Towards Privacy Preserving Access Control in the Cloud

Mohamed Nabeel, Elisa Bertino
Purdue University
305 N. University Street
West Lafayette, IN 47907
Email: {nabeel, bertino}@cs.purdue.edu

Murat Kantarcioglu, Bhavani Thuraisingham
The University of Texas at Dallas
800 W. Campbell Road
Richardson, TX 75080
Email: {muratk, bhavani.thuraisingham}@utdallas.edu

***Abstract*—It is very costly and cumbersome to manage database systems in-house especially for small or medium organizations. Data-as-a-Service (DaaS) hosted in the cloud provides an attractive solution, which is flexible, reliable, easy and economical to operate, for such organizations. However security and privacy issues concerning the storage of the data in the cloud and access via the Internet have been major concerns for many organizations. The data and the human resources are the life blood of any organization. Hence, they should be strongly protected. In this paper, we identify the challenges in securing DaaS model and propose a system called CloudMask that lays the foundation for organizations to enjoy all the benefits of hosting their data in the cloud while at the same time supporting fine-grained and flexible access control for shared data hosted in the cloud.**

***Index Terms*—Privacy, Access Control, Storage as a Service**

I. Introduction

Many benefits, including on-demand provisioning that enables organizations to grow efficiently and in a cost effective manner, have been the force driving many organizations to move into the cloud. Storage as a service (SaaS) and Data as a service (DaaS) are emerging cloud services by which organizations can seamlessly store data in the cloud and retrieve them based on access control policies (ACPs) that cover legal requirements and organizational policies. While SaaS provides a virtual storage, DaaS provides a higher level interface to store and query data on a data structure. Amazon S3 and Microsoft Azure storage service are two such popular services currently available.

While cloud data services provide many benefits, data privacy and security issues have been major concerns. Data stored in the cloud often encode sensitive information and should be protected as mandated by various organizational policies and legal regulations. A commonly adopted approach to address security and privacy is to encrypt the data before uploading them to the cloud. Encryption alone however is not sufficient as often organizations have to enforce fine-grained access control on the data. Such control is often based on information such as role of data users in the organization, projects on which users are working and so forth. Therefore, an important requirement is to support fine-grained access control, based on policies specified in a expressive access control language, over encrypted data hosted in the cloud.

In particular, an expressive access control model, such as XACML, allows one to specify ACPs on protected objects in terms of the properties of subjects, referred to as *identity attributes*. The email address, the role a user plays in her organization, the age and the location a user accesses from are a few examples of such identity attributes. The identity attributes that subjects should possess in order to access protected objects are referred to as *conditions*. Such an attribute based access control model is crucial in order to support fine-grained access control policies to data. The electronic health record (EHR) in Figure 1 shows an example of attribute based access control policies for different portions of the record. For example, a user who plays the role of data analyst can only access lab reports.

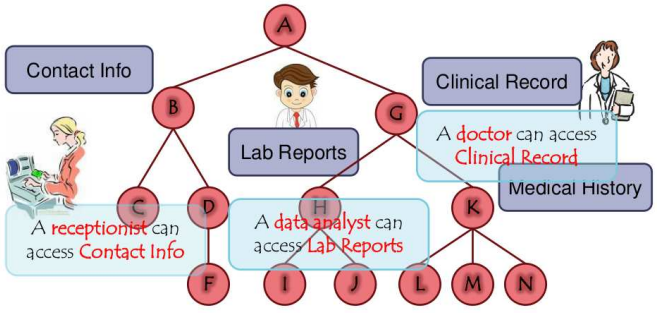

Fig. 1. A simplified electronic record

A crucial issue in this context is that the identity attributes in the access control policy conditions often encode privacy-sensitive information. The privacy of the users is thus not protected if their identity attributes are not protected. Privacy, both individual as well as organizational, is considered a key requirement in all solutions, including cloud services, for digital identity management. Further, as insider threats [27] are one of the major sources of data theft and privacy breaches, identity attributes must be strongly protected even from accesses within the organization. With cloud computing initiatives the scope of insider threats is no longer limited to the organizational perimeter. Therefore, protecting the identity attributes of the users while enforcing attribute-based access

COLLABORATECOM 2011, October 15-18, Orlando, United States

DOI 10.4108/icst.collaboratecom.2011.247061

control both within the organization and in the cloud is an important requirement.

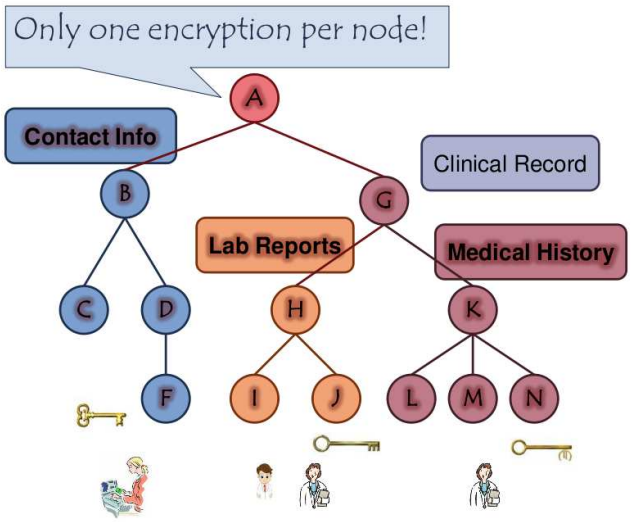

Fig. 2. A simplified EHR with sections having the same ACP are encrypted with the same symmetric key.

In order to utilize minimum storage and bandwidth when supporting fine-grained access control over encrypted data, it is preferable to encrypt each data item only once. A simple approach to do so is to encrypt the data items covered by different ACPs with different keys and give keys to each user depending on which ACPs they satisfy. Figure 2 shows a simplified EHR with such an encryption based access control. For example, the users playing the role of lab analyst or doctor share the same key for the lab reports.

Such an approach, however, has some major drawbacks with respect to privacy and key management. As shown in Figure 3, the users' privacy is not preserved since the users are required to reveal their identity attributes to obtain the keys. Further, the key management is not scalable since every time the policies applicable to users change due to joining of new users or leaving of existing users, the keys need to be reissued to existing users in order to provide forward and backward secrecy to the data. The goal of this work is to develop a system, called *CloudMask*, to overcome these shortcomings while enjoying the benefits of the cloud data service models. CloudMask is an extension of our previous work [29], [24].

In CloudMask, encryption is performed like in the simple approach mentioned above. However, unlike such simple approach, users are not given the symmetric keys directly. Instead, users are given one or more secrets that they can use to derive the key on the fly in conjunction with some variable public information. Such public information is stored on the cloud and can be directly accessed by all users. CloudMask provides forward and backward secrecy without affecting the secrets given to existing users. In other words, CloudMask handles rekey operations efficiently since it does not directly share the same key with multiple users.

Fig. 3. A user obtaining a key when no privacy issues are considered.

The rest of the paper is organized as follows. Section II gives an overview of the architecture of CloudMask. Section III provides an overview of the two key building blocks used in CloudMask. Section IV describes the main phases in key management. Section V discusses possible extensions to our Cloud Mask. Section VI reviews selected work related to ours and Section VII concludes the paper.

II. CloudMask Architecture

We assume that the organizational data are grouped into *documents*. Each data item in a document is called a *subdocument*. CloudMask consists of the following entities:

- *Document Manager* (DM) is usually an in-house entity that manages subscriptions and performs policy based encryption of documents. Some parts of the computations performed by the DM can be moved to a cloud infrastructure, such as Amazon EC2. However, we need to exercise care in doing so since we need to make sure that the actual keys are not exposed to the cloud.
- *Cloud Data Service* (CDS) is a third party cloud service hosting the encrypted documents. The CDS may work under the SaaS or DaaS model.
- *Users* (Usrs) are the employees of the organization. They register with the DM and retrieve documents from the CDS.
- *Identity Providers* (IdPs) are independent entities that issue certified identity tokens, i.e., commitments [1] of identity attributes, to Usrs.

Consider the following scenario. A hospital wants to move its EHRs [16] (documents) to a CDS. Usrs are the hospital employees playing different roles such as receptionist, cashier, doctor, nurse, pharmacist, system administrator, and non-employees such as patients. A cashier, for example, need not have access to the data in EHRs except for the billing information (subdocument) in them, while a doctor or a nurse need not have access to billing information. The typical identity attributes used by the Usrs, such as role, location and position, can be used as good contextual information that could be linked with other publicly available information in

[1] A commitment is a cryptographic primitive which allows a user to commit to a value while keeping it hidden and preserving the user's ability to reveal the committed value later. In other words, it unconditionally hides a value while computationally binding the value to the user.

order to learn sensitive information about individuals, leading to privacy violations. There have been many incidents where hospital employees steal coworkers' identity attributes to impersonate them and carry out highly damaging insider attacks [27]. Our system is a promising step forward to minimize such identity theft incidents as well as new incidents in the cloud [15] since the DM and CDS do not learn identity attributes of Usrs.

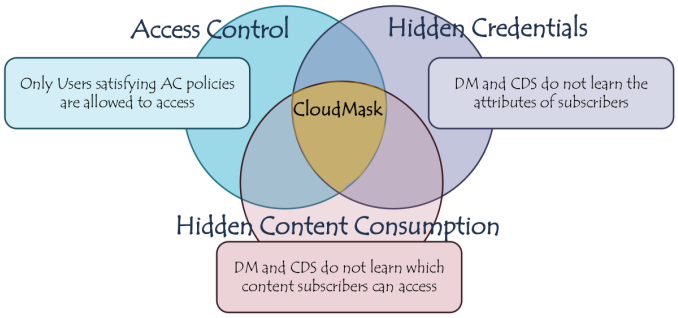

Fig. 4. The privacy and security requirements that CloudMask satisfies.

In summary, as shown in Figure 4, CloudMask satisfies the following privacy and security requirements:

- The DM and the CDS do not learn the identity attributes of Usrs.
- The CDS does not learn the content of the subdocuments.
- Attribute-based access control is enforced for the subdocuments.
- The CDS provides a mechanism to restrict the access to subdocuments only to authorized Usrs without learning their identity attributes.

III. Building Blocks

CloudMask is based two key building blocks: Oblivious Commitment Based Envelope (OCBE) protocols and Broadcast Group Key Management (BGKM) schemes.

A. OCBE Protocols

OCBE protocols, proposed by Li and Li [20], provide a way to obliviously deliver a message to the Usrs who satisfy certain conditions. The DM and Usrs engage in OCBE protocols for the Usrs to obtain secrets for the identity tokens, expressed as commitments, they have. As shown in Figure 5, for a given condition, a Usr sends her identity token, obtained from an IdP, to the DM. The DM, in turn, sends the Usr an envelope, that is, an encrypted message, containing a secret. The Usr can open (i.e., decrypt) the envelope only if she knows the committed value in her identity token. In other words, the Usr can derive the symmetric key only if her identity token (e.g., commitment of "age" = 25) satisfies the condition (e.g., "age" $\geq$ 21). An OCBE protocol guarantees the following properties:

- The DM does not learn the credentials of the Usrs.
- A Usr can open the envelope only if her credential satisfies the condition.

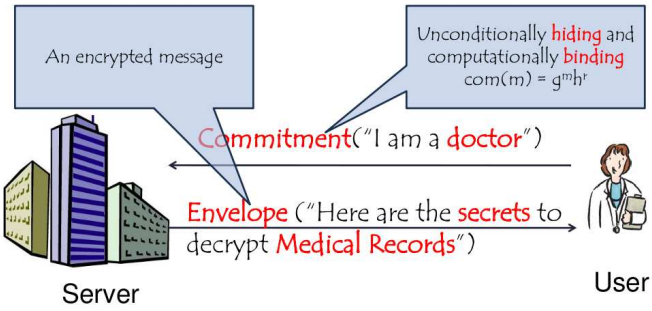

Fig. 5. Shows the high-level interactions between the server and a Usr. The underlying protocol is based on the zero knowledge proof of knowledge where the Usr proves to the server that it can open the committed identity attribute without showing the actual value. If the server is satisfied, it uses the Usr's identity token to create a symmetric key and then an envelope.

B. BGKM Schemes

Group Key Management (GKM) is widely used to securely distribute a message to a group of Usrs. In such a setting, confidentiality is the key. The main idea is that the Usrs in the group share a symmetric key K, called the group key, and whenever they want to communicate a message securely, the message is encrypted with K and broadcast to all the Usrs in the group. Since K is known only to the Usrs in the group, only they can decrypt and obtain the message. When the group dynamics changes, i.e., a new Usr joins or an existing Usr leaves the group, a new group key must be generated and redistributed in a secure way to all current group Usrs, so that a new member of the group cannot recover earlier transmitted messages (backward secrecy), and a Usr who has left the group cannot learn anything from future communications in the group (forward secrecy). This process is called *rekeying*. A traditional GKM scheme requires to setup private communication channels with all the Usrs in the group to update the group key when the group dynamic changes. Such an approach is not desirable if there are frequent leaves/joins with many Usrs in the group. Further, it does not preserve the privacy of the Usrs as the direct user-key relationship should be maintained. Broadcast GKM (BGKM) schemes overcome these issues. As shown in Figure 6, instead of distributing keys to users, each user obtains one or more secrets which they can combine with some variable public information to obtain the group key. Whenever the group dynamics changes, only the public information needs to be changed and no private communication channel needs to be setup.

A few BGKM schemes have been proposed [12], [11], [32]. In this work, we focus on the first provably secure BGKM scheme ACV-BGKM [29]. ACV-BGKM is based on the really simple idea of matrix null spaces. As show in Figure 7 (a), the key generation server, the DM, creates a matrix $\mathbf{A}$ of size $n \times m$ ($m > n$) where each row is constructed using the secret(s) given to each Usr in the group. Since the secrets of each Usr are unique, $\mathbf{A}$ is a full rank matrix. The DM then computes (Figure 7 (b)) the null space of $\mathbf{A}$, $\mathbf{B}$ of size $t \times m$ ($t = m - n$)

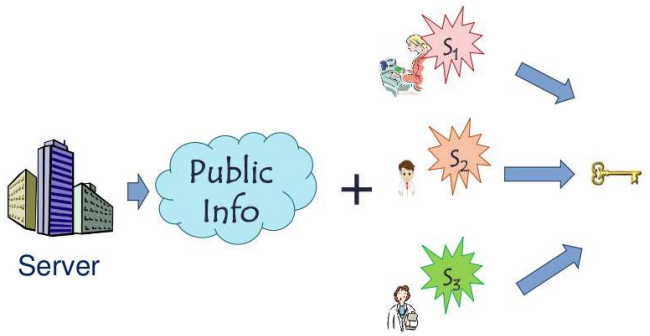

Fig. 6. Shows the concept behind BGKM. Notice that users are not given the key directly. Instead, they are given some secrets which can be combined with public information to obtain the key.

and choose a random vector from **B** and embeds the group key k in it. We call this final vector Access Control Vector (ACV). As shown in Figure 7 (c), a valid Usr can construct a vector in the row space of **A** using her secret(s). We call this vector Key Extraction Vector (KEV). The inner product of the ACV and the KEV gives the group key k. In order to change k, only the public ACV should be changed while the user secrets remain unchanged. At high-level, removing a row from **A** prevents an existing Usr from obtaining the key and adding a new row to **A** allows a new Usr to obtain the key.

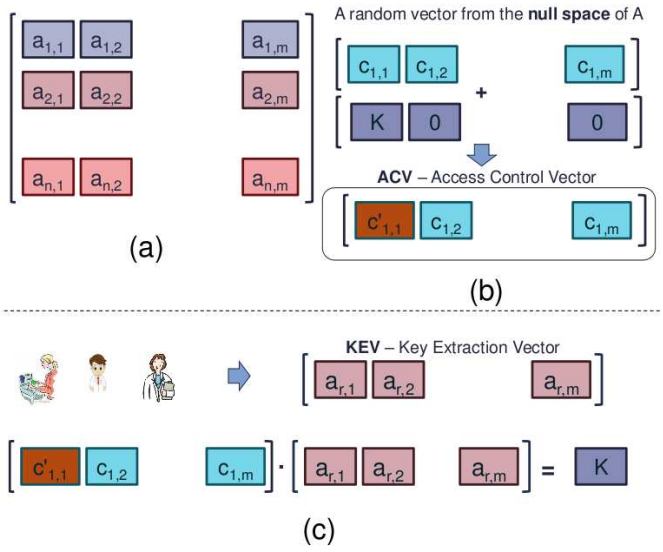

Fig. 7. Shows the main steps of the ACV-BGKM scheme.

Now we provide technical details of BGKM in general and the above described ACV-BGKM scheme in particular. In general, a BGKM scheme consists of the following five algorithms: **Setup**, **SecGen**, **KeyGen**, **KeyDer**, and **Update**.

- **Setup(ℓ)**: It initializes the BGKM scheme using a security parameter ℓ. It also initializes the set of used secrets **S**, the secret space $\mathcal{SS}$, and the key space $\mathcal{KS}$.
- **SecGen()**: It picks a random bit string $s \notin$ **S** uniformly at random from $\mathcal{SS}$, adds s to **S** and outputs s.
- **KeyGen(S)**: It picks a group key k uniformly at random from $\mathcal{KS}$ and outputs the public information tuple PI computed from the secrets in **S** and the group key k.
- **KeyDer(s, PI)**: It takes the user's secret s and the public information PI to output the group key. The derived group key is equal to k if and only if $s \in$ **S**.
- **Update(S)**: Whenever the set **S** changes, a new group key k' is generated. Depending on the construction, it either executes the **KeyGen** algorithm again or incrementally updates the output of the last **KeyGen** algorithm.

Using the above abstract algorithms, we provide an overview of the construction of the ACV-BGKM scheme under a client-server architecture. The ACV-BGKM scheme satisfies the requirements of *minimal trust*, *key indistinguishability*, *key independence*, *forward secrecy*, *backward secrecy* and *collusion resistance* [11]. The ACV-BGKM algorithms are executed by a trusted key server DM and a group of users $\mathsf{Usr}_i, i = 1, 2, \ldots, n$.

- **Setup(ℓ)**: DM initializes the following parameters: an ℓ-bit prime number q, the maximum group size N ($\geq n$ and N is usually set to $n+1$), a cryptographic hash function $H(\cdot) : \{0,1\}^* \rightarrow \mathbb{F}_q$, where $\mathbb{F}_q$ is a finite field with q elements, the keyspace $\mathcal{KS} = \mathbb{F}_q$, the secret space $\mathcal{SS} = \{0,1\}^\ell$ and the set of issued secrets **S** $= \emptyset$.
- **SecGen()**: DM chooses the secret $s_i \in \mathcal{SS}$ uniformly at random for Usr_i such that $s_i \notin$ **S**, adds s_i to **S** and finally outputs s_i.
- **KeyGen(S)**: DM picks a random $k \in \mathcal{KS}$ as the group key. DM chooses N random bit strings $z_1, z_2, \ldots, z_N \in \{0,1\}^\ell$. DM creates an $n \times (N+1)$ $\mathbb{F}_q$-matrix

$$A = \begin{pmatrix} 1 & a_{1,1} & a_{1,2} & \ldots & a_{1,N} \\ 1 & a_{2,1} & a_{2,2} & \ldots & a_{2,N} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_{n,1} & a_{n,2} & \ldots & a_{n,N} \end{pmatrix},$$

where

$$a_{i,j} = \begin{cases} 1 & \text{if } j = 0 \\ H(s_i || z_j) & \text{if } 1 \leq i \leq n, 1 \leq j \leq N, s_i \in \mathcal{S} \end{cases}$$

DM then solves for a nonzero $(N+1)$-dimensional column $\mathbb{F}_q$-vector Y such that $AY = 0$. Note that such a nonzero Y vector always exists as the nullspace of matrix A is nontrivial by construction. Here we require that DM chooses Y from the nullspace of A uniformly at random. DM constructs an $(N+1)$-dimensional $\mathbb{F}_q$-vector

$$ACV = k \cdot e_1^T + Y,$$

where $e_1 = (1, 0, \ldots, 0)$ is a standard basis vector of $\mathbb{F}_q^{N+1}$, v^T denotes the transpose of vector v, and k is the chosen group key. The ACV vector controls the access to the group key k and is called an *access control vector*. DM lets

$$PI = \langle ACV, (z_1, z_2, \ldots, z_N) \rangle,$$

and outputs public PI and private k.

- **KeyDer(s_i, PI)**: Using its secret s_i and the public information tuple PI, Usr_i computes $a_{i,j}, 1 \leq j \leq N$, as in the above formula and sets an $(N+1)$-dimensional row $\mathbb{F}_q$-vector

$$v_i = (1, a_{i,1}, a_{i,2}, \ldots, a_{i,N}).$$

v_i is called a Key Extraction Vector (KEV) and corresponds to a unique row in the access control matrix A. Usr_i derives the key k' from the inner product of v_i and ACV:

$$k' = v_i \cdot ACV.$$

The derived key k' is equal to the actual group key k if and only if s_i is a valid secret used in the computation of PI, i.e., $s_i \in \mathbf{S}$.

- **Update(S)**: It runs the **KeyGen(S)** algorithm and outputs the new public information PI' and the new group key k'.

The above construction becomes impractical with large number of users since the complexity of the matrix and the public information is $O(n)$. In our technical report [23], we propose using subset-cover techniques [25], [19] with BGKM to make the complexity sublinear in n. The high-level idea is that instead of giving only one secret, Usrs are given multiple secrets which overlap with some of the secrets given to other Usrs. When a Usr is revoked, one or more secrets become invalid. When generating a new group key k, instead of selecting one secret per Usr, we select only the minimal subset of secrets that covers all the group members. Some subset-cover techniques reduce the number of secrets required to build the access control matrix to $\log n$ which in turn vastly improves all the algorithms of the ACV-BGKM scheme.

IV. Main Phases in Key Management

CloudMask undergoes the following three distinct phases each of which works independently:

- Identity token issuance
- Identity token registration
- Document publication and authorization

A. Identity Token Issuance

IdPs issue identity tokens ($\mathcal{IT}$s) to Usrs. As shown in Figure 8, a Usr shows proofs of identity attributes to an IdP. If the IdP is convinced that identity attributes belong to the Usr, it issues a identity token for each such identity attribute. We assume that every IdP issues these tokens in a uniform format $\mathcal{IT} = (\mathsf{nym}, \mathsf{id\text{-}tag}, c, \sigma)$, where nym is a pseudonym for uniquely identifying a Usr in CloudMask, id-tag is the tag of the identity attribute under consideration, c a Pedersen commitment[2] for the identity attribute value, and σ is the IdP's digital signature for nym, id-tag and c. For example, an identity token that a Usr, identified by "eid-1515", receives for the identity attribute tag "age", looks like $\mathcal{IT}$ = (eid-1515, age, 543243243556, 4322348998254219).

[2]A Pedersen commitment is a type of cryptographic commitment.

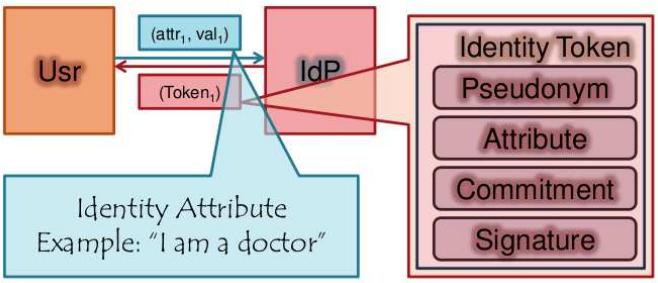

Fig. 8. Shows the interactions between the Usr and the IdP during the token issuance phase.

B. Identity Token Registration

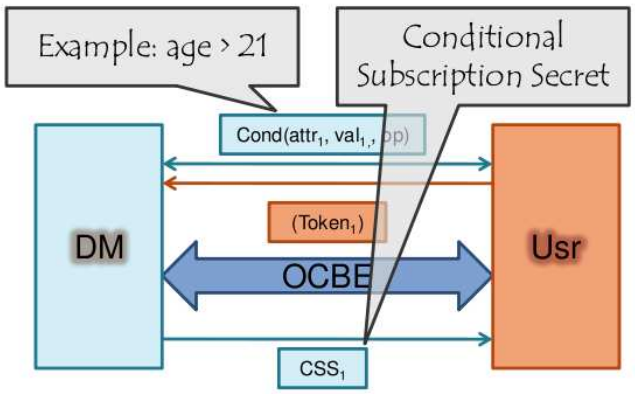

Fig. 9. Shows the interactions between the Usr and the DM during the token registration phase.

In this phase, Usrs use the identity tokens obtained from IdPs. Usrs register with the DM in order to access content, encoded as subdocuments, from the CDS. During the registration, Usrs first retrieve the DM's ACPs, each of which is of the form (*subject*, policy-id), and the *policy configurations*, each of which is a disjunction of some ACPs. The subject of each access control policy, identified by the policy-id, is a conjunction of conditions. A condition has the form $\mathsf{cond} = (\mathsf{id\text{-}tag} \quad \mathsf{op} \quad l)$, where op is a comparison operator such as $=$, $<$, $>$, $\leq$, $\geq$, $\neq$, and l is a value that can be assumed by the identity attribute referred to by the tag id-tag. A policy is satisfied if and only if all the conditions in that policy are satisfied. An example policy with two conditions is ("level $\geq$ 58" $\wedge$ "role $=$ nurse"). A policy configuration is associated with a subdocument. In order to access a subdocument, a Usr should satisfy at least one policy in the policy configuration of the subdocument. After determining which policy configurations are to be satisfied, as shown in Figure 9, the Usr presents its required identity tokens through OCBE protocols. Note that during this process the DM learns neither the attributes of the Usr nor the satisfiability of

the corresponding condition. For each identity token received from the Usr during registration, the DM sends an encrypted *conditional subscription secret* (CSS). CSSs are used by Usrs to derive the decryption keys for the subdocuments for which they satisfy the policy configuration and are managed by the BGKM scheme. Note that Usrs can obtain the CSS from the encrypted response message if and only if they possess a valid identity token.

C. Document Publication and Authorization

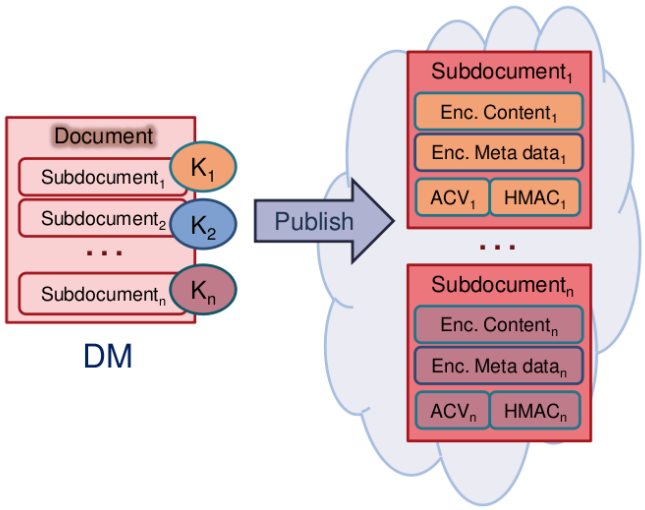

Fig. 10. Shows the DM publishing a sample document in the CDS.

As shown in Figure 10, the DM encrypts the subdocuments having the same policy configuration with the same symmetric key K and publishes the subdocuments together with certain additional information. Each encrypted subdocument has the following information:

- Subdocument identifier
- Encrypted content of the subdocument
- Encrypted meta-data identifying some key fields in the subdocument
- ACV
- Hash-based message authentication code of the ACV using the key K, HMAC(K, ACV)

The ACV is generated as outlined in Section III-B. The Usrs are aware of which subdocuments they satisfy. However, the CDS sees these encrypted subdocuments simply as objects and cannot differentiate one subdocument from another. Anonymous access is provided to the identifier and the ACV of each subdocument. In other words, anyone can access these information. One could allow anyone to access all the subdocuments since they are encrypted. However organizations could suffer from such a policy. Usually a CDS charges based on the bandwidth utilization. Therefore the utilization of bandwidth is controlled in two ways. Firstly, the HMAC in each subdocument is used as an authorization code. The idea is that, an authorized Usr is able to construct the HMAC using its private CSS values and the public ACV. A Usr is allowed to access the encrypted meta-data and the content only if the constructed HMAC matches the HMAC attached to the encrypted subdocument. An advantage of such a simple authorization method is that the condition can be specified in a simple access control list (ACL) which many commercial CDS providers already support. Secondly, the Usrs first download only the smaller encrypted meta-data to decide if they want to download the typically large encrypted content.

If a key needs to be updated due to the changes in the user dynamics (i.e., a new Usr satisfying a policy configuration or an existing Usr ceasing to satisfy a policy configuration), the DM generates a new ACV, re-encrypt only the affected subdocuments and publishes them in the cloud without requiring to update any CSS values issued to existing Usrs. That way keys are managed transparently to the existing Usrs, while, at the same time, assuring forward and backward secrecy of the content.

V. Discussion

In this section, we discuss some useful extensions to CloudMask.

A. Audit log generation for regulatory compliance

Due to various regulatory requirements, organizations need to maintain audit logs related to data access. For example, in USA, health care security and privacy regulations require organizations to keep audit logs that capture who did what to which health record, when, and on which system. In CloudMask, such logging could be easily implemented in different parts of the system. For example, during our OCBE protocol execution, the DM can keep the logs of user credentials that are used to get secrets to generate encryption keys for accessing subdocuments. Since user credentials are never revealed to the DM, we need a separate technique to keep track of user credentials. One possible solution is to use a conditional key escrow which reveals the credentials of a user only when a certain condition is met. Such an approach keeps the credentials of honest users secret from the DM while only revealing the credentials of misbehaving users. The logs kept by the DM could be used later on to identify who may have accessed any given subdocuments.

One issue with keeping logs with the DM is that a user who executed the OCBE protocol later on may claim that even though she got the secrets to generate encryption keys, she never used keys to access data. To counter against such issues, another auditing layer could be added to the CDS. For example, each user may be provided pseudonyms and public keys signed by CloudMask. Using such signed pseudonyms and public keys, users can prove their identity to the CDS before retrieving any document. The CDS can easily store logs that are capturing the pseudonyms of the users who are downloading any given document. Given the logs kept by the CDS and DM, we can combine them to precisely learn users who downloaded documents and encryption keys together. We

leave the integration of such audit logs to our system as a future work.

B. Improving the query efficiency

There is a trade-off between the amount of security/privacy and query performance when the data is encrypted and stored in the cloud. Data encryption makes querying and retrieving selected files a challenging task. A naive solution is to let the Usrs download the complete encrypted data set, decrypt it and filter the data they want to access based on the keywords or phrases. Such a solution is not acceptable since it requires high network bandwidth and the Usrs require high capacity storage and processing capabilities as they are required to decrypt all the documents to obtain the document(s) of interest to them. Ideally, the system should be able to let the Usrs download only those authorized encrypted documents they are interested in. One approach to address such issue is to support keyword based search. Since keywords can reveal sensitive information about Usrs, they should be encrypted as well. While traditional search over encrypted data techniques [30], [5], [17], [7] can be utilized to provide a basic keyword based querying capability, it is still an open problem to support authenticated querying beyond simple presence or absence of a set of keywords expressed as a Boolean formula. We plan to support authenticated querying in CloudMask.

VI. Related Work

Searchable Encryption: Search in encrypted data is a privacy-preserving technique used in the *outsourced storage model* where a user's data is stored on a third-party server and encrypted using the user's public key. The user can use a query in the form of an encrypted token to retrieve relevant data from the server, whereas the server does not learn any more information about the query other than whether the returned data matches the search criteria. There have been efforts to support simple keyword queries [30], [5], conjunctive keyword queries [17] and more recently complex ones involving conjunctive, subset and range queries [7]. The primary focus of such work is to protect the confidentiality of the published data from the third-party servers. Issues related to fine-grained access control (FGAC), such as key management, are not considered and the servers are trusted to preserve the privacy of the users who query the encrypted content. Further, these approaches are not able to support general monotonic access control policies. There have been some recent attempts to provide keyword based searches in the cloud [31], [10], [21]. While these approaches provide different capabilities, such as fuzzy keyword search [21], ranked keyword search [31] and multi-keyword search [10], they do not provide authenticated search capabilities and do not address key management issues.
Attribute Based Encryption: The concept of attribute-based encryption (ABE) has been introduced by Sahai and Waters [28]. ABE can be considered as a generalization of identity based encryption [6], [13] (IBE), where the encryption is based on some identity. Thus, ABE is more expressive than IBE. In an ABE system, the plaintext is encrypted with a set of attributes. The key generation server, which possesses the master key, issues different private keys to users after authenticating the attributes they possess. Thus, these private keys are associated with the set of attributes each user possesses. In its basic form, a user can decrypt a ciphertext if and only if there is a match between the attributes of the ciphertext and the user's key.

The initial ABE system is limited only to threshold policies where there should be at least k out of n attributes common between the attributes used to encrypt the plaintext and the attributes users possess. Pirretti et al. [26] gave an implementation of such a threshold ABE system using a variant of the Sahai-Waters Large Universe construction [28]. Since the initial threshold scheme, a few variants have been introduced to provide more expressive ABE systems. Goyal et al. [18] introduced the idea of key-policy ABE (KP-ABE) systems and Bethencourt et al. [4] introduced the idea of ciphertext-policy ABE (CP-ABE) systems. Even though these constructs are expressive and provably secure, it is hard to support group management, especially to provide forward security when a user leaves the group (i.e. attribute revocation) and to provide backward security when a new user joins the group. Some of the above schemes suggest using an expiration attribute along with other attributes. However, such a solution is not suitable for a dynamic group where joins and departures are frequent.
Fine-grained Access Control: Fine-grained access control (FGAC) allows one to enforce selective access to the content based on expressive policy specifications. Research in FGAC can be categorized into two dissemination models: *push-based* and *pull-based* models. In a push-based system, content publishers push the content to users either by broadcasting or making the content available in a public location. In a pull-based system, every time users want to access some content, they login to the content provider and retrieve based on the access control policies. Our work focuses on the pull based model, but the techniques introduced can be used to construct push-based systems supporting FGAC.

Under the push-based model, the database and security communities have carried out research concerning techniques for the selective dissemination of documents based on access control policies [3], [22]. In all such work, subdocuments are encrypted with different keys, which are provided to users at the registration phase, and broadcast the encrypted subdocuments to all users. However, such approaches require all [3] or some [22] keys be distributed in advance during user registration phase. This requirement makes it difficult to assure forward and backward key secrecy when user groups are dynamic with frequent join and leave operations. Further, the rekey process is not transparent, thus shifting the burden of acquiring new keys on existing users when others leave or join. In contrast, our approach makes rekey transparent to users by not distributing actual keys during the registration phase. Another distinction is that all these approaches focus on achieving confidentiality of the content and privacy of the users who access the content is not considered. In contrast, our goal is not only to provide confidentiality but also to preserve

the privacy of users who access the documents.

Under the pull-based model, the content publisher is required to be online in order to access the content. There has been some recent research efforts [14], [8] to construct privacy preserving access control systems by combining oblivious transfer [9], [1] and anonymous credentials [2]. The goal of such work is similar to ours but we identify the following limitations. Each transfer protocol allows one to access only one record from the database, whereas our approach does not have any limitation on the the number of records that can be accessed at once since we separate the access control from the authorization. Another drawback is that the size of the encrypted database is not constant with respect to the original database size. Redundant encryption of the same record is required to support policies involving disjunctions. However, our approach encrypts each data item only once as we have made the encryption independent of the policies. Further, such approaches are not designed to support privacy preserving content based access control.

VII. Conclusion

Current trends in cloud computing and associated services are further pushing publishing functions to third-party providers to achieve flexibility and economies of scale. However, recent surveys have found that one of the key resistance factors for organizations to move to the cloud is represented by data privacy and security concerns. We believe that CloudMask is a promising solution to address privacy and security concerns in the context of attribute based access control of organizational data over a cloud data service.

We have implemented and demonstrated a preliminary system that supports most of the functionality described in Section IV except the querying facility [24]. We plan to adapt the system to build CloudMask and also extend it to support the capabilities discussed in Section V.

Acknowledgement

This material is based upon work supported by the Air Force Office of Scientific Research under Awards No. FA9550-08-1-0260, FA9550-09-0468 and FA9550-08-1-0265. We thank Dr. Robert Herklotz for his support of our work.

References

[1] W. Aiello, Y. Ishai, and O. Reingold. Priced oblivious transfer: How to sell digital goods. In *EUROCRYPT '01: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, pages 119–135, London, UK, 2001. Springer-Verlag.

[2] M. H. Au, W. Susilo, and Y. Mu. Constant-size dynamic k-taa. In *Security and Cryptography for Networks, LNCS vol. 4116*, pages 111–125. Springer-Verlag, 2006.

[3] E. Bertino and E. Ferrari. Secure and selective dissemination of XML documents. *ACM Trans. Inf. Syst. Secur.*, 5(3):290–331, 2002.

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

[5] D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano. Public-key encryption with keyword search. In *EUROCRYPT 2004, Advances in Cryptology*, 2004.

[6] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO '01: Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, pages 213–229. Springer-Verlag, 2001.

[7] D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. *Theory of Cryptography*, pages 535–554, May 2007.

[8] J. Camenisch, M. Dubovitskaya, and G. Neven. Oblivious transfer with access control. In *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*, pages 131–140, New York, NY, USA, 2009. ACM.

[9] J. Camenisch, G. Neven, and A. Shelat. Simulatable adaptive oblivious transfer. In *EUROCRYPT '07: Proceedings of the 26th annual international conference on Advances in Cryptology*, pages 573–590, Berlin, Heidelberg, 2007. Springer-Verlag.

[10] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou. Privacy-preserving multi-keyword ranked search over encrypted cloud data. In *INFOCOM 2011. 30th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 10-15 April 2011, Shanghai, China*, pages 829–837, 2011.

[11] Y. Challal and H. Seba. Group key management protocols: A novel taxonomy. *International Journal of Information Technology*, 2(2):105–118, 2006.

[12] G. Chiou and W. Chen. Secure broadcasting using the secure lock. *Software Engineering, IEEE Transactions on*, 15(8):929–934, Aug 1989.

[13] C. Cocks. An identity based encryption scheme based on quadratic residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, pages 360–363, London, UK, 2001. Springer-Verlag.

[14] S. Coull, M. Green, and S. Hohenberger. Controlling access to an oblivious database using stateful anonymous credentials. In *Irvine: Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography*, pages 501–520, Berlin, Heidelberg, 2009. Springer-Verlag.

[15] Cloud Security Alliance. http://cloudsecurityalliance.org/.

[16] M. Eichelberg, T. Aden, J. Riesmeier, A. Dogac, and G. B. Laleci. A survey and analysis of Electronic Healthcare Record standards. *ACM Comput. Surv.*, 37(4):277–315, 2005.

[17] P. Golle, J. Staddon, and B. Waters. Secure conjunctive keyword search over encrypted data. In *ACNS 04: 2nd International Conference on Applied Cryptography and Network Security*, pages 31–45. Springer-Verlag, 2004.

[18] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98, New York, NY, USA, 2006. ACM.

[19] D. Halevy and A. Shamir. The lsd broadcast encryption scheme. In *Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '02, pages 47–60, London, UK, 2002. Springer-Verlag.

[20] J. Li and N. Li. OACerts: Oblivious attribute certificates. *IEEE Transactions on Dependable and Secure Computing*, 3(4):340–352, 2006.

[21] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. Fuzzy keyword search over encrypted data in cloud computing. In *INFOCOM 2010. 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 15-19 March 2010, San Diego, CA, USA*, pages 441–445, 2010.

[22] G. Miklau and D. Suciu. Controlling access to published data using cryptography. In *VLDB '2003: Proceedings of the 29th international conference on Very large data bases*, pages 898–909. VLDB Endowment, 2003.

[23] M. Nabeel and E. Bertino. Attribute based group key management. Technical Report CERIAS TR 2010, Purdue University, 2010.

[24] M. Nabeel, N. Shang, J. Zage, and E. Bertino. Mask: A system for privacy-preserving policy-based access to published content. In *SIGMOD '10: Proceedings of the 2010 ACM SIGMOD international conference on Management of data*, 2010.

[25] D. Naor, M. Naor, and J. B. Lotspiech. Revocation and tracing schemes for stateless receivers. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '01, pages 41–62, London, UK, 2001. Springer-Verlag.

[26] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters. Secure attribute-based systems. In *CCS '06: Proceedings of the 13th ACM conference*

on Computer and communications security, pages 99–112, New York, NY, USA, 2006. ACM.

[27] R. Richardson. CSI Computer Crime and Security Survey. Technical report, Computer Security Institute, 2008.

[28] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Eurocrypt 2005, LNCS 3494*, pages 457–473. Springer-Verlag, 2005.

[29] N. Shang, M. Nabeel, F. Paci, and E. Bertino. A privacy-preserving approach to policy-based content dissemination. In *ICDE '10: Proceedings of the 2010 IEEE 26th International Conference on Data Engineering*, 2010.

[30] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *SP '00: Proceedings of the 2000 IEEE Symposium on Security and Privacy*, page 44, Washington, DC, USA, 2000. IEEE Computer Society.

[31] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou. Secure ranked keyword search over encrypted cloud data. *Distributed Computing Systems, International Conference on*, 0:253–262, 2010.

[32] X. Zou, Y. Dai, and E. Bertino. A practical and flexible key management mechanism for trusted collaborative computing. *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 538–546, April 2008.