ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (739-746), Month: April - June 2015, Available at: www.researchpublish.com

Secure and Scalable Data Management in Cloud

[1]Shaik Shakeel Ahmed, [2]Afroze Ansari

[1]M.Tech, [2]Assistant Professor, Department Of Computer Science & Engineering
[1,2] Khaja Banda Nawaz college Of Engineering, Gulbarga, Karnataka, Visvesvaraya Technological University, India

Abstract: **Cloud computing confers strong economic advantages, but many clients are reluctant to implicitly trust a third-party cloud provider. To address these security concerns, data may be transmitted and stored in encrypted form. Major challenges exist concerning the aspects of the generation, distribution, and usage of encryption keys in cloud systems, such as the safe location of keys, and serving the recent trend of users that tend to connect to contemporary cloud applications using resource-constrained mobile devices in extremely large numbers simultaneously; these characteristics lead to difficulties in achieving efficient and highly scalable key management. In this work, a model for key distribution based on the principle of dynamic data re-encryption is applied to a cloud computing system in a unique way to address the demands of a mobile device environment, including limitations on client wireless data usage, storage capacity, processing power, and battery life. The proposed cloud-based re-encryption model is secure, efficient, and highly scalable in a cloud computing context, as keys are managed by the client for trust reasons, processor-intensive data re-encryption is handled by the cloud provider, and key redistribution is minimized to conserve communication costs on mobile devices. A versioning history mechanism effectively manages keys for a continuously changing user population. Finally, an implementation on commercial mobile and cloud platforms is used to validate the performance of the model.**

Keywords: **Distributed systems, mobile computing and security.**

1. INTRODUCTION

Cloud computing is an evolutionary new model for distributed computing consisting of centralized data centres that provide resources for massively scalable units of computing. These computational facilities are delivered as a service to users over an insecure medium such as the Internet, and may be bridged to wireless packet data networks. A client of a cloud provider can address changes in demand for its processing needs by replicating applications in the cloud to many runtime instances, and by running them on cloud servers in concurrent fashion. Unanticipated burst demands such as flash traffic on a web server may be met automatically without noticeable delay. The client does not need to incur a high capital expense up front in anticipation of future application usage patterns that may be difficult to predict accurately, and could otherwise lead to outages if left unaddressed; excess capacity and idle cycles are avoided. The easy scalability of cloud applications results in equal opportunity of benefits to firms large and small.

Yet, despite all of its economic benefits, the cloud computing model poses very significant risks to its users. User data is stored and executed within the domain of the cloud, and there is little or no visibility into how the cloud is implemented and internally managed by the cloud provider. There is significant concern over the security and privacy of transactions and data permanently stored in the cloud. Dominant opinion is that data ought to be kept confidential not only from other users sharing the cloud, but also from the cloud provider itself, as much as possible. Indeed, a survey of IT executives by IDC rated security as the chief concern in the use of cloud computing services [1]; the concern is that the client requests storage of application logic and data in the cloud without assurance of exactly where it is stored, whether it is replicated or cached, how long it is kept for, and who exactly has access to it.

Despite its need for protection, cloud data must remain highly accessible, especially for a mobile user population. The goal of security researchers is to develop techniques to ensure communication security in cloud computing systems at

reasonable cost. Only by overcoming these challenges, will enterprise companies invest in and migrate to the cloud to reap its economic benefits. The topic of this work is the adaptation of a leading key management scheme, that was originally designed for a traditional client-server context, in a novel way that addresses the communication security challenges of the cloud environment. The intent is to find viable ways to protect the security of the communication between the cloud and its users, as well as to protect the privacy of the stored cloud data, in an efficient and highly scalable way manner meanwhile providing people an eased way of collecting sensor data.

The main contribution of this work is a novel solution that entails a key management scheme based on re-encryption that effectively utilizes the cloud for cryptographic computation while supporting a frequently-changing mobile user population that does not need to trust the cloud provider; novel aspects such as a versioning array, key material sharing tactics by users, and intelligent timing of re-encryptions, make it possible. A cloud-based prototype has also been built to provide real world data and demonstrate the viability of the approach. This work is the only one that the authors are aware of that provides a secure communication solution for a forward-looking cloud system accessed by potentially millions of resource-constrained device users.

2. SYSTEM MODEL

The system under study is a public cloud provider operating a centralized data centre that is accessed by a large mobile user population over an unprotected public Internet network infrastructure bridged to a wireless network. A user may access the cloud application from a mobile device such as a tablet, smartphone, or even a wireless sensor. A highly scalable multi-user cloud application is envisioned; it may service a large user population for the purpose of e-collaboration, a social network, or customer relationship management. It is continuously accessed by a multitude of heterogeneous mobile device users. Each mobile device user typically opens a direct TCP/IP connection to a cloud application portal. Users may upload and download content to and from the cloud by having the cloud-hosted front end access a data partition provisioned by a cloud's blade server. Each data partition is made accessible to a set of authorized mobile users. The client organization is responsible for granting access rights. The cloud provider is not fully trusted, and although it may assist in enforcing those rights for users, it cannot gain access to the data itself. This notion is in keeping with Gartner's notion of cloud computing security implying a shared environment in which data is segregated and encrypted [2]. Members of the same group, such as a work project team or a social community having common interests, will typically require access to a common data partition resident in the cloud storage system. The objective is to subdivide and isolate all cloud data into partitions and enforce appropriate access rights on each. Robustness of the cloud provider is not a significant concern in this study, as a cloud by design is typically engineered as a distributed system with data replication, reliable servers, multiple endpoints, and other safeguards that virtually guarantee its continuous operation.

3. RELATED WORK

There is a clear need for a scalable and efficient key management solution for cloud computing systems, but so far it has not been fully addressed in commercial cloud systems. The Key Management Interoperability Protocol (KMIP) addresses the issue of interoperability of key management services, but fails to account for the unique scalability potential in cloud systems and the performance problems that can result. OpenID is an open-source Single Sign-On (SSO) solution that permits a single login to access different sites and resources, but effectively the same password is used to access multiple sites, with no fine-grained access control. A traditional approach to communication security has been centralized key management, which requires public-key certificates to be generated by the authority and deployed to all users before communication can occur. In a highly scalable system, the authorization server may become overloaded as a result of this responsibility. Security enforcement based on monitoring of user behaviour can mitigate these performance concerns, such as in TrustCube [7], but the cloud provider must be entrusted with aggregated data on user contexts and activities, thus relaxing the trust model. In Certificateless Public Key Cryptography (CLPKC) [8], the Key Generation Centre (KGC) residing in the cloud does not have access to users' private keys, but the KGC would need to ensure that partial private keys would be delivered securely to the right users using some secure, or out-of-band, transport. Broadcast encryption may be employed, in which the key manager generates symmetric keys for multiple users, but whenever the membership changes, then new keys must be rebroadcast to all users, which is an unrealistic proposition in a highly scalable system.

The high turnover of cloud user membership poses a great challenge; expensive re-keying operations are normally required whenever group membership changes. In the Logical Key Hierarchy [9] scheme, the processing time per request scales linearly with the logarithm of group size, and the signing of rekey messages increases server processing time by an order of magnitude. Another approach is distributed key management, where multiple distributed public key generators (PKGs) hold shares of a master key using the concept of threshold decryption [10], or portions of a private key are distributed among users [11]. The problem with these approaches is that a user must assemble a key from multiple sources, resulting in expensive communication sessions. Data re-encryption is gaining traction as a viable mechanism for controlling access to data stored in the cloud. Re-encryption has previously been applied to an encrypted file storage system, where a content owner encrypts blocks of content with unique, symmetric content keys that are then encrypted using an asymmetric master key to form a lockbox [12]. Concerns include the following: the content owner manages access control for all other users, which is a great burden if the owner is a mobile device user; it requires dynamic re-encryption of the same data whenever multiple users want to access it; access rights need not be enforced by individual users; and it is possible for a single user to divulge the keys of all other users to the cloud provider.

A related work proposes the merging of attribute-based encryption with proxy re-encryption while attempting to offload re-encryption activity to the cloud provider [13]. However, the data owner (originator) is involved in generating a key for each new user that joins or leaves the system, which is not only a prohibitive cost for a mobile user, but also impractical due to the user's mobility and hence occasional unavailability. A secret key must be regenerated and re-distributed for each user, in lazy fashion, whenever user revocation occurs. Also, the data re-encryption activity is aggregated in lazy fashion, rather than on-demand. Similar limitations are evident in another related approach that combines Hierarchical Identity-Based

Encryption (HIBE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE), which uses hierarchical domain masters to distribute user keys and uses the cloud provider to re-encrypt data on user revocation depending on the attribute keys held by the revoked user [14]; this is done at the cost of increased storage requirements for key material held by users and a greater amount of processing when generating ciphertext. Another method of trusted data sharing over untrusted cloud providers has been proposed that uses a progressive elliptic curve encryption scheme [15]. However, it relies upon a writer uploading encrypted data to the cloud, then distributing credentials to the cloud to perform re-encryption, and also to the reader on each data access attempt; this is clearly impractical when applied to resource-constrained devices.

4. MANAGER-BASED RE-ENCRYPTION

The overall goal of this work is to explore, adapt, and evaluate system security engineering techniques to achieve a high level of communication security for cloud computing systems. A key management scheme is now described that is closely based on the original work suggested in [12]; however, it has been mapped to a cloud computing system. Its primary involvement here is to demonstrate a technique that will serve as a foundation and point of comparison for the novel scheme proposed in the following Section V. Some novel variations of the original scheme are still suggested here, however. The scheme permits access to a common data partition in the cloud among multiple users, ensures confidential data storage not privy even to the cloud provider, and offers greater data access efficiency in a mobile-based cloud system at lower overall communication and processing cost than traditional centralized solutions; all of these features are accomplished through the process of data re-encryption. Table I summarizes the notation used.

TABLE I
LEGEND FOR THE SYMBOLIC NOTATION USED IN THE DESCRIPTION OF THE KEY MANAGEMENT MODELS.

Symbol	Description
P	Cloud data partition
U_P	User group with authorized access to P
M	Manager or trusted proxy
A, B, C	Users Alice, Bob, Charlie
m	Plaintext
C_x	Ciphertext encrypted using key x
PK_{X_v}	Public key of entity X (with version v optionally specified)
SK_{X_v}	Private key of entity X (with version v optionally specified)
$RK_{X \to Y}$	Re-encryption key for converting from content unlocked by SK_X to that unlocked by SK_Y

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (739-746), Month: April - June 2015, Available at: www.researchpublish.com

A manager, or trusted proxy node, controls the access of its users to the cloud. This manager is typically under the control of the client organization, and ensures that key management functions need not be outsourced to an untrusted cloud provider. The manager may comprise a server situated behind the firewall of the client organization that is securely accessed by a mobile user population. At the same time, the cloud stores user data in encrypted form such that it is accessible to all authorized users at any time; it does so by regularly performing one-way re-encryption of the data in the cloud as it is being accessed, so that a reader in the authorized user group can decode it using the reader's own private key.

A. System operation:

1) Key generation and encryption: Consider a proxy reencryption scheme [12], based on the BBS encryption method [19] and the El Gamal crypto-system [20]. The proof of the underlying encryption technique is presented in [12], and is assumed here. The manager generates public and private keys (PKX and SKX) for each user X belonging to the system, and is responsible for maintaining an access control list for enforcing the authorized user set. A data partition P in the cloud is accessible by a user group UP and belongs to the entire set of partitions P. In this example, Manager M manages the access of user group UP to data partition P. Note that a single user may belong to multiple groups. A high-level diagram is shown in Figure 1.

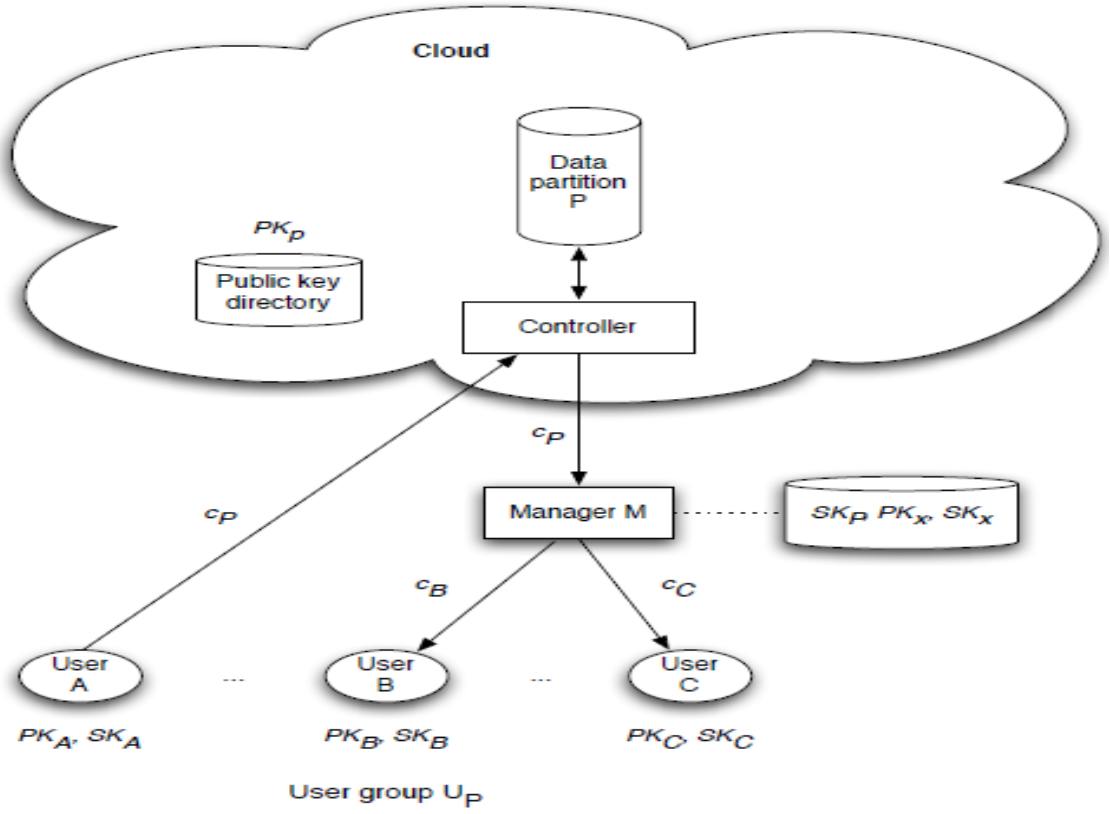


Fig. 1. Model of key management using manager-based re-encryption

The system model of the proposed work is shown in Fig. 2.

Let G1, G2 be groups of prime order q with a bilinear map such that: $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. The system parameters are the random generator $g \in \mathbb{G}_1$ and

$Z = e(g, g) \in \mathbb{G}_2$. A secret key SKX is randomly selected for each user $X \in U_P$.

Let: $SK_X = x \in \mathbb{Z}_q^*$. A public key PKX is also chosen for user X as follows: PKX = gx. Similarly, the manager M also creates a private key $SK_P = p \in \mathbb{Z}_q^*$ and public key PKP = gp for Data Partition P in the cloud that it manages. The public partition key may reside in a directory inside the cloud that is accessible by all users in the system, or be distributed to all users in UP by the manager; it is considered public information. The manager, however, retains the

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (739-746), Month: April - June 2015, Available at: www.researchpublish.com

private decryption key SKP required to read the cloud data; the cloud provider and other users cannot decode the data even if they download it directly from the cloud, with or without authentication. A unique property of this model is that all read requests initiated by users are normally serviced through the manager.

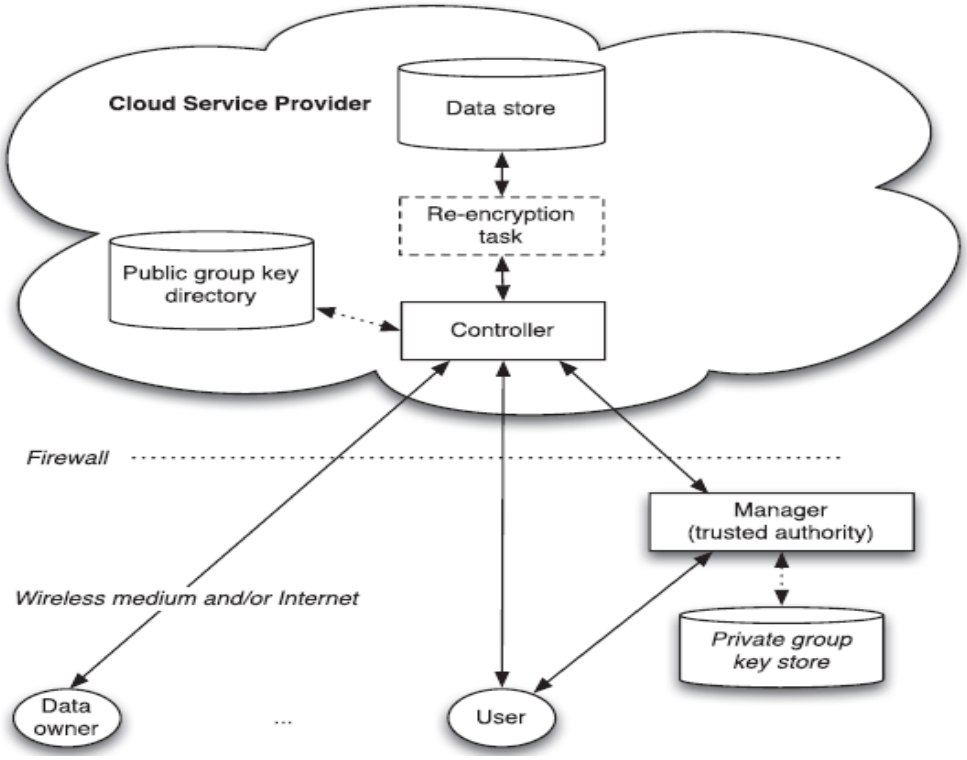


Fig. 2. Cloud computing system model

User A, Alice, encrypts a message $m \in \mathbb{G}_2$ using random $r \in \mathbb{Z}_q^*$ and the public key PKP of the data partition where it is to be stored, and uploads the cipher-text Cp to the cloud, where $C_p = (Z^r \cdot m, g^{pr})$, so that it is stored in encrypted form in partition P. The cloud provider will be unable to extract the original content m.

2) Re-encryption: Suppose that a user B, Bob, belonging to the same group, makes a request to the cloud provider for the same message m stored earlier by Alice. The cloud provider does not send it to B directly; instead, it sends it to M, which decides whether that data should be accessible by B based on its Access Control List (ACL). If so, then the manager creates a re-encryption key $RK_{P \to B}$ using the private key of the partition. The manager then fetches the encrypted message Cp from the cloud, and computes a re-encryption key using B's private key SKB. Note that SKB is equal to $b \in \mathbb{Z}_q^*$, chosen randomly by M. In general, the re-encryption key computed for user X in UP is: $RK_{P \to X} = g^{\frac{SK_X}{SK_P}}$.

For user B, as in this example, the re-encryption keycomputed is $RK_{P \to B} = g^{\frac{b}{p}}$. Using this key, M re-encrypts the ciphertext Cp as Cb and sends it to B directly.

Compute:

$$e(g^{pr}, RK_{P \to B}) = e(g^{pr}, g^{\frac{b}{p}}) = Z^{br}$$

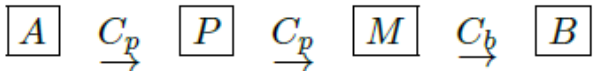
$$\text{Publish: } C_b = (Z^r \cdot m, Z^{br})$$

3) Decryption: The recipient B can then decode the ciphertext Cb using his own private key SK_B: $m = \frac{Z^r \cdot m}{(Z^{br})^{\frac{1}{b}}}$

If the original user Alice wished to decrypt the message, then a similar process would unfold; the manager would create a re-encryption key $RK_{P \to A}$ and Alice would decrypt her ciphertext Ca using her private key SKA. Thus, the manager can allow any user within the group to access the encrypted data stored within the cloud. Here, first-level encryption is

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (739-746), Month: April - June 2015, Available at: www.researchpublish.com

demonstrated [12], where the content published by the manager may be decrypted only by the holder of SKB; the content may not be re-encrypted a second time and read by a third party such as user C in UP . If C requires access, then the use of RKP!C to carry out a re-encryption of Cp to Cc is required. The flow of ciphertext in the system is as follows:



4) Key re-generation: If a new user Charlie, or C, joins the group, then he registers with the manager which grants authorization, and is given a decryption key SKC. C will be able to receive and decrypt only the content that the manager is willing to re-encrypt for him, as ciphertext Cc. If C leaves the group, then the manager removes him from its access list; it will no longer re-encrypt data for C on a retrieval attempt.

5. PROPOSED ALGORITHM

The proposed algorithm for key generation, distribution, and usage is now described. It consists of key management techniques that ensure highly secure data outsourcing to the cloud in a highly scalable manner for mobile cloud computing applications. In the discussion that follows, the following improvements are proposed to the basic functions of the original CP-ABE scheme, such that key components have been reassigned to the various entities in the system model to achieve scalable key management while reducing the mobile user computational and communication workload:

- A single authority does not generate all key material; the mobile data owner and cloud entity cooperate to jointly compute keys. The cloud provider has insufficient information to decode the user data that it permanently stores; yet, it assists in the distribution of a portion of the whole key material to all authorized users to minimize the communication cost for the data owner.
- The cloud possesses highly scalable computational ability, unlike a resource-constrained mobile user; a trusted manager also has greater computational resources. Pairing operations, which are the most

6. DISCUSSION

6.1Security Analysis:

The proposed scheme offers a dual layer of security through the combination of attribute-based encryption and public key encryption techniques, the latter which may be optionally applied. The security proof for the underlying attribute-based cryptography appears in [4] and is relied upon here. The security intuition of the proposed scheme is now described.

In the basic case, where only attribute-based encryption is utilized, the data owner generates the owner secret key OSK and does not share it. The public form of the key, the owner public key OPK, is used to generate the data secret key DSK necessary to decrypt the data. The data secret key is unique to each collection of attributes sufficient to decrypt the data. Thus, the security of the system relies upon the trustworthiness of the data owner in keeping the owner key secret; the owner has the freedom of sharing its data with any user directly, anyway. If a user lacking the required attributes attempts to decrypt the ciphertext, then such an attempt will be unsuccessful, as the wrong access tree will be input. Note that the requisite set of attributes to perform decryption may be shared by multiple users who are all equally privileged.

6.2Performance Analysis:

Critically, performance implications are modest for mobile users. The data owner must only perform exponentiation operations during its key generation phases, while the manager performs the more expensive pairing operation during partition key generation in the SETUP algorithm.

Also, with the assistance of the manager, the user performs only a single pairing operation on decryption. The group key may be shared among a group or simply possessed by a single user; the tradeoff made in its use is the required distribution of the group key and the extra pairing operation required during the encryption phase, but it is advantageously computed by the cloud provider. The manager can also assist with distribution. Furthermore, it may be advantageous for the manager to compute new key versions and re-encryption keys, and manage their storage and distribution. A suitable key versioning mechanism is suggested for this purpose, such as the one found in [9].

7. IMPLEMENTATION

The proposed protocol was implemented and profiled to gauge its performance. It was realized on popular existing commercial platforms, including the Google Android mobile and the Google App Engine (GAE) cloud platforms. A simulation calibrated to the performance benchmarks was then run to examine the scalability of the proposed algorithm.

7.1Performance Measurement:

An existing implementation in Java [19], [20] that relies upon the original CP-ABE scheme [4] served as the baseline implementation. From this starting point, the implementation was significantly rebuilt to reflect the proposed protocol described herein. The implementation uses the Java pairing-based cryptography library (jPBC) version 1.2.1 [21], a port of the pairing-based cryptography library (PBC) in C [22]. The use of Java 6 Standard Edition permits the protocol to be ported to a wide range of computing environments. Although a version of the implementation in C or C++ was not tested, the Java-based jPBC library has been found to have comparable performance to the C-based PBC library [20]. An implementation in Java is useful to evaluate because of its development kit support on popular mobile platforms such as Android and BlackBerry, and cloud platforms such as the Google App Engine and Amazon web services.

8. SUMMARY

A cryptographic protocol based on data re-encryption has been adapted to a cloud computing system model in order to gauge its viability in improving communication security and supporting highly scalable and secure cloud computing applications serving an extremely large mobile device user population. A novel protocol based on data re-encryption has been proposed that leverages the cloud provider's scalability to perform the required re-encryption tasks inside the cloud itself, rather than inside the manager; at the same time, this must occur without granting the cloud provider access to sufficient key material to decode the user data. The manager, as a trusted authority is only responsible for key re-generation, but the evolving key material to construct iterations of secret keys can be securely shared through the cloud provider itself, resulting in a more efficient and scalable security protocol.

REFERENCES

[1] P.K. Tysowski and M.A. Hasan, "Hybrid Attribute-Based Encryption and Re-Encryption for Scalable Mobile Applications in Clouds," Technical Report 13, Centre for Applied Cryptographic Research (CACR), Univ. of Waterloo, 2013.

[2] R.L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Comm. ACM,vol. 26, no. 1, pp. 96-99, Jan. 1983.

[3] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications," Proc. Ninth ACM SIGCOMM Conf. Internet Measurement Conf. (IMC '09), pp. 280-293, 2009.

[4] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," Proc. IEEE Symp. Security and Privacy (SP '07), pp. 321-334, 2007.

[5] . A. Tassanaviboon and G. Gong, "OAuth and ABE Based Authorization in Semi-Trusted Cloud Computing: Aauth," Proc. Second Int'l Workshop Data Intensive Computing in the Clouds (DataCloud-SC '11), pp. 41-50, 2011.

[6] X. Liang, R. Lu, and X. Lin, "Ciphertext Policy Attribute Based Encryption with Efficient Revocation," Technical Report BBCR, Univ. of Waterloo, 2011.

[7] J. Hur and D.K. Noh, "Attribute-Based Access Control with Efficient Revocation in Data Outsourcing Systems," IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 7, pp. 1214-1221, July 2011.

[8] G. Zhao, C. Rong, J. Li, F. Zhang, and Y. Tang, "Trusted Data Sharing over Untrusted Cloud Storage Providers," Proc. IEEE Second Int'l Conf. Cloud Computing Technology and Science (CLOUDCOM '10), pp. 97-103, 2010.

[9] P.K. Tysowski and M.A. Hasan, "Towards Secure Communication for Highly Scalable Mobile Applications in Cloud Computing Systems," Technical Report 33, Centre for Applied Cryptographic Research (CACR), Univ. of Waterloo, 2011.

[10] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage," ACM Trans. Information and System Security, vol. 9, pp. 1-30, Feb. 2006.

[11] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-Based Access Control in Social Networks with Efficient Revocation," Proc. Sixth ACM Symp. Information, Computer and Comm. Security (ASIACCS '11), pp. 411-415, 2011.

[12] Q. Liu, G. Wang, and J. Wu, "Clock-Based Proxy Re-Encryption Scheme in Unreliable Clouds," Proc. 41st Int'l Conf. Parallel Processing Workshops (ICPPW), pp. 304-305, Sept. 2012.

[13] J.-M. Do, Y.-J. Song, and N. Park, "Attribute Based Proxy Re- Encryption for Data Confidentiality in Cloud Computing Environments," Proc. First ACIS/JNU Int'l Conf. Computers, Networks, Systems and Industrial Eng. (CNSI), pp. 248-251, May 2011.

[14] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute Based Data Sharing with Attribute Revocation," Proc. Fifth ACM Symp.Information, Computer and Comm. Security (ASIACCS '10),pp. 261-270, 2010.

[15] Y. Ming, L. Fan, H. Jing-Li, and W. Zhao-Li, "An Efficient Attribute Based Encryption Scheme with Revocation for Outsourced Data Sharing Control," Proc. First Int'l Conf. Instrumentation, Measurement, Computer, Comm. and Control, pp. 516-520, 2011.

[16] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing," Proc. IEEE INFOCOM '10, pp. 534-542, 2010.

[17] K. Yang and X. Jia, "Attributed-Based Access Control for Multi- Authority Systems in Cloud Storage," Proc. IEEE 32nd Int'l Conf. Distributed Computing Systems (ICDCS), pp. 536-545, 2012.

[18] K. Yang, X. Jia, K. Ren, and B. Zhang, "DAC-MACS: Effective Data Access Control for Multi-Authority Cloud Storage Systems," Proc. IEEE INFOCOM, pp. 2895-2903, 2013.

[19] J. Wang, "Java Realization for Ciphertext-Policy Attribute-Based Encryption," http://github.com/wakemecn, 2012.

[20] A.D. Caro and V. Iovino, "jPBC: Java Pairing Based Cryptography," Proc. IEEE Symp. Computers and Comm. (ISCC), 2011.

[21] A.D. Caro, "Java Pairing-Based Cryptography Library," http:// libeccio.dia.unisa.it/projects/jpbc/, 2012.

[22] B. Lynn, "PBC (Pairing-Based Cryptography) Library," http:// crypto.stanford.edu/pbc/, 2012.

[23] G. Wang, Q. Liu, and J. Wu, "Hierarchical Attribute-Based Encryption for Fine-Grained Access Control in Cloud Storage Services," Proc. 17th ACM Conf. Computer and Comm. Security (CCS '10), pp. 735-737, 2010.