A Systems Approach to Cyber Security
A. Roychoudhury and Y. Liu (Eds.)

Attribute-Based Secure Messaging in the Public Cloud

Zhi Yuan Poh, Hui Cui, Robert H. Deng, and Yingjiu Li
School of Information Systems, Singapore Management University

Abstract. Messaging systems operating within the public cloud are gaining popularity. To protect message confidentiality from the public cloud including the public messaging servers, we propose to encrypt messages in messaging systems using Attribute-Based Encryption (ABE). ABE is an one-to-many public key encryption system in which data are encrypted with access policies and only users with attributes that satisfy the access policies can decrypt the ciphertexts, and hence is considered as a promising solution for realizing expressive and fine-grained access control of encrypted data in public servers. Our proposed system, called Attribute-Based Secure Messaging System with Outsourced Decryption (ABSM-OD), has three key features: enabling expressive and fine-grained access control of encrypted messages by users, supporting outsourced decryption to the cloud while without compromising confidentiality of decrypted messages, and allowing server-aided revocation to provide effective and instant user revocations.

Keywords. Attribute-Based Encryption, Secure Messaging, Outsourced Decryption

1. Introduction

Messaging systems such as WhatsApp, Facebook Messenger, WeChat, Line, Viber, etc are becoming very popular. Users from different localities have their preferred choices of messaging services[1]. Since these messaging systems reside in the public domain and are subjected to threats on the Internet, security savvy users might be reluctant to trust the service providers to protect the privacy of their messages and there is a growing demand to provide end-to-end encryption in public messaging services. Furthermore, in a threat landscape study, instant messaging platforms are becoming attack vectors which can result in further damages[2].

Messages can be in the form of Electronic Mail (Email), Short Message Service (SMS), Instant Messaging (IM), etc which allow users to share information and collaborate effectively. Previous messaging services focus on functionality over security and the threats on the Internet poses challenges on these messaging services. In 1991, Phil Zimmermann introduced the Pretty Good Privacy (PGP)[3] to protect the confidentiality and authenticity of emails. Later in 1995, Secure/Multipurpose Internet Mail Extensions (S/MIME) (currently version 3.2[4]) was introduced to provide a standard way to protect

[1] https://www.similarweb.com/blog/worldwide-messaging-apps
[2] https://www.fortinet.com/content/dam/fortinet/assets/white-papers/Executive-Summary-CTAP.pdf
[3] PGP https://en.wikipedia.org/wiki/Pretty_Good_Privacy
[4] S/MIME version 3.2 Message Specification https://tools.ietf.org/html/rfc5751

Multipurpose Internet Mail Extensions (MIME) email messages. However, due to the complexity of the solutions and the need for user involvement, both PGP and S/MIME are not widely adopted today [14]. In 2014, Google started implementing the OpenPGP[5] standard (IETF RFC4880[6]) as a Chrome Extension (End-To-End)[7] to enhance the messaging security within the browser. However, PGP and S/MIME are based on the traditional public key encryption hence have a drawback in scalability. Transport messaging protocols like Secure Sockets Layer (SSL), Transport Layer Security (TLS), and Secure Shell (SSH), etc were introduced and enhanced to protect the delivery of messages. However, transport protection is inadequate in protecting message security and privacy as these messaging systems assume that the servers are trusted.

The Signal protocol is used in Signal [7], WhatsApp[8], Facebook Messenger[9] and Allo[10] to provide end-to-end encryption. These systems rely on trusted servers to exchange users' public keys during communication. With the threats in the public domain, key management and transport protection are insufficient in protecting the data security of messaging systems. In addition, these messaging systems assume that the servers residing in the public domain are trusted. Public threats and malicious insiders can reduce the data security and user privacy of these messaging systems.

ABE is a one-to-many public-key encryption where private keys of users and access policy of encrypted data are based on user attributes. ABE allows a sender to embed access policy with encrypted data and only authorized users will be able to gain access to the original data and is widely considered as a promising technique for providing expressive and fine-grained access control of end-to-end encrypted data.

Some studies have considered secure messaging from an access control perspective by integrating ABE with existing messaging systems [4][11][17]. However, these systems require the users to perform full decryption which can be resource intensive for mobile users, especially if the ciphertexts have complex access policies. In [4] and [11], revocation is achieved by issuing decryption keys with expiry date, however a direct revocation approach might be more desirable.

In this work, we propose an Attribute-Based Secure Messaging System with Outsourced Decryption (ABSM-OD) which provides end-to-end message security on the cloud. ABSM-OD is designed to operate in environments where the messaging servers reside in a public untrusted domain. Messages are stored on the cloud and only authorized users will be able to obtain the original data. Specifically, ABSM-OD possesses the following three features.

Fine-grained Access Control of Encrypted Messages The use of ABE allows expressive and fine-grained access control to be enforced on encrypted messages which ensures end-to-end message confidentiality.

Outsourced Decryption The computation of ABE decryption is offloaded to the cloud, hence keeping the resource requirements on the users to the minimal and without exposing users' messages to the cloud.

[5]MIME Security with OpenPGP https://tools.ietf.org/html/rfc3156
[6]OpenPGP Message Format https://tools.ietf.org/html/rfc4880
[7]End-To-End https://github.com/google/end-to-end
[8]https://whispersystems.org/blog/whatsapp-complete/
[9]https://whispersystems.org/blog/facebook-messenger/
[10]https://whispersystems.org/blog/allo/

Effective User Revocation Compromised users can be directly revoked from the system with the use of server-aided revocation technique. Users and attributes can be managed effectively within the system.

With the features of ABSM-OD, enterprises operating their messaging systems in the cloud will be able to preserve message security. Since only ciphertexts are available on the cloud, cloud service providers will have no access to the underlying messages. If the messaging systems are compromised, adversaries will only obtain the ciphertexts and not the original messages. Furthermore, enterprises will be able to manage the access control of messages effectively within a cloud environment.

2. Preliminaries

This section describes the notions that are to be used in the construction of ABSM-OD.

Symmetric Key Encryption: The scheme consists of the following:

- Key Generation $k \leftarrow \mathsf{Gen}^{SE}$: It outputs a random key k.
- Encryption $\mathsf{CT}_{\mathsf{M}} \leftarrow \mathsf{Enc}^{SE}(k, \mathsf{M})$: Takes a key k and message M. It encrypts message M with key k and outputs ciphertext CT_{M}.
- Decryption $\mathsf{M}/\perp \leftarrow \mathsf{Dec}^{SE}(k, \mathsf{CT}_{\mathsf{M}})$: Takes a key k and ciphertext CT_{M}. It decrypts ciphertext CT_{M} with key k and outputs message M or $\perp$ indicates error.

Digital Signature: The scheme consists of the following:

- Key Generation $(pk, sk) \leftarrow \mathsf{Gen}^{DS}$: It outputs a signing key sk and verifying key pk.
- Message Signing $\sigma_{\mathsf{M}} \leftarrow \mathsf{Sign}^{DS}(sk, \mathsf{M})$: Takes a signing key sk and message M. It signs message M with signing key sk and outputs message signature σ_{M}.
- Message Verification $1/0 \leftarrow \mathsf{Verify}^{DS}(pk, \mathsf{M}, \sigma_{\mathsf{M}})$: Takes a verifying key pk, message M and message signature σ_{M}. It verifies message M with verifying key pk. It outputs 1 when the message is valid, 0 if otherwise.

Ciphertext-Policy ABE with outsourced decryption: An ABE with outsourcing decryption scheme proposed by Green et al. [9] that we denote as ABE-OD consists of the following:

- $(\mathsf{MSK}, \mathsf{MPK}) \leftarrow \mathsf{Setup}^{ABE}(\kappa, \mathbb{U})$: Takes a security parameter κ and generates master secret key MSK and master public key MPK.
- $\mathsf{CT}_{A_{\mathsf{M}}} \leftarrow \mathsf{Encrypt}^{ABE}(\mathsf{MPK}, \mathsf{M}, A_{\mathsf{M}})$: Encrypts M using MPK and access structure A_{M} and outputs an ABE ciphertext $\mathsf{CT}_{A_{\mathsf{M}}}$.
- $(\mathsf{DK}_{A_u}, \mathsf{TK}_{A_u}) \leftarrow \mathsf{KeyGen}^{ABE}(\mathsf{MSK}, A_u)$: Generates transformation and decryption keys using input master secret key MSK and user attributes A_u. It outputs a decryption key DK_{A_u} and a transformation key TK_{A_u}.
- $\mathsf{CT}_{out} \leftarrow \mathsf{Transform}^{ABE}(\mathsf{TK}_{A_u}, \mathsf{CT}_{A_{\mathsf{M}}})$: Partially decrypts ciphertext $\mathsf{CT}_{A_{\mathsf{M}}}$ using transformation key TK_{A_u} and outputs a partially decrypted ciphertext CT_{out}.
- $\mathsf{M}/\perp \leftarrow \mathsf{Decrypt}^{ABE}(\mathsf{DK}_{A_u}, \mathsf{CT}_{out})$: Decrypts partially decrypted ciphertext CT_{out} with decryption key DK_{A_u} and outputs a message M or $\perp$ where $\perp$ indicates error.

3. Proposed System Overview

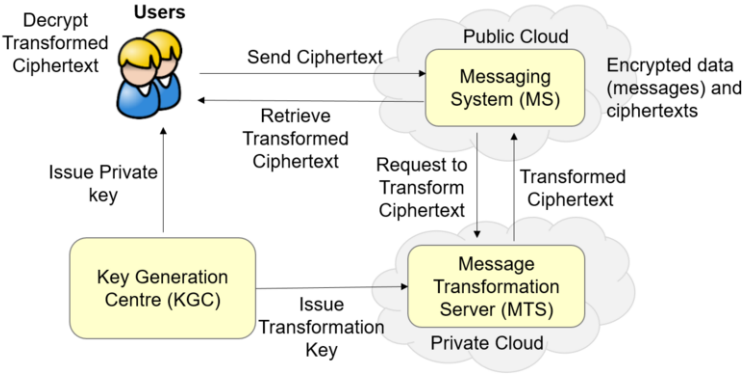

Figure 1. Attribute-Based Secure Messaging System with Outsourced Decryption

Figure 1 illustrates the overall proposed system architecture of ABSM-OD. The system comprises of four parties namely Key Generation Centre (KGC), Message Transformation Server (MTS), Messaging System (MS) and users. The KGC manages user attributes, issues decryption keys to users based on their attributes and generates transformation keys to the MTS. The MTS determines recipients using the access policies of the ciphertexts and transforms the ciphertexts for each recipient. The MS stores two types of ciphertexts, ciphertexts received from the users and ciphertexts transformed by the MTS. Users interact with the MS to send/retrieve ciphertexts.

The KGC generates the public parameter and the master private key, and issues MTS with a signing and verifying key pair. Before a user can send/retrieve ciphertexts, a user needs to be registered in the system. The user generates a signing and verifying key pair and provides the verifying key to the KGC. The KGC registers the user with the verifying key, a set of user attributes and an user identifier, and generates ABE attribute keys (i.e. transformation and decryption keys) for the user. The KGC issues the decryption key to the user and distributes the user identifier, user verifying key, user attributes and transformation key to the MTS. Thereafter, the user will be able to send/retrieve ciphertexts.

When the user wants to send a message, the user encrypts the message with an access policy and signs the ciphertext with a timestamp and the user identifier. The user then sends the ciphertext, timestamp, user identifier and signature to the MS. When the MS receives the ciphertext, the MS requests the user verifying key from the MTS using the user identifier. If the user is unauthorized or revoked, no user verifying key is returned and the MS will discard the ciphertext. If the user is valid, the MTS will return the user verifying key to the MS for ciphertext verification. If MS fails to verify the ciphertext, the ciphertext will be discarded. Once the ciphertext is verified, the MS will forward the verified ciphertext to the MTS. As the MS is subjected to attacks in the public domain, the MTS will verify the ciphertext again. If the verification fails, MTS will discard the ciphertext. Once the ciphertext is verified, MTS resolves the recipients based on the access policy of the ciphertext, transforms the ciphertext for each recipient and signs on the transformed ciphertexts. Thereafter, the MTS requests for MS to store the transformed ciphertexts. When the users request for the ciphertexts, the MS will send the

transformed ciphertexts to the users. Users will verify the transformed ciphertexts using the MTS verifying key. If verification fails, the transformed ciphertexts are discarded. If the ciphertexts are valid, users will use their decryption keys to decrypt the ciphertexts to obtain the original messages.

When a user is compromised, KGC will remove the user verifying key, user attributes and transformation key from the MTS. Subsequently, the compromised user will not be able to retrieve existing and new ciphertexts and any new ciphertexts generated by the compromised user will be rejected by the MS or MTS.

3.1. Threat Model

We assume that the KGC is trusted. MTS is honest-but-curious such that it will honestly follow the protocol as required but will attempt to obtain sensitive information such as the message encrypted in a ciphertext. MTS operates within a private cloud and is assumed not to collude with users. MS is untrusted and resides in the public cloud.

Any adversary will be able to compromise and gain access to the MS within the public domain. If the adversary gain access to the MS, the ciphertexts and the transformed ciphertexts should not reveal sensitive information.

3.2. Design Objectives

- **Confidentiality**
  The main objective is to design a secure messaging system that can provide end-to-end message confidentiality. The sender determines the access policies of the encrypted messages and only privileged users can access the underlying messages.

- **Efficient Decryption**
  As mobile users may have limited resources, it is desirable to reduce the computational cost of users in decryption. Towards this end, we adopt an ABE with outsourced decryption [9] to offload decryption operations to the cloud.

- **Message Authenticity**
  With the prevalent of spam and malicious messages, it can be challenging for users to recognize if received messages are valid, hence it will be desirable that received messages are indeed authentic. As the sender will sign on the message ciphertexts and MTS will sign on the transformed ciphertexts, the signing of ciphertexts will provide the authenticity of the messages.

- **Traceability**
  The source and changes of ciphertexts can be crucial for audit purposes, hence it is desirable for the system to maintain the traceability of the ciphertexts. To achieve traceability, user identifiers are included into the ciphertexts and digital signatures are used by the sender and MTS to sign the ciphertexts.

- **Revocation**
  Another consideration is to provide an effective user and attribute revocation, any

revoked users should no longer be able to access the system to send/retrieve messages. The use of direct revocation of users with server-aided technique achieves this objective.

4. ABSM-OD Construction

In this section, we describe a concrete construction of ABSM-OD based on ABE-OD [9], symmetric encryption and digital signature.

Assuming that the KGC keeps a list $KGCSTORE$, storing the user identifier oid_u, user verifying key pk_u, user attributes A_u. A tuple in $KGCSTORE$ is represented as $\langle oid_u, pk_u, A_u \rangle$.

The MTS keeps a list $MTSSTORE$, storing the user identifier oid_u, user verifying key pk_u, user attributes A_u, transformation key TK_u. A tuple in $MTSSTORE$ is represented as $\langle oid_u, pk_u, A_u, \mathsf{TK}_u \rangle$.

The MS keeps two types of ciphertexts (message ciphertexts and transformed ciphertexts) in the list $MSGSTORE$. A message ciphertext includes a message identifier mid_{M}, user identifier oid_u, access structure A_{M}, message timestamp ts_{M}, message ciphertext $\mathsf{CT}_{\mathsf{M}}^{ABE}$ and ciphertext signature $\sigma_{\mathsf{CT}_{ABE}}$. A tuple of message ciphertext in $MSGSTORE$ is represented as $\langle mid_{\mathsf{M}}, oid_u, A_{\mathsf{M}}, ts_{\mathsf{M}}, \mathsf{CT}_{\mathsf{M}}^{ABE}, \sigma_{\mathsf{CT}_{ABE}} \rangle$. A transformed ciphertext includes a new message identifier mid_{TM_i}, sender user identifier oid_s, recipient user identifier oid_{r_i}, a creation timestamp ts_{TM_i}, transformed ciphertext $\mathsf{CT}_{\mathsf{TM}_i}^{ABE}$ and transformed ciphertext signature σ_{T_i}. A tuple of transformed ciphertext in $MSGSTORE$ is represented as $\langle mid_{\mathsf{TM}_i}, oid_s, oid_{r_i}, ts_{\mathsf{TM}_i}, \mathsf{CT}_{\mathsf{TM}_i}^{ABE}, \sigma_{T_i} \rangle$.

System Initialization $(\mathsf{MPK}, \mathsf{MSK}, pk_{mts}, sk_{mts}) \leftarrow \mathsf{Setup}(\kappa, \mathbb{U})$: During the initialization, KGC runs the ABE setup to create the master secret key MSK and public parameters MPK (i.e $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}^{ABE}(\kappa, \mathbb{U})$). In addition, KGC issues a pair of signing key and verifying key to the MTS (i.e. $(pk_{mts}, sk_{mts}) \leftarrow \mathsf{Gen}^{DS}$ for MTS).

User Registration $(\mathsf{DK}_u, \mathsf{TK}_u) \leftarrow \mathsf{RegisterUser}(\mathsf{MSK}, oid_u, pk_u, A_u)$: To allow a user u to use the system, KGC needs to register the user u in the system. First, the user u generates a signing and verifying key pair $(pk_u, sk_u) \leftarrow \mathsf{Gen}^{DS}$ and provides the verifying key pk_u to the KGC. KGC registers the user u in the system with the user identifier oid_u, user verifying key pk_u, user attributes A_u and provides the user identifier oid_u, user verifying key pk_u to the MTS. KGC performs the *Attribute Key Generation* (i.e $(\mathsf{DK}_u, \mathsf{TK}_u) \leftarrow \mathsf{KeyGen}(oid_u, \mathsf{MSK}, A_u)$) to update transformation key TK_u and user attributes A_u of user u to the MTS and issues the decryption key DK_u to user u.

Attribute Key Generation $(\mathsf{DK}_u, \mathsf{TK}_u) \leftarrow \mathsf{KeyGen}(oid_u, \mathsf{MSK}, A_u)$: When a user u requires a set of attribute keys, KGC performs the ABE Key Generation (i.e $(\mathsf{DK}_u, \mathsf{TK}_u) \leftarrow \mathsf{KeyGen}^{ABE}(\mathsf{MSK}, A_u)$) to obtain the decryption key DK_u and transformation key TK_u of user u.

Attribute Key Update $(\mathsf{DK}_u^{'}, \mathsf{TK}_u^{'}) \leftarrow \mathsf{KeyUpdate}(oid_u, \mathsf{MSK}, A_u^{'})$: When the user attributes of user u changes from A_u to $A_u^{'}$, KGC updates the user attributes from A_u to $A_u^{'}$.

KGC revokes the transformation key TK_u of user u on the MTS. Also, KGC performs *Attribute Key Generation* (i.e $(\mathsf{DK}_u^{'}, \mathsf{TK}_u^{'}) \leftarrow \mathsf{KeyGen}(oid_u, \mathsf{MSK}, A_u^{'})$) and updates MTS with the new transformation key $\mathsf{TK}_u^{'}$ and new user attributes $A_u^{'}$. KGC issues the new decryption key $\mathsf{DK}_u^{'}$ to user u.

User Revocation $1/0 \leftarrow \mathsf{RevokeUser}(oid_u)$: When a user u is compromised, the user u will need to be revoked in the system. KGC and MTS will need to revoke the user verifying key pk_u, user attributes A_u and transformation key TK_u of user u. The tuples in *KGCSTORE* and *MTSSTORE* will be updated as $\langle oid_u, -, - \rangle$ and $\langle oid_u, -, -, - \rangle$ respectively. Once revoked, any verification of ciphertexts signed by user u with identifier oid_u at the MS and MTS will fail and the MTS will not transform any new ciphertexts for user u.

Message Encryption $\mathsf{CT}_{ABE} \leftarrow \mathsf{EncryptMessage}(\mathsf{MPK}, \mathsf{M}, A_\mathsf{M}, oid_u, sk_u)$: When a user u wants to send a message M, the user u will specify the access structure A_M to encrypt the message. The user u generates a random key $k_\mathsf{M} \leftarrow \mathsf{Gen}^{SE}$ to encrypt the message M (i.e $\mathsf{CT}_\mathsf{M} = \mathsf{Enc}^{SE}(k_\mathsf{M}, \mathsf{M})$). The user u performs the ABE encryption on the random key k_M with access structure A_M to get the key ciphertext $\mathsf{CT}_{k_\mathsf{M}}^{ABE} = \mathsf{Encrypt}^{ABE}(\mathsf{MPK}, k_\mathsf{M}, A_\mathsf{M})$. The ciphertext $\mathsf{CT}_\mathsf{M}^{ABE} = (\mathsf{CT}_\mathsf{M}, \mathsf{CT}_{k_\mathsf{M}}^{ABE})$ forms the ciphertext for the message M. The ciphertext $\mathsf{CT}_\mathsf{M}^{ABE}$, a message identifier mid_M, user identifier oid_u and ciphertext timestamp ts_M forms the ciphertext $\mathsf{CT}_{ABE_\mathsf{M}} = (mid_\mathsf{M}, oid_u, A_\mathsf{M}, ts_\mathsf{M}, \mathsf{CT}_\mathsf{M}^{ABE})$ for the MS. Before sending the ciphertext $\mathsf{CT}_{ABE_\mathsf{M}}$, the user u signs the ciphertext $\mathsf{CT}_{ABE_\mathsf{M}}$ with the user signing key sk_u (i.e $\sigma_{\mathsf{CT}_{ABE}} \leftarrow \mathsf{Sign}^{DS}(sk_u, \mathsf{CT}_{ABE_\mathsf{M}})$) and sends the ciphertext $\mathsf{CT}_{ABE} = (\mathsf{CT}_{ABE_\mathsf{M}}, \sigma_{\mathsf{CT}_{ABE}})$ to MS.

MS and MTS will verify the ciphertext $\mathsf{CT}_{ABE} = (\mathsf{CT}_{ABE_\mathsf{M}}, \sigma_{\mathsf{CT}_{ABE}})$ using the user verifying key pk_u (i.e $\mathsf{Verify}^{DS}(pk_u, \mathsf{CT}_{ABE_\mathsf{M}}, \sigma_{\mathsf{CT}_{ABE}})$). Once the ciphertext is verified, the MTS determines recipients with attributes that satisfies the access structure A_M of the ciphertext CT_{ABE}. With the list of recipients R_{A_M}, the MTS uses the transformation key TK_{r_i} of each recipient $r_i \in R_{A_\mathsf{M}}$ to transform the ciphertext CT_{ABE}.

Message Transformation $\mathsf{CT}_{\mathsf{TM}_i} \leftarrow \mathsf{TransformMessage}(\mathsf{CT}_{ABE}, oid_{r_i}, \mathsf{TK}_{r_i}, sk_{mts})$: The MTS parses the ciphertext $\mathsf{CT}_{ABE} = (\mathsf{CT}_{ABE_\mathsf{M}}, \sigma_{\mathsf{CT}_{ABE}})$, then parses ciphertext $\mathsf{CT}_{ABE_\mathsf{M}} = (mid_\mathsf{M}, oid_u, A_\mathsf{M}, ts_\mathsf{M}, \mathsf{CT}_\mathsf{M}^{ABE})$ and transforms the ciphertext $\mathsf{CT}_\mathsf{M}^{ABE}$ for a user u. The MTS transforms the key ciphertext $\mathsf{CT}_{k_\mathsf{M}}^{ABE}$ ($\mathsf{CT}_\mathsf{M}^{ABE} = (\mathsf{CT}_\mathsf{M}, \mathsf{CT}_{k_\mathsf{M}}^{ABE})$) to get the transformed key ciphertext $\mathsf{CT}_{out_i}^{ABE} = \mathsf{Transform}^{ABE}(\mathsf{TK}_{r_i}, \mathsf{CT}_{k_\mathsf{M}}^{ABE})$ for a user u. Once the key ciphertext $\mathsf{CT}_{k_\mathsf{M}}^{ABE}$ is transformed, MTS needs the MS to store the transformed ciphertext $\mathsf{CT}_{out_i}^{ABE}$. The ciphertext $\mathsf{CT}_{\mathsf{TM}_i}^{ABE} = (\mathsf{CT}_\mathsf{M}, \mathsf{CT}_{out_i}^{ABE})$ forms the transformed ciphertext for user u. A transformed message identifier mid_{TM_i}, sender user identifier oid_u, recipient user identifier oid_{r_i} and timestamp ts_{TM_i} form the transformed ciphertext $\mathsf{CT}_{T_i} = (mid_{\mathsf{TM}_i}, oid_u, oid_{r_i}, ts_{\mathsf{TM}_i}, \mathsf{CT}_{\mathsf{TM}_i}^{ABE})$ for user u that will be stored by MS. MTS signs the transformed ciphertext CT_{T_i} with the MTS signing key sk_{mts} to get the

signature σ_{T_i} (i.e $\sigma_{T_i} \leftarrow \mathsf{Sign}^{DS}(sk_{mts}, \mathsf{CT}_{T_i})$). The MTS will then send the ciphertext $\mathsf{CT}_{\mathsf{TM}_i} = (\mathsf{CT}_{T_i}, \sigma_{T_i})$ to the MS. It outputs the transformed ciphertext $\mathsf{CT}_{\mathsf{TM}_i}$.

Message Decryption $\mathsf{M}/\perp \leftarrow \mathsf{DecryptMessage}(\mathsf{CT}_{\mathsf{TM}_i}, pk_{mts}, \mathsf{DK}_{r_i})$: When a user u receives the transformed ciphertext $\mathsf{CT}_{\mathsf{TM}_i} = (\mathsf{CT}_{T_i}, \sigma_{T_i})$, the user u verifies the transformed ciphertext $\mathsf{CT}_{\mathsf{TM}_i}$ with MTS verifying key pk_{mts} (i.e $\mathsf{Verify}^{DS}(pk_{mts}, \mathsf{CT}_{T_i}, \sigma_{T_i})$). Once the transformed ciphertext is verified, the user will decrypt the transformed key ciphertext $\mathsf{CT}_{out_i}^{ABE}$ using the decryption key DK_{r_i} to retrieve the random key $k_{\mathsf{M}}/\perp = \mathsf{Decrypt}^{ABE}(\mathsf{DK}_{r_i}, \mathsf{CT}_{out_i}^{ABE})$ or $\perp$ indicates error.

If the user u is able to retrieve the random key k_{M}. The user u performs the decryption to retrieve the message $\mathsf{M}/\perp = \mathsf{Dec}^{SE}(k_{\mathsf{M}}, \mathsf{CT}_{\mathsf{M}})$. If the decryption fails, an error $\perp$ will be returned to the user indicating an error.

4.1. Security Analysis

This section analyses the security of the system.

- **Confidentiality**
  Since all the ciphertexts are stored in the encrypted forms, malicious users whose attributes do not satisfy the access policy of the ciphertext cannot obtain the content of the underlying message. Therefore, our ABSM-OD system preserves the confidentiality of the data users.

- **Efficient Decryption**
  With the use of ABE-OD scheme, the MTS will help the privileged users with decryption and each user will only be required to perform the final decryption to retrieve the original message, hence reducing the computational cost requirements on the users.

- **Message Authenticity**
  Any modification of ciphertexts will be detected by the signature verification.

- **Traceability**
  When some ciphertexts or transformed ciphertexts are found with issues, the ciphertexts can be traced back to the source according to the ciphertext signature.

- **Revocation**
  Once a user is revoked from the system, KGC updates the list in the MTS. The ciphertext with a signature signed by a user that is not in the user list in the MTS will not be accepted by MTS. Also, no ciphertext will be transformed for revoked users as there is no transformation key in the MTS for revoked user. Users are effectively revoked from the system.

5. Implementation

We implemented the ABSM-OD as a Simple Chat System in Java. Tests were conducted for the main functions of the system. In the tests, we measure the recipients resolution

time, encryption time, transformation time and decryption time with different number of attributes and data size. The recipients resolution time is the time taken to determine the number of recipients based on the access policy of a ciphertext. Other performance overheads like database, disk and network latency which are dependent on the deployment are not the focus of our tests.

We tested the performance of recipients resolution on a system with Intel i7 2.6GHz processor with 768 MB of RAM. Assuming that the system is supporting 10K users, we generated random messages that are encrypted with access policies of 10 attributes and operators (”AND”/”OR”). The results indicate that the system is able to resolve on average 1404 recipients in around 514ms.

We also tested the performance of cryptographic operations on a system with Intel i7 2.6GHz processor with 8 GB of RAM. We generated random messages of different sizes (1-64 MB) with access policies of 10 attributes and ”AND” operators. The messages were encrypted with an average time of around 556ms, transformed around 144ms and decrypted around 7ms. To test for possible worst case situation, we generated messages of different sizes (1-64 MB) with access policies of 50 attributes and ”AND” operators. The messages were encrypted with an average time of around 2.8s, transformed around 728ms and decrypted around 7ms. Table 1 summarized the preliminary results.

Table 1. Performance of Cryptographic Operations

No of Attributes	Message Size	Operators	Average Time		
			Encryption	Transformation	Decryption
10	1-64 MB	AND	556ms	144ms	7ms
50	1-64 MB	AND	2.8s	728ms	7ms

6. Related Work

Attribute-Based Encryption - ABE schemes can be broadly categorized into two categories namely, Key Policy ABE (KP-ABE) and Ciphertext- Policy ABE (CP-ABE). In KP-ABE schemes, ciphertexts are associated with attributes while decryption keys are associated with access structures. However, in CP-ABE schemes, the association is reversed where ciphertexts are associated with access structures while decryption keys are associated with attributes.

The concept of ABE was first proposed by Sahai and Waters [13] as a type of Fuzzy Identity Based Encryption (Fuzzy-IBE) scheme. Subsequently, Goyal et al. [8] introduced the notion of KP-ABE and CP-ABE and provided the construction for KP-ABE scheme. Soon after, Bethencourt et al. [3] provided the construction for CP-ABE. Initial work on ABE schemes focus on monotonic access structures, but soon, Ostrovsky et al. [10] proposed an ABE scheme with non-monotonic policies to represent negative constraints. To meet the need for complex access policy, several efforts focus on providing more fine-grained access control. For instance, Bobba et al. [5] extended CP-ABE by representing attributes as a recursive set structure and named the scheme Ciphertext-policy Attribute-Set-Based Encryption (CP-ASBE or ASBE) while Wang et al. [16] introduced Hierarchical Attribute-Based Encryption (HABE) by combining Hierarchical Identity-based Encryption (HIBE) with CP-ABE. Thereafter, Wan et al. [15] introduced Hierar-

chical Attribute-Set-Based Encryption (HASBE) by extending ASBE with users organized into a hierarchical structure to provide a more scalable, flexible and fine-grained control of ABE.

Revocable ABE - Attrapadung and Imai [1] proposed two types of revocation techniques, namely direct revocation and indirect revocation. Direct revocation is performed by sender specifying the revocation list while indirect revocation is enforced by the key authority providing key updates to non-revoked users. Attrapadung and Imai [2] also presented a hybrid revocable attribute encryption (HR-ABE) by combining the two techniques and data owner may select either technique. Boldyreva et al. [6] proposed an efficient revocable KP-ABE scheme by improving the key updates. Sahai et al. [12] proposed a revocable scheme by revoking stored data together with key updates. Yang et al. [18] proposed a revocable ABE scheme by denying user decryption capability via a cloud server.

Secure Messaging - Messaging protocols and solutions secure messages with a wide range of techniques [14]. In particular, ABE is an upcoming and promising technique that can achieve secure messaging. Bobba et al. [4] proposed Attribute-Based Messaging (ABM) by integrating ABE with Mail Transfer Agent (MTA). Weber et al. [17] proposed MundoMessage which integrated ABE and Location-Based Encryption (LBE) into emergency communication. In healthcare, Picazo-Sanchez et al. [11] incorporated ABE into messaging protocol for monitoring and managing the medical wireless body area networks (WBANs). Unger et al. [14] evaluated existing messaging systems and established that messaging systems face three key challenges: trust establishment, conversation privacy and transport privacy. They also proposed a framework to evaluate the properties of messaging systems.

7. Conclusion

In this paper, we proposed a system architecture for an Attribute-Based Secure Messaging System with Outsourced Decryption (ABSM-OD) that provides end-to-end message security in the cloud. ABSM-OD is built upon ABE-OD scheme [9] and achieves three key features namely fine-grained access control, outsourced decryption and effective user revocation. Our prototype implementation demonstrates the feasibility of the architecture with reasonable performance. From our preliminary results, the system is able to determine recipients from the access polices of ciphertexts within an acceptable amount of time. Also, the decryption of transformed ciphertexts are more manageable for mobile users. Hence, ABSM-OD can provide scalable and secure messaging within the public cloud.

Acknowledgments

This research work is supported by the Singapore National Research Foundation under the NCR Award Number NRF2014NCR-NCR001-012.

References

[1] N. Attrapadung and H. Imai. *Attribute-Based Encryption Supporting Direct/Indirect Revocation Modes*, pages 278–300. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[2] N. Attrapadung and H. Imai. *Conjunctive Broadcast and Attribute-Based Encryption*, pages 248–265. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[3] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, SP '07, pages 321–334, Washington, DC, USA, 2007. IEEE Computer Society.

[4] R. Bobba, O. Fatemieh, F. Khan, A. Khan, C. A. Gunter, H. Khurana, and M. Prabhakaran. Attribute-based messaging: Access control and confidentiality. *ACM Trans. Inf. Syst. Secur.*, 13(4):31:1–31:35, Dec. 2010.

[5] R. Bobba, H. Khurana, and M. Prabhakaran. Attribute-sets: A practically motivated enhancement to attribute-based encryption. In *Proceedings of the 14th European Conference on Research in Computer Security*, ESORICS'09, pages 587–604, Berlin, Heidelberg, 2009. Springer-Verlag.

[6] A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, pages 417–426, New York, NY, USA, 2008. ACM.

[7] T. Frosch, C. Mainka, C. Bader, F. Bergsma, T. Holz, et al. How secure is textsecure? In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 457–472. IEEE, 2016.

[8] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. ACM, 2006.

[9] M. Green, S. Hohenberger, and B. Waters. Outsourcing the decryption of abe ciphertexts. In *Proceedings of the 20th USENIX Conference on Security*, SEC'11, pages 34–34, Berkeley, CA, USA, 2011. USENIX Association.

[10] R. Ostrovsky, A. Sahai, and B. Waters. Attribute-based encryption with non-monotonic access structures. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS '07, pages 195–203, New York, NY, USA, 2007. ACM.

[11] P. Picazo-Sanchez, J. E. Tapiador, P. Peris-Lopez, and G. Suarez-Tangil. Secure publish-subscribe protocols for heterogeneous medical wireless body area networks. *Sensors*, 14(12):22619, 2014.

[12] A. Sahai, H. Seyalioglu, and B. Waters. *Dynamic Credentials and Ciphertext Delegation for Attribute-Based Encryption*, pages 199–217. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[13] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'05, pages 457–473, Berlin, Heidelberg, 2005. Springer-Verlag.

[14] N. Unger, S. Dechand, J. Bonneau, S. Fahl, H. Perl, I. Goldberg, and M. Smith. Sok: Secure messaging. In *2015 IEEE Symposium on Security and Privacy*, pages 232–249, May 2015.

[15] Z. Wan, J. Liu, and R. H. Deng. 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, April 2012.

[16] G. Wang, Q. Liu, J. Wu, and M. Guo. Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers. *Comput. Secur.*, 30(5):320–331, July 2011.

[17] S. G. Weber, Y. Kalev, S. Ries, and M. Mühlhäuser. Mundomessage: Enabling trustworthy ubiquitous emergency communication. In *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication*, ICUIMC '11, pages 29:1–29:10, New York, NY, USA, 2011. ACM.

[18] Y. Yang, X. Ding, H. Lu, Z. Wan, and J. Zhou. *Achieving Revocable Fine-Grained Cryptographic Access Control over Cloud Data*, pages 293–308. Springer International Publishing, Cham, 2015.