The time required for encrypting digital content by both applications was used as main performance metric in this evaluation. It is important to recall that the confidentiality and access control services are provided by the DET-ABE and AES4SeC applications. However, AES4SeC also provides authentication and integrity services by including digital signing functions. For a fair performance comparison between the DET-ABE and AES4SeC applications, it was necessary to switch off the digital signing functions in AES4SeC, limiting its services only to provide confidentiality (AES cipher) and access control (CP-ABE

Table 3: Key sizes for different security levels. Security strength and period of protection are specified by NIST.

Security strength ζ	Period of protection	Key size (bits)				
		AES	G_1	G_2	G_T	Z_r
128	2030-2040	128	512	1024	3072	256
192	>2030	192	1280	2560	7680	640
256	>2030	256	2560	5120	15360	1280

implementation). In order to distinguish between the original AES4SeC application and its incomplete version, without including integrity and authentication services, we named this version of AES4SeC as AES4SeCi. As a first approach, we were interested in evaluating performance in applications providing confidentiality and access control services using a particular CP-ABE strategy. However, after comparing performance of the DET-ABE and AES4SeCi applications, we also evaluated how the digital signing function impacts performance when running the complete version of AES4SeC. This evaluation is useful for decision makers to determine how feasible is to include authentication and integrity services in a file sharing system in a private cloud environment, when using different levels of security and where different number of attributes can be involved in the cipher policies.

As mentioned in Section 3, DET-ABE implements CP-ABE using an access structure over attributes in form of a tree, where its nodes represent threshold gates (AND, OR) and the leaves describe attributes. For this evaluation we only used AND gates, defining the following encryption policies in postfix notation:

- A 1-of-1 : One attribute (A)
- A B 2-of-2 : Two attributes $(A \wedge B)$
- A B 2-of-2 C 2-of-2 : Tree attributes $(A \wedge B \wedge C)$
- A B 2-of-2 C 2-of-2 D 2-of-2 : Four attributes $(A \wedge B \wedge C \wedge D)$

In order to create the digital content used in this evaluation, we generated a set of files of the following sizes: 1KB, 1MB, 10MB and 100MB. Each file was encrypted with the three security levels: minimum (128-bit keys), medium (192-bit keys), high(256-bit keys).

Since AES4SeC is based on the concept of Formatted Boolean Formula (FBF), as was described in Section 3, the four previous policies as FBF formulas were defined as following:

- $F_1 = (A, 1)$
- $F_2 = (A, B, 2)$
- $F_3 = (A, B, C, 3)$
- $F_4 = (A, B, C, D, 4)$

Similar to the DET-ABE evaluation, files of different sizes (1Kb, 1MB, 10MB, and 100MB) were generated in this experiment, and every file was secured considering the tree security levels (minimum, medium and high) and every cipher policy.

The securing process for each file using every encryption policy and every security level was carried out 31 times, and the median was taken as a way to normalize the results for DET-ABE and AES4SeC.

4.4 Performance Evaluation

This section presents a summary of the performance evaluation carried out to DET-ABE and AES4SeC, based on the test scenario and configurations defined in the previous section. Figure 4 shows time required by DET-ABE to secure files of different sizes (vertical axis), considering the four previously defined policies and minimum, medium and high security levels (horizontal axis). Notations in Figure 4 should be read as follows: <name of application>-<size of the encryption key 128,192,256>-<number of attributes considered 1A, 2A, 3A, 4A>. For instance, in Figure 4, the first four bars grouped with the tag DETABE-128-1A in the X axis show the time in seconds (Y axis) required by the DET-ABE application considering a minimum (128bit key) security level and only one attribute (1A).

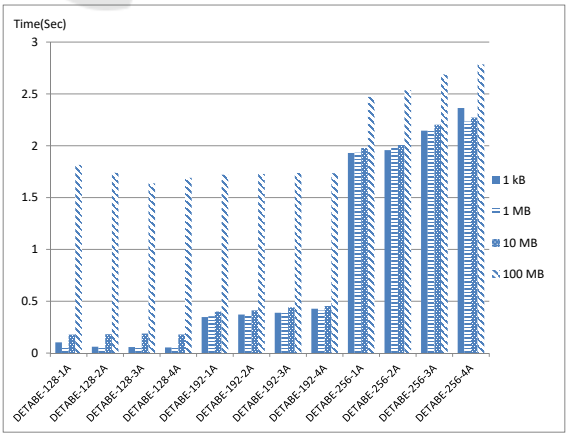

Figure 4: Time required to secure files using DET-ABE.

Focusing on every security level (groups of 128bit, 192bit and 256bit key sizes), it is revealed that the

number of attributes is not having a considerable impact (in terms of absolute time) in DET-ABE performance. For instance, the difference between the maximum and minimum time required for securing a file of 100MB with a 192bit key using one, two, three or four attributes is only about 9 milliseconds. For the user perspective, this means that they will not notice the difference when sharing files secured using different encryption policies. Something similar happens with the time required to secure files whose size is ranging from 1KB to 10MB using DET-ABE in a particular security level. The users will not notice important differences when sharing files with this range of size as the difference in performance is null. It is clear in Figure 4 that, as expected, impact on performance is observed when DET-ABE moves to a higher security level. In this scenario, when moving from minimum security level to medium security level a performance degradation of about 1.5x is perceived (e.g., 150% more time is required for securing a file of 10MB). Moving to the next security level, from medium to high, performance goes down about 4x, e.g. 400% more time required for securing a file of 10MB), and it gets worst when moving from minimum to high security level, decreasing more than 11x, e.g., 1100% more time required for securing a file of 10MB). Even though the percentage of additional time required for securing documents with a high security level seems unfeasible, in absolute terms it will depend on the number of files to be secured. For example, for securing a 100MB file in a high security level using DET-ABE with 4 attributes only requires 2.8seconds, which is a very acceptable value compared with the time that will be required to transfer the file in a commercial network; at this rate we could secure more than 20 files per minute, which is an acceptable value for some organizations.

The next evaluation is focused on the performance produced by AES4SeCi (AES4SeC version similar to DET-ABE). As we can see in Figure 5, AES4SeCi shows a very similar behavior to DET-ABE in terms of low negative performance impact, when the number of attributes changes in a particular security level and securing files whose size is ranging from 1KB to 10MB. The evident negative performance impact in a specific security level, especially in minimum and medium, occurs when securing files of 100MB. Notation rules in this figure are similar to those presented for DET-ABE, for instance, AES4SECi-128-1A refers to the AES4SeCi application using a 128bit key with one attribute.

The most important impact on performance degradation produced by AES4SECi arises when moving from lower to higher security levels. In this sense,

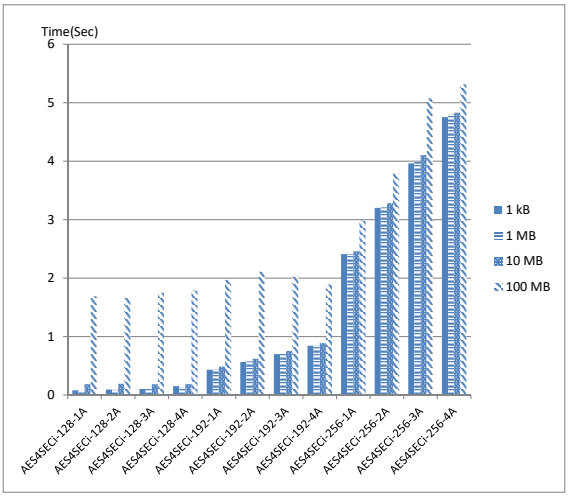

Figure 5: Time required to secure files using AES4SeCi.

its performance degradation is higher than DET-ABE, for example, the performance degradation obtained when moving from minimum to high security level when securing a 10MB file is about 24x, it seems unfeasible, but in absolute values it means more than 5 seconds of difference.

4.5 Comparison

Taking into account that our main interest is focused on providing confidentially and access control to a file sharing system, where users from four different organization (or four attributes in our evaluation context) will exchange content using different security levels, in Figure 6, we show a comparison between DET-ABE and AES4SeCi, both using 4 attributes implementing different security levels, minimum, medium and maximum.

As we can see, with minimum security levels and using four attributes DET-ABE and AES4SeCi show very similar performance. This behavior changes gradually with medium security levels and an evident difference occurs with high security levels. The average performance degradation rate of AES4SeCi compared with DET-ABE is about 110%, especially in medium and high security levels. In our test scenario, these results reveal that the access structures over attributes in form of a tree implemented in DET-ABE offer similar performance than the matrix implementation and the use of the concept of Formatted Boolean Formula (FBF) in AES4SeCi when a minimum security level is required. However, the tree structures of CP-ABE implemented in DET-ABE allow it to improve performance in higher security levels when compared with the matrix implementation of AES4SeC.

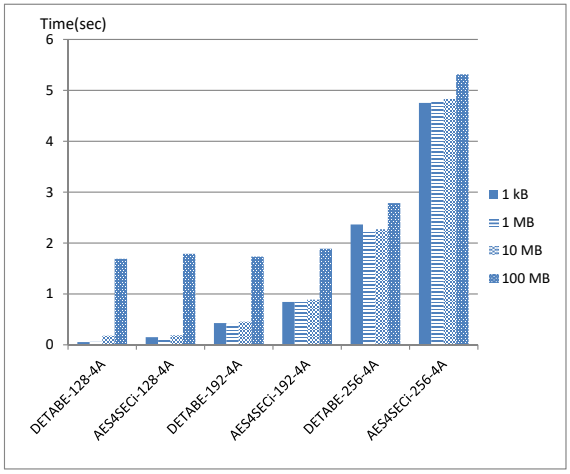

Figure 6: DET-ABE and AES4SeCi comparison using encryption policies with 4 attributes.

Since AES4SeC also provides the authentication and integrity services, which are important security requirements in some organizations, we carried out an experiment to verify how these functions impact in the performance of AES4SeC, running the incomplete (AES4SeCi) and complete (AES4SeC) versions of AES4SeC on the same previous scenario using encryption policies with four attributes.

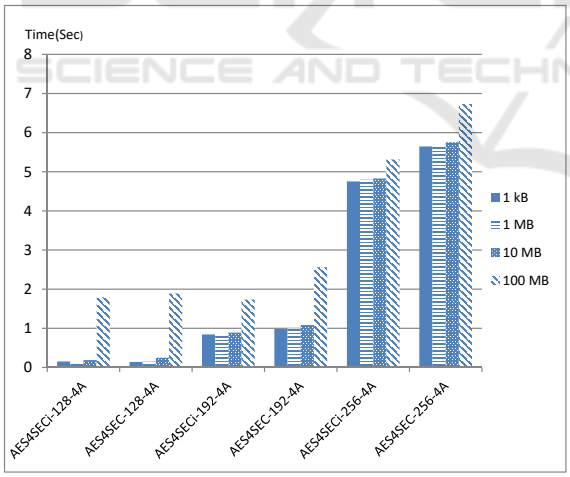

Figure 7: Impact of signing functions in AES4SeC.

Figure 7 reveals that the degradation impact in performance of the signing functions of AES4SeC is almost null when minimum security levels are used. The largest impact occurs in high security level with 27% of performance degradation, which occurs when securing files of 100MB. This results are still encouraging for organizations requiring all of the security services (confidentiality, access control, authentication and integrity services).

5 CONCLUSION

This paper presented useful insights on performance of two Java applications, DET-ABE and AES4SeC, which implement different CP-ABE strategies to provide confidentiality and access control security services. These two applications offer a Java programming interface that hides the complexity of implementing encryption policies and Attribute-Based Encryption (CP-ABE) algorithms, security parameters and type and size of elliptic curves needed for minimum (128-bit key), medium (192-bit key) and maximum (256-bit key) security levels, making simple their use and inclusion as building blocks in end user applications. AES4SeC also provides authentication and integrity services, for a fair comparison these functions were switch off, named this incomplete version AES4SeCi. Experiments carried out on DET-ABE and AES4SeCi showed that the impact on performance is almost null when the number of attributes change in the encryption process, when a particular security level is applied. However, they suffer an evident performance impact when moving from low to high security levels. An interesting finding was that the two applications showed very similar performance when securing files of 1KB, 1MB and 10MB and 100MB using a minimum security level. However AES4SeCi showed lower performance when medium and high security levels were applied, almost doubling the time required for securing the same group of files. In general terms, results revealed that the CP-ABE implementation in DET-ABE, based on tree structures, improves performance of the matrix implementation made in AES4SeCi, which is based on the concept of Formatted Boolean Formula (FBF). It is worth recalling that the complete version of AES4SeC also provide integrity and authentication services, in this sense, experiments showed that impact on performance of its signing functions is almost null when a minimum security level is used, and time required to secure the same group of files in medium and high security levels (less than 5 and 6 seconds respectively in average), represents an attractive option for users demanding secure file sharing with integrity and authentication functionalities.

ACKNOWLEDGEMENTS

This work was partially supported by the sectoral fund of research, technological development and innovation in space activities of the Mexican National Council of Science and Technology (CONACYT) and the Mexican Space Agency (AEM), project No.262891.

REFERENCES

Bethencourt, J., Sahai, A., and Waters, B. (2007). Ciphertext-policy attribute-based encryption. In *Security and Privacy, 2007. SP'07. IEEE Symposium on*, pages 321–334. IEEE.

Bobba, R., Khurana, H., and Prabhakaran, M. (2009). Attribute-sets: A practically motivated enhancement to attribute-based encryption. In *European Symposium on Research in Computer Security*, pages 587–604. Springer.

Chow, R., Golle, P., Jakobsson, M., Shi, E., Staddon, J., Masuoka, R., and Molina, J. (2009). Controlling data in the cloud: outsourcing computation without outsourcing control. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, pages 85–90. ACM.

Fu, J.-y., Huang, Q.-l., Yang, Y.-x., et al. (2014). Secure personal data sharing in cloud computing using attribute-based broadcast encryption. *The Journal of China Universities of Posts and Telecommunications*, 21(6):4577–51.

Gonzalez, J. L., Perez, J. C., Sosa-Sosa, V. J., Sanchez, L. M., and Bergua, B. (2015). SkyCDS: A resilient content delivery service based on diversified cloud storage. *Simulation Modelling Practice and Theory*, 54:64–85.

Koo, D., Hur, J., and Yoon, H. (2013). Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage. *Computers & Electrical Engineering*, 39(1):34–46.

Liu, J., Huang, X., and Liu, J. K. (2015). Secure sharing of personal health records in cloud computing: ciphertext-policy attribute-based signcryption. *Future Generation Computer Systems*, 52:67–76.

Liu, Z. and Wong, D. S. (2016). Practical attribute-based encryption: traitor tracing, revocation and large universe. *The Computer Journal*, 59(7):983–1004.

Menezes, A. J., Van Oorschot, P. C., and Vanstone, S. A. (1996). *Handbook of applied cryptography*. CRC press.

Morales-Sandoval, M. and Diaz-Perez, A. (2015). DET-ABE: A Java API for data confidentiality and fine-grained access control from attribute based encryption. In *IFIP International Conference on Information Security Theory and Practice*, pages 104–119. Springer.

Morales-Sandoval, M., Gonzalez-Compean, J. L., Diaz-Perez, A., and Sosa-Sosa, V. J. (2017). A pairing-based cryptographic approach for data security in the cloud. *International Journal of Information Security*, pages 1–21.

Sahai, A. and Waters, B. (2005). Fuzzy identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 457–473. Springer.

Saikeerthana, R. and Umamakeswari, A. (2015). Secure data storage and data retrieval in cloud storage using cipher policy attribute based encryption. *Indian Journal of Science and Technology*, 8(S9):318–325.

Theoharidou, M., Papanikolaou, N., Pearson, S., and Gritzalis, D. (2013). Privacy risk, security, accountability in the cloud. In *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, volume 1, pages 177–184. IEEE.

Wan, Z., Liu, J., and Deng, R. H. (2012). HASBE: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE Transactions on information forensics and security*, 7(2):743–754.

Waters, B. (2011). Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *International Workshop on Public Key Cryptography*, pages 53–70. Springer.

Zickau, S., Thatmann, D., Butyrtschik, A., Denisow, I., and Küpper, A. (2016). Applied Attribute-based Encryption Schemes. *19th International ICIN Conference - Innovations in Clouds, Internet and Networks*.