POLICY BASED FILE ASSURED DELETION (FADE) ON CLOUD STORAGE

Veda Zapdekar, Manish Pirdankar, Nidhi Parab
Students IT Dept, KCCEMSR Thane (E)

Abstract Cloud Storage is an emerging business model for data outsourcing. Cloud storage will be a cost – saving business solution for unused storage and provide technical support for data backups. It can also save electric power and maintenance costs for data centers. Cloud must provide security guarantees for the outsourced data, which is maintained by third parties. This project proposes a secure overlay cloud storage system that achieves fine – grained, policy - based access control and File Assured DEletion (FADE). This scheme reliably deletes files of revoked file access policies. FADE enables to have a fine – grained control of how to delete files. To achieve this, FADE is build upon standard cryptographic techniques, such that it encrypts outsourced data files to guarantee their privacy and integrity and deletes files to make them unrecoverable to anyone. Design is geared toward the objective that it acts as an overlay system that works on cloud storage services.
Evaluation is based on the performance of FADE in terms of running time overhead and monetary cost. Performance results also depend on the deployment environment. Evaluation results show that the performance overhead of FADE becomes less significant when the size of the actual data file content increases. Thus, FADE is more suitable for enterprises that need to archive larger files with a substantial amount of data.
Keywords- Cloud computing, FADE, Cryptography, The Federated Broadcast Cloud System,FBCS.

I Introduction

The long-held dream of computing as a utility has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased. Developers with innovative ideas for new Internet services no longer require the large capital outlays in hardware to deploy their service or the human expense to operate it. They need not be concerned about over-provisioning for a service whose popularity does not meet their predictions, thus wasting costly resources, or under-provisioning for one that becomes wildly popular, thus missing potential customers and revenue. Moreover, companies with large batch-oriented tasks can get results as quickly as their programs can scale, since using 1000 servers for one hour costs no more than using one server for 1000 hours. This elasticity of resources, without paying a premium for large scale, is unprecedented in the history of IT.

I. From a hardware point of view, three aspects are new in Cloud Computing.

II. 1. The illusion of infinite computing resources available on demand, thereby eliminating the need for Cloud Computing users to plan far ahead for provisioning.

III. 2. The elimination of an up-front commitment by Cloud users, thereby allowing companies to start small and increase hardware resources only when there is an increase in their needs.

IV. 3. The ability to pay for use of computing resources on a short term basis as needed (e.g., processors by the hour and storage by the day) and release them as needed, thereby rewarding conservation by letting machines and storage go when they are no longer useful.

The construction and operation of extremely large-scale, commodity-computer datacenters at low-cost locations was the key necessary enabler of Cloud Computing, for they uncovered the factors of 5 to 7 decrease in cost of electricity, network bandwidth, operations, software, and hardware available at these very large economies of scale. These factors, combined with statistical multiplexing to increase utilization compared a private cloud, meant that cloud computing could offer services below the costs of a medium-sized datacenter and yet still make a good profit.

II Related Work

In FADE, active data files that remain on the cloud are associated with a set of user-defined file access policies (e.g., time expiration, read/write permissions of authorized users), such that data files are accessible only to users who satisfy the file access policies. FADE generalizes time – based file assured deletion into a more fine – grained approach called file assured deletion in which data files are assuredly deleted when the associated file access policies are revoked and become obsolete. The design intuition of FADE is to decouple the management of encrypted data and cryptographic keys, such that encrypted data remains on third-party (untrusted) cloud storage providers, while cryptographic keys are independently maintained and operated by a quorum of key managers that altogether form trustworthiness. To provide guarantees of access control and assured deletion, FADE leverages off-the-shelf cryptographic schemes including threshold secret sharing and attribute-based encryption (ABE) and performs various cryptographic key operations that provide security protection for basic file upload/download operations.
The design of FADE is based on the thin-cloud interface, meaning that it only requires the cloud to support the basic data access operations such as put and get. Thus, FADE is applicable for general types of storage back ends, as long as such back ends

provide the interface for uploading and downloading data. On the other hand, in the context of cloud storage, we need to specifically consider the performance metrics that are inherent to cloud storage, including data transmission performance and monetary cost overhead.

B. COMPLICATIONS IN FILE DELETION

File deletion is all about control. This used to not be an issue. Your data was on your computer, and you decided when and how to delete a file. You could use the delete function if you didn't care about whether the file could be recovered or not, and a file erase program. If you wanted to ensure no one could ever recover the file.

As we move more of our data onto cloud computing platforms such as Gmail and Facebook, and closed proprietary platforms such as the Kindle and the iPhone, deleting data is much harder.

You have to trust that these companies will delete your data when you ask them to, but they're generally in doing so. Sites like these are more likely to make your data inaccessible than they are to physically delete it. Facebook is a known culprit: actually deleting your data from its servers requires a complicated procedure that may or may not work. And even if you do manage to delete your data, copies are certain to remain in the companies' backup systems. Gmail explicitly says this in its privacy notice.

Online backups, SMS messages, photos on photo sharing sites, smartphone applications that store your data in the network: you have no idea what really happens when you delete pieces of data or your entire account, because you're not in control of the computers that are storing the data.

This notion of control also explains how Amazon was able to delete a book that people had previously purchased on their Kindle e-book readers. The legalities are debatable, but Amazon had the technical ability to delete the file because it controls all Kindles. It has designed the Kindle so that it determines when to update the software, whether people are allowed to buy Kindle books, and when to turn off people's Kindles entirely .

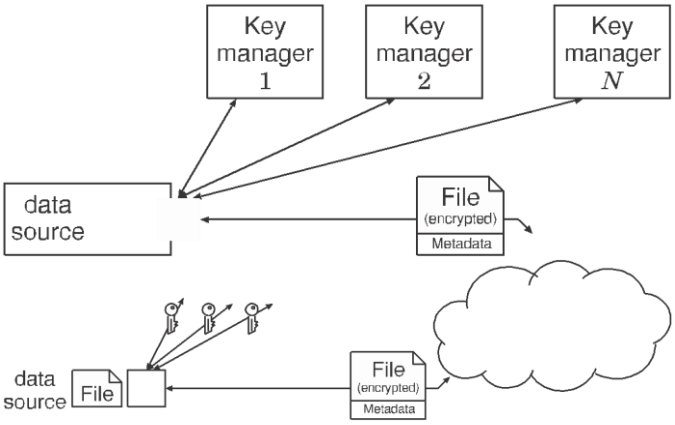
C Forward-Secure Broadcast Encryption Scheme

A broadcast encryption Scheme allows the sender to securely distribute data to a dynamically changing set of users over an insecure channel.

The main idea of the Subset-Cover framework [27] is defining a family *F* of subsets of the universe of users in the system. Moreover, each subset owns a key, which is available to all users belonging to the given subset. To broadcast a message to the users except those in some sets of revocation users R, firstly a provider covers the set of privileged users using subsets from the family *F*. This is done by identifying a partition of $\epsilon \setminus R$, where all the subsets are elements of *F*. Then, the provider encrypts the message for all the subsets in that partition. To decrypt, a user $u \notin R$ first identifies the subset in the partition of $\epsilon \setminus R$ to which he belongs, and then recovers private keys of from his secret information. The subsets are organized into groups, and a private key, groupkey, is owned to each of these groups. Such an organization of the subsets of the family *F* produces a hierarchy, in which the leaves are elements of *F* and each internal node corresponds to a group of subsets. The complete explanation of the algorithm is found in [14].

D FADE OVERVIEW

A system that provides guarantees of access control and assured deletion for outsourced data in cloud storage is called FADE.



The cloud hosts data files on behalf of a group of FADE users who want to outsource data files to the cloud based on their definitions of file access policies. FADE can be viewed as an overlay system atop the underlying cloud. It applies security protection to the outsourced data files before they are hosted on the cloud.

From all proviruses work, theirs protocols may be highly vulnerable to a side-channel attacks on the stored keys, because their protocols do not evolve a forward-secure mechanism which is important to guarantee the privacy of past transactions if the long-term key or current session keys are compromised.

III. Federated Broadcast Cloud System (FBCS)

A. System Architecture

In this paper, we propose a new idea about FBCS. According to fs-BE HIBE scheme [14]. The layered cloud security architecture is illustrated in Fig.1; we establish a unified root Global Private Key Generator Broadcast (GPKGB) in FBCS to provide a trust service for different clouds such as the private cloud, the public cloud, and the community cloud. The GPKGB is responsible for managing top-level Broadcast Private Key Generator (BPKG) authorities in the system that allocates hierarchical identities to users in their domains. Those clouds are the first level, and users in those clouds are leaves in that hierarchical. Moreover, each cloud is a self-contained, and its domain with an independent BPKG responsible for certificating cloud entities in its own domain.

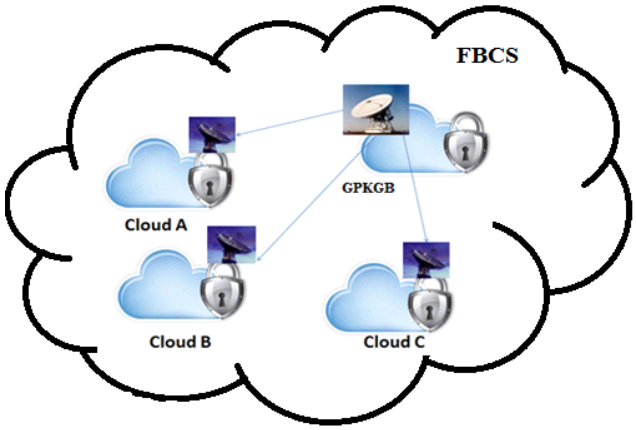


Fig2. The Federated Broadcast Cloud System (FBCS)

IMPORTANCE OF SECURITY IN CLOUD

The way cloud has been dominating the IT market, a major shift towards the cloud can be expected in the coming years. Already organizations have started moving into the cloud and a few of them includes: Schneider Electric implementing their CRM solutions on salesforce.com SaaS platform, Japanese automaker Toyota's pact with Microsoft to develop a new content delivery network for its automobiles on the latter's Azure cloud computing platform. More and more IT organizations will be moving into the cloud and with the emergence of NoSQL built around the technologies like Hadoop/HBase and Cassandra, collecting and using massive amount of data is no more considered as a headache.

Questions such as how a cloud infrastructure is built will be superseded by how and in what way, to better utilize the cloud. Enhanced cross cloud connectivity and integration wherein different cloud deployment models will be integrated to provide a better infrastructure with feasible data migration options. Increasing tablet use and file based collaboration techniques will give way to cloud based service deployment models and an increased user-base in the cloud. Technologies like Ruby on rails, HTML5 will continue to improve cloud experience in comparison to legacy options. Mobile cloud computing is expected to emerge as one of the biggest market for cloud service providers and cloud developers. Split processing techniques will come into picture and will be an enabling platform for mobile devices. Although cloud computing has revolutionized the computing UNC to a private cloud of the Cairo University, then it picks a random s1 Z*q and keeps it secret.

Thirdly, the GPKGB generates an initial master key to a private cloud. For example, UNC as PKUNC = H1 (t || UNC) and GPKGB can generate an initial master key as $KUNC_{t,ID}$=s0 PKUNC.

In the second case, BPKG authorizes cloud users. Firstly, the user registers his identifiable information in its own domain. Secondly, the Broadcast cloud allocates an identity such as the identity of a user Bob in the private cloud of the Cairo University can be UNC…IDBob$_{i,j}$ from a root down to a leaf in a tree. Thirdly, BPKG generates the user's private key and a public key. A user joins the cloud system at time t, and the BPKG uses its current master secret key $KUNC_{t,ID}$ to derive groupkeys given for the user , the Cloud Broadcast run recursively apply the Lower-Level Setup algorithm to derive the private key SKt,ID$_{i,j}$ Bob`s parent. Users can refresh their private keys on their own to avoid any communication overhead with any BPKG. Set the Bob`s private key $BobSK_{t,u}$ = {⟨ID$_{i,j}$,SKt,ID$_{i,j}$⟩ | 1 ⩽ i < j ⩽ n}. The BPKG and cloud users evolve their secret keys with time autonomously by calling the Update algorithm.

D Encryption and Digital Signature in a Federated Broadcast Cloud

In the cloud, Encryption and Digital Signature are most important security problems in mutual authentication between users and cloud provider, protection of data confidentiality and integrity during data transmission by encryption using secret keys. In a FBCS, any user has a unique identity. Moreover, any user can get the identity of any other users by requesting it from BPKGs. Any user enables to broadcast data. Sender encrypt the message not only with respect to the nodes in the hierarchy that represents

the subsets in the partition of $\epsilon \setminus R$, but also to the current time t. Forward-security is guarantee the privacy of past transactions if the long-term key or current session keys are compromised. A broadcast encryption scheme allows the sender to securely distribute data to a dynamically changing set of users over an insecure channel.
Federated broadcast cloud with FS-BE HIBC, a public key distribution is greatly simpler then in [25], because public key is a public during a root setup. The sender can encrypt the message as long as he knows the current time and the ID-tuple of the receiver, along with the public parameters of the system. Encryption is joining-time-oblivious, which means that the encryption does not require knowledge of when a user or any of his ancestors joined the hierarchy. Each service provider does not need to know when each user joins the system in order to broadcast the encrypted contents.
When a cloud provider wants to send a message to a group of users, first run the Cover algorithm [27] to partition the set $\epsilon \setminus R$ into the subsets $\{Si_1, j_1, ..., Si_a, j_a\}$. The cloud provider then shares a group session key with all the users in the subset. Input to encryption algorithm message M the current time period t public key PK and a set R. Then, run algorithm Encrypt (params, t, ID_h, M) to compute ciphertext. When a user wants to send a message to a group of users. The user then shares a group session key with all the users in the subset. To send a message, this party only needs to know the parameters system params and identifier ID (ID_1,. . ., ID_h) of subset given user, and then it can compute:
Cj ← Encrypt (params,ID(ID_1, . . . , ID_h),M) for all users in the subset Sij in the cover.
Then it is easy for a sender to add a digital signature [12] using its private key and for a receiver to verify a digital signature using the sender's public key. A user can use its private key, parameters, Tc is current time, and message M to generate a digital signature and sends to the receiver. Receiver and verify the signature using the parameters, message M, and the sender's ID. Signature = Signing (parameters, k, T_c, M), k is the sender's private key. Verification = (parameters, sender ID, M, T_c, Signature).

E. **A mutual Authentication Protocol for FBCS**

In this section, based on the former fs-BE HIBE scheme, a secure mutual authentication for Cloud (MAFBC) is proposed. Key agreement protocol to create a shared secret key to be guaranteed that they are indeed sharing this secret key between two users. Since Bob s private key that can only be allocated from the BPKG , thus Bob can be verified that he is a legal user of this cloud. A key agreement protocol is providing implicit key authentication to each parties. Key agreement protocol achieve perfect forward secrecy. [43, 44]. TLS [38] offers option forward Secrecy for key agreement, encryption and network authentication. Forward Secrecy greatly increases the level security over normal SSL session key communications.
According to [42], using Identity-Based Cryptography for every two parties, it is easy for them to calculate a shared secret key using its own private and public key. Two users need authentication key agreement protocol to create a shared secret key to be guaranteed them to authenticate each other.

A. Mutual authentication between user and Broadcast cloud

We assume if Bob with identity Bob _ID and a server with identity PRK_A want to authenticate each other.
1. The Bob's system generate a nonce N_Bob, and then sends to the authentication server the message M_Bob, containing the Bob _ID and an encrypted value of the nonce N_Bob, as M_Bob = E(Bob_ID|| N_Bob) . Here N_Bob is encrypted using public key PK_A. Encrypted value of N_Bob is used to achieve confidentiality.
2 BPKG_ A receives the message M_Bob,The cloud -A check subset R revocation, if user is the privileged in the system. After that BPKG_A decrypts the message M_Bob with PRK_A and extracts N_Bob as D (E (N_Bob|| Bob_ID), PRK_A).
3. The BPKG_ A generates a nonce N_SERVER, M_SERVER = E (N_Bob || N_SERVER) Then BPKG_ A generates a signature sig =H(M_SERVER) with private key PRK_A and send sig (M_SERVER) to Bob.
4. Upon receiving the message sig (M_SERVER), firstly Bob verifies the signature and decrypts the message with PRK_Bob and extract N_Bob and N_SERVER. User's system verifies that the received N_Bob, is equal to the sent N_Bob.

5. If both are equal, then sends M = E (N_SERVER) to the server.
6. Upon receiving the message M, The BPKG_ A decrypts M and then extracts N_SERVER. AS verifies that N_SERVER received is equal to N_SERVER sent.
Eventually, a shared secret key KBA between Bob and BPKGs calculated between them.
The key K established between the user's system and the cloud server can be used as a secret key for secure communication.

B. User Access Authentication Protocol In HBCSD

Also, Another mutual authentication, when a user belonging to HBCSD, A needs to access RBCSD.
Suppose the cloud A shares a secret key ShKBoA with Bob, also cloud A shares a secret key ShKBA with the cloud B.
1. Bob in broadcast cloud -A sends the request message for communicating with the broadcast cloud B to the cloud A. Bob generate a nonce number N_Bob , then Bob update the shared key by rehashing it ShKBoA1 = H(ShKBoA) and encrypts a request message with shared a secret key ShKBoA1(N_Bob || ID_Bob) .

2-Upon the receipt of the request message, the broadcast cloud -A decrypts message with private key PRK_A to retrieve ID_Bob. The broadcast cloud -A check subset R revocation, if user is privileged in the system, The broadcast cloud -A update the shared key by rehashing it ShKBA1 = H(ShKBA)and encrypt Bob 's request message to the broadcast cloud –B with a shared a secret key ShKBA1(N_Bob || ID_Bob) .
3. On receiving cloud -A's request message, The broadcast cloud-B decrypts message with private key PRK_B to retrieve ID_Bob and N_Bob, and then judges the ID_Bob's HBCSD through N_Bob. The broadcast cloud-B generates a nonce number N_CB, then the broadcast cloud -B encrypt message with a shared a secret key . The broadcast cloud-B sends a response message to broadcast cloud-A ShKBA1 (N_CA ||N_Bob).
4. Upon the receipt of the request message, the broadcast cloud -A decrypts message with private key PRK_A to retrieve N_CA. After that, the broadcast cloud -A encrypt request message to Bob with a shared a secret key ShKBoA1 (N_CA ||N_Bob).
5.Upon receiving the message , firstly Bob decrypts the message with PRK_Bob and extract N_Bob and N_CA. Bob verifies the received N_Bob, is equal to the sent N_Bob. If both are equal, Bob sends ShKBoA1 (N_CA) to broadcast cloud -A.
6. Upon receiving the message, the broadcast cloud -A decrypts message and then extracts N_CA. The broadcast cloud -A encrypt Bob's request message to the broadcast cloud –B with a shared a secret key ShKBA1 (N_CA) .
7. Upon receiving the message, The broadcast cloud –B decrypts message and then extracts N_CA. broadcast cloud –B verifies that N_CA received is equal to N_CA sent. The broadcast cloud -B sends privat key PRB ShK_BA 1 (PRB) to the broadcast cloud.
8. Upon receiving the message, the broadcast cloud -A decrypts message. Then, the broadcast cloud –A can calculate shared a secret key ShK_BB between Bob and the broadcast cloud –B with their public keys and private keys. The broadcast cloud -A send. Shared a secret key ShKBoA1 (ShK_BB) to Bob. The broadcast cloud -A send shared a secret key ShKBA1 (ShK_BB) to the broadcast cloud –B.
The cloud server trusts the user and allows him to communicate with the cloud server.
10. The key K established between the user's system and the cloud server can be used as a secret key for secure communication.

IV CONCLUSION

Cloud Computing will grow, so developers should take it into account. Regardless whether a cloud provider sells services at a low level of abstraction like EC2 or a higher level like AppEngine, cloud computing, storage and networking must all focus on horizontal scalability of virtualized resources rather than on single node performance. A practical cloud storage system called FADE, which aims to provide access control assured deletion for files that are hosted by today's cloud storage services. Associate files with file access policies that control how files can be accessed are implemented. Then the presentation of policy-based file assured deletion, in which files are assuredly deleted and made unrecoverable by anyone when their associated file access policies are revoked. The essential operations on cryptographic keys so as to achieve access control and assured deletion are explained. FADE also leverages existing cryptographic techniques, including attribute-based encryption and a quorum of key managers based on threshold secret sharing. References

[1] Amazon, "Case Studies," http://aws.amazon.com/solutions/ case-studies/#backup, 2012.
Amazon S3, http://aws.amazon.com/s3, 2010.
[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A View of Cloud Computing." Comm. ACM, vol. 53, no. 4, pp. 50-58, Apr. 2010.

[3] Dropbox, http://www.dropbox.com, 2010.
[4] R. Geambasu, T. Kohno, A. Levy, and H.M. Levy, "Vanish: Increasing Data Privacy with Self-Destructing Data," Proc. 18th Conf. USENIX Security Symp, Aug. 2009.
[5] S. Kamara and K. Lauter, "Cryptographic Cloud Storage," Proc. 14th Int'l Conf. Financial Cryptography and Data Security, 2010.

[6] R. Perlman, "File System Design with Assured Delete," Proc. Network and Distributed System Security Symp. ISOC (NDSS), 2007.

[7] B. Schneier, "File Deletion," http://www.schneier.com/blog/ archives/2009/09/file_deletion.html, Sept. 2009.

[8] Y. Tang, P.P.C. Lee, J.C.S. Lui, and R. Perlman, "FADE: Secure Overlay Cloud Storage with File Assured Deletion," Proc. Sixth Int'l ICST Conf.Security and Privacy in Comm. Networks (SecureComm), 2010.