ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

A Survey on Key-Aggregate Cryptosystem for Scalable Data Sharing In Cloud Storage

T.Divya
M.Tech Student,
Computer Science Engineering Department,
GITAM School of Technology, Hyd.

Mr. Arif Mohammad Abdul
Assistant Professor,
Computer Science Engineering Department,
GITAM School of Technology, Hyd.

Abstract:

We propose a perfect key aggregate system with scalable sharing of data in cloud. This scheme provide secure data storage and retrieval. Along with the security the access policy is also hidden for hiding the user's identity. This scheme is so powerful since we use aggregate encryption and string matching algorithms in a single scheme. The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but concluding the power of all the keys being aggregated. The scheme detects any change made to the original file and if found clear the error's.The algorithm used here are very simple so that large number of data can be stored in cloud without any problems. The security, authentication, confidentiality are comparable to the centralized approaches.

Keywords:

Aggregate key cryptosystem, Cloud storage, data sharing, key-aggregate encryption.

INTRODUCTION:

Cloud is gaining popularity recently. In professional settings, we see the hike in demand for data outsourcing, which help in the strategic management of useful data. It is also used as a main technology behind many online services for personal applications and some other applications.

Nowadays, it is very easy to apply for free accounts for email, file sharing and/or remote access, with storage size more than 25GB (or a few dollars for more than 1TB). Together with the modern technology, users can access almost all of their files and emails by a mobile phone in any corner of the world. Storing data in cloud reduce the risk.

Considering data privacy, a traditional way to ensure it is to rely on the server to enforce the access control after authentication which means any unexpected privilege escalation will expose all those data. In a shared tenancy cloud computing environment becoming worse even more. Data from different clients can be hosted on individual virtual machines (VMs) but reside on a separate single physical machine. Data in a target VM could be stolen by any another VM co-resident with the target one. Regarding availability of files, there are several type of cryptographic models which go as far as allowing a third party auditor to check the availability of files on behalf of the data owner without leaking anything about the data, or without compromising the data owner's anonymity.Likewise, cloud users probably will not hold the strong belief that the cloud server is doing a good job in terms of confidentiality.

A cryptographic solution, with proven security relied on number-theoretic assumptions is more acceptable, whenever the user is not perfectly happy with trusting the security of the VM or the honesty of the staff. These users are motivated to encrypt their data with their own keys before uploading them to the server. Cloud is a market-oriented distributed computing system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements (SLAs) established through negotiation between the service provider and consumers. In cloud computing, users can outsource their computation and storage to servers (also called clouds) using Internet. Clouds can provide several types of services like applications (e.g., Google Apps, Microsoft online), infrastructures (Nimbus), and platforms to help developers write applications (Windows Azure).Security is needed because data stored in clouds is highly sensitive, for example, medical records and other social networks.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

User privacy is also required so that the cloud or other users do not know the identity of the user. Thus it is a complex system which possess highly securable processes. Consider that Alice sends all her private photos or any other important data's on Drop box or any other cloud application, and she does not want to make visible her photos to everyone. Due to various data exposing possibility Alice cannot feel relieved by just depends on the security protection mechanisms provided by Drop box, so she encrypts all the photos using her own keys before uploading for security. One day, Alice's friend, Bob, asks her to share the photos taken over all these years. Alice can then use the share function of Drop box, but the problem now is how to delegate the decryption rights for these photos owned to Bob. A possible option Alice can choose is to securely send Bob the secret keys involved for authenticating the data's.Naturally, there are two extreme ways for her under the traditional encryption paradigm:

- Alice encrypts files with a single encryption key and gives Bob the corresponding secret key directly.

- Alice encrypts all files with distinct keys and sends Bob the corresponding secret keys.

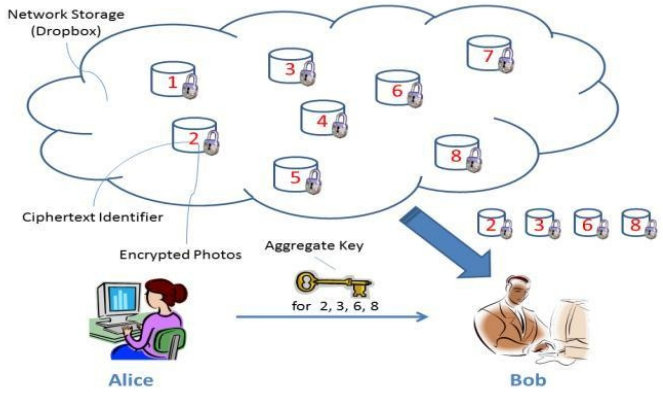
Clearly, the first method is not much secure because all unchoosen data may be also leaked to Bob. For the second method, there are some practical issues on symmetric encryption, when Alice wants the data to be came from a third party, she has to give the encryptor her secret key; Clearly, this is not always desirable. By contrast, the encryption key and decryption key differs in public-key encryption. The use of public-keyThe above two methods didn't provide security completely. The key handling process looks very simple but not promising. Thus our proposed system will solve these two problems by providing a proper security structure. These two problems are very difficult and should be sorted out. encryption gives more flexibility for our applications.

For example, in an organisation every employee can upload encrypted data on the cloud storage server without the knowledge of the company's master-secret key. Therefore, the best solution for the above problem is that Alice encrypts files with distinct public-keys, but only sends Bob a single (constant-size) decryption key.

The decryption key should be sent via a secure channel or authenticated channel and kept secret, small key size is always desirable. For example, we can't expect large storage for decryption keys in the resource-constraint devices like smart phones or wireless sensor nodes. Especially, these secret keys are usually stored in the tamper-proof memory, which is relatively expensive .However, not much has been done about the key itself.

1.1 Our Contribution:

In latest cryptography area, a fundamental problem we often study is about leveraging the secrecy of a small piece of knowledge into the ability to perform cryptographic functions (e.g. encryption, authentication) several times. In this paper, we study how to create a decryption key more powerful in the sense that it allows decryption of multiple cipher texts, without increasing its size. Specifically, our problem statement is – " To design an efficient public-key encryption scheme which supports flexible delegation in the sense that any subset of the cipher texts (produced by the encryption scheme) is decryptable by a constant-si4EWze decryption key (generated by the owner of the master-secret key)."



We now solve this problem by introducing a different type of public-key encryption which we call key-aggregate cryptosystem (KAC). In KAC, users encrypt a message not only under a public-key, but also under an identifier of cipher text called class. That means the cipher texts are further categorized into various classes. The key owner holds a master-secret called master-secret key, which can be used to extract secret keys for various different classes.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Importantly, the extracted key have can be an aggregate key which is as compact as a secret key for a single class, but aggregates the power of many such keys, i.e., the decryption power for any subset of cipher text classes. With our solution, Alice can simply send Bob a single aggregate key via a secure e-mail. Now Bob can download the encrypted photos from Alice's Drop box space and then use the same aggregate key to decrypt these encrypted photos.The sizes of cipher text, public-key, master -secret key and aggregate key in our KAC schemes are all of constant size. The public system parameter has size linear in the number of cipher text classes, but only a small part of it is needed each time and it can be fetched on demand from large cloud storage. Previous results may achieve a similar property featuring a constant-size decryption key, but the classes need to conform to some pre-defined hierarchical relationship. Our work is flexible in the sense that this constraint is eliminated, that is, no special relation is required between the classes. The detail and other related works can be found in Section 3. We propose several concrete KAC schemes with different security levels and extensions in this article.

All constructions can be proven secure in the standard model.

2. RELATED WORK:

This section we compare our basic KAC scheme with other possible solutions on sharing in secure cloud storage.

2.1 Cryptographic Keys for a Predefined Hierarchy:

We start by discussing the most relevant study in the literature of cryptography/security. Cryptographic key assignment schemes (e.g., [11], [12], [13], [14]) aim to minimize the expense in storing and managing secret keys for general cryptographic use. Data management is the important aspects of cloud. So this area must be checked clearly. Securing huge amount of data is very much risky. So our proposed scheme is made to solve all these problems. The below tabular column depicts our scheme efficiency with other various schemes.

	Decryption key size	Ciphertext size	Encryption type
Key assignment schemes for a predefined hierarchy (e.g., [7])	most likely non-constant (depends on the hierarchy)	constant	symmetric or public-key
Symmetric-key encryption with Compact Key (e.g., [8])	constant	constant	symmetric-key
IBE with Compact Key (e.g., [9])	constant	non-constant	public-key
Attribute-Based Encryption (e.g., [10])	non-constant	constant	public-key
KAC	constant	constant	public-key

TABLE 1
Comparisons between our basic KAC scheme and other related schemes

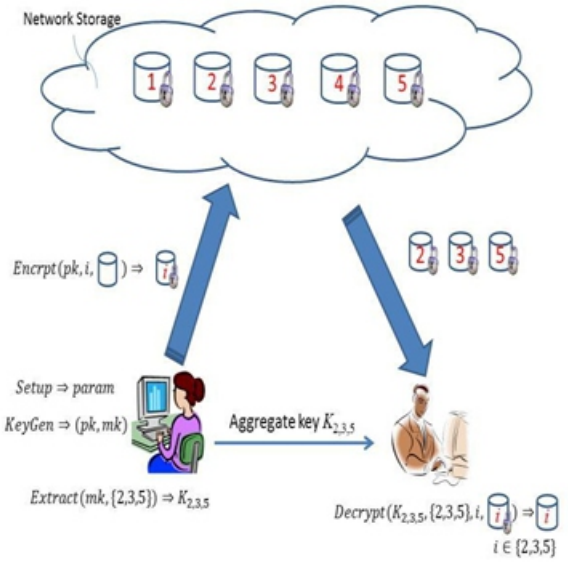


Fig. 2. Using KAC for data sharing in cloud storage

Utilizing a tree structure, a key for a given branch can be used to derive the keys of its decreasing nodes. Just giving the parent key implicitly grants all the keys of its descendant nodes. Sandhu [15] proposed a method to generate a tree hierarchy of symmetric keys by using repeated evaluations of pseudorandom function/ block-cipher on a fixed secret. The concept can be generalized from a tree to a graph. More advanced cryptographic key assignment schemes support access policy that can be modeled by an acyclic graph or a cyclic graph [16], [17], [7]. Most of these schemes produce keys for symmetric-key cryptosystems, even though the key derivations may require modular arithmetic as used in public-key cryptosystems, which are generally more expensive than "symmetric-key operations" such as pseudorandom function. We take the tree structure as an example.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

take the tree structure as an example. Alice can first classify the cipher text classes according to their sub-jects like Figure 3. Each node in the tree represents a secret key, while the leaf nodes represents the keys for individual cipher text classes. Filled circles represent the keys for the classes to be delegated and circles circumvented by dotted lines represent the keys to be granted. Note that every key of the non-leaf node can derive the keys of its descendant nodes.

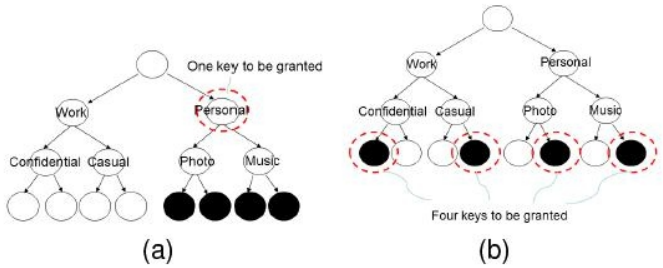


Fig. 3. Compact key is not always possible for a fixed hierarchy

In Figure 3(a), if Alice needs to share all the files in the "personal" category, she only needs to grant the key for the node "personal", which automatically grants the delegatee the keys of all the descendant nodes. This is the ideal case, where most classes to be shared belong to the same branch and thus a parent key of them is sufficient.

However, it is still difficult for general cases. As shown in Figure 3(b), if Alice shares her demo music at work with a colleague who also has the rights to see some of her personal data, what she can do is to give more keys, which leads to an increase in the total key size. One can see that this approach is not flexible when the classifications are more complex and she wants to share different sets of files to different people.

For this delegatee in our example, the number of granted secret keys becomes the same as the num-ber of classes. In general, hierarchical approaches can solve the problem partially if one intends to share all files under a certain branch in the hierarchy. On aver-age, the number of keys increases with the number of branches. It is unlikely to come up with a hierarchy that can save the number of total keys to be granted for all individuals simultaneously.

3 CONCRETE CONSTRUCTIONS OF KAC:

Let G and GT be two cyclic groups of prime order p and ^e : G_G ! GT be a map with the following properties: _ Bilinear: 8g1; g2 2 G, a; b 2 Z, ^e(ga 1 ; gb 2) = ^e(g1; g2) ab. _ Non-degenerate: for some g 2 G, ^e(g; g) 6= 1. G is a bilinear group if all the operations involved above are efficiently computable. Many classes of elliptic curves feature bilinear groups.

3.1 Public-Key Extension:

If a user needs to classify his ciphertexts into more than n classes, he can register for additional key pairs. Each class now is indexed by a 2-level index in f(i; j)j1 _ i _ `; 1 _ j _ ng and the number of classes is increased by n for each added key. Since the new public-key can be essentially treated as a new user, one may have the concern that key aggregation across two independent users is not possible. We achieve "local aggregation", which means the secret keys under the same branch can always be aggregated.

We use a quaternary tree for the last level just for bet-ter illustration of our distinctive feature. Our advantage is still preserved when compared with quaternary trees in hierarchical approach, in which the latter either del-egates the decryption power for all 4 classes (if the key for their parent class is delegated) or the number of keys will be the same as the number of classes. For our approach, at most 2 aggregate keys are needed in our example. Below we give the details on how encryption and decryption work when the public-key is extended, which is similar to the "pn-approach" [31].

• Setup and KeyGen: Same as the basic construction.

• Extend(pkl; mskl): Execute KeyGen() to get (vl+1; l+1) 2 G _ Zp, output the extended public and master-secret keys as pkl+1 = (pkl; vl+1); mskl+1 = (mskl; l+1)

• Encrypt(pkl; (a; b);m): Let pkl = fv1; _ _ _; vlg. For an index (a; b); 1 _ a _ l; 1 _ b _ n, pick t 2R Zp, output the ciphertext as C = hgt; (vagb)t;m _ ^e(g1; gn)ti

• Extract(mskl; Sl): Let mskl = f1; 2; _ _ _ ; lg. For a set Sl of indices (i; j); 1 _ i _ l; 1 _ j _ n, get gn+1_j = g_n+1_j from param, output: KSl = (Y(1;j)2Sl g1 n+1_j ; Y (2;j)2Sl g2 n+1_j ; _ _ _ ; Y (l;j)2Sl gl n+1_j)

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

• Decrypt(KSl ; Sl; (a; b); C): If (a; b) =2 Sl, output ?. Otherwise, let KSl = (d1; _ _ _ ; dl) and C = hc1; c2; c3i. Output the message: m = c3 _ ^e(da _ Q (a;j)2Sl;j6=b gn+1_j+b; c1) ^e(Q (a;j)2Sl gn+1_j ; c2)

Just like the basic construction, the decryption can be done more efficiently with the knowledge of i's. Correctness is not much more difficult to see:

c3 _ ^e(da _ Q (a;j)2Sl;j6=b

gn+1_j+b; c1) = ^e(Q (a;j)2Sl

gn+1_j ; c2)

= c3 _ ^e(Q (a;j)2Sl
ga

n+1_j _ Q (a;j)2Sl;j6=b gn+1_j+b; gt)

=^e(Q (a;j)2Sl gn+1_j ; (vagb)t)

=c3 _ ^e(Q (a;j)2Sl;j6=b gn+1_j+b; gt)=^e(Q (a;j)2Sl

gn+1_j ; gt b)

= m _ ^e(g1; gn)t=^e(gn+1; gt) = m:

We can also prove the semantic security of this extended scheme. The proof is very similar to that for the basic scheme and therefore is omitted. The public-key of our CCA construction to be presented below can also be extended using the same Extend algorithm.

4 PERFORMANCE ANALYSIS:

4.1 Compression Factors:

For a concrete comparison, we investigate the space requirements of the tree-based key assignment approach we described in Section 3.1.

This is used in the Complete Sub tree scheme, which is a representative solution to the broadcast encryption problem following the well-known Subset-Cover framework .

It employs a static logical key hierarchy, which is materialized with a full binary key tree of height h and thus can support up to 2h cipher text classes, a selected part of which is intended for an authorized delegatee.In an ideal case as depicted in Figure 3(a), the delegate can be granted the access to 2hs classes with the possession of only one key, where hs is the height of a certain sub tree (e.g., hs = 2 in Figure 3(a)). On the other hand, to decrypt cipher texts of a set of classes, sometimes the delegatee may have to hold a large number of keys, as depicted in Figure 3(b).

Therefore, we are interested in na, the number of symmetric-keys to be assigned in this hierarchical key approach, in an average sense.We assume that there are exactly 2h cipher text classes, and the delegatee of concern is entitled to a portion r of them.

That is, r is the delegation ratio, the ratio of the delegated cipher text classes to the total classes. Obviously, if r = 0, na should also be 0, which means no access to any of the classes; if r = 100%, na should be as low as 1, which means that the possession of only the root key in the hierarchy can grant the access to all the 2h classes.

Consequently, one may expect that na may first increase with r, and may decrease later. We set r = 10%; 20%; _ _ _ ; 90%, and choose the portion in a random manner to model an arbitrary "delegation pattern" for different delegatees. For each combination of r and h, we randomly generate 104 different combinations of classes to be delegated, and the output key set size na is the average over random delegations.

We tabulate the results in Table 2, where h = 16; 18; 20 respectively6. For a given h, na increases with the delegation ratio r until r reaches _ 70%. An amazing fact is that, the ratio of na to N(= 2h+1 _1), the total number of keys in the hierarchy (e.g., N = 15 in Figure 3), appears to be only determined by r but irrelevant of h.

This is because when the number of cipher text classes (2h) is large and the delegation ratio (r) is fixed, this kind of random delegation achieves roughly the same key assignment ratios (na=N). Thus, for the same r, na grows exponentially with h.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

h	r	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.95
16	n_a	6224.8	11772.5	16579.3	20545.8	23520.7	25263.8	25400.1	23252.6	17334.6	11670.2
	$\frac{n_a}{N}$	4.75%	8.98%	12.65%	15.68%	17.94%	19.27%	19.38%	17.74%	13.23%	8.90%
18	n_a	24895.8	47076.1	66312.4	82187.1	94078.8	101052.4	101594.8	93025.4	69337.4	46678.8
	$\frac{n_a}{N}$	4.75%	8.98%	12.65%	15.68%	17.94%	19.27%	19.38%	17.74%	13.23%	8.90%
20	n_a	99590.5	188322.0	265254.1	328749.5	376317.4	404205.0	406385.1	372085.2	277343.1	186725.4
	$\frac{n_a}{N}$	4.75%	8.98%	12.65%	15.68%	17.94%	19.27%	19.38%	17.74%	13.22%	8.90%

TABLE 2
Compression ratios for different delegation ratios and tree heights

We can easily estimate how many keys we need to assign when we are given r and h. We then turn our focus to the compression7 factor F for a certain h, i.e., the average number of delegated classes that each granted key can decrypt. Specifically, it is the ratio of the total number of delegated classes (r2h) to the number of granted keys required (na).

Certainly, higher compression factor is preferable because it means each granted key can decrypt more cipher texts. Figure 5(a) illustrates the relationship between the compression factor and the delegation ratio. Somewhat surprisingly, we found that F = 3:2 even for delegation ratio of r = 0:9, and F < 6 for r = 0:95, which deviates from the intuition that only a small number of "powerful" keys are needed for delegating most of the classes. We can only get a high (but still small) compression factor when the delegation ratio is close to 1.

5 CONCLUSION AND FUTURE WORK:

How to protect users' data privacy is a central question of cloud storage. With more mathematical tools and simulations, cryptographic schemes are getting more versatile and often involve multiple keys for a single application. In this article, we consider how to "compress" secret keys in public-key cryptosystems which support delegation of secret keys for different cipher text classes in cloud storage.

No matter which one among the power set of classes, the delegatee can always get an aggregate key of constant size. Our approach is more flexible than hierarchical key assignment which can only save spaces if all key-holders share a similar set of privileges. A limitation in our work is the predefined bound of the number of maximum cipher text classes. In cloud storage, the number of cipher texts usually grows rapidly without any restrictions.

So we have to reserve enough cipher text classes for the future extension. Otherwise, we need to expand the public-key. Although the parameter can be downloaded with cipher texts, it would be better if its size is independent of the maximum number of cipher text classes. On the other hand, when one carries the delegated keys around in a mobile device without using special trusted hardware, the key is prompt to leakage, designing a leakage resilient cryptosystem [22].

REFERENCES:

[1] S. S. M. Chow, Y. J. He, L. C. K. Hui, and S.-M. Yiu, "SPICE -Simple Privacy-Preserving Identity-Management for Cloud Environment,"in Applied Cryptography and Network Security - ACNS2012, ser. LNCS, vol. 7341. Springer, 2012, pp. 526–543.

[2] L. Hardesty, "Secure computers aren't so secure," MIT press, 2009, http://www.physorg.com/news176107396.html.

[3]C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Secure Cloud Storage," IEEE Trans. Computers, vol. 62, no. 2, pp. 362–375, 2013.

[4]B. Wang, S. S. M. Chow, M. Li, and H. Li, "Storing Shared Dataon the Cloud via Security- Mediator," in International Conference on Distributed Computing Systems - ICDCS 2013. IEEE, 2013.

[5]S. S. M. Chow, C.-K. Chu, X. Huang, J. Zhou, and R. H. Deng, "Dynamic Secure Cloud Storage with Provenance," in Cryptography and Security: From Theory to Applications - Essays Dedicated to Jean-Jacques Quisquater on the Occasion of His 65th Birthday, ser. LNCS, vol. 6805. Springer, 2012, pp. 442–464.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

[6]D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," in Proceedings of Advances in Cryptology - EUROCRYPT '03, ser. LNCS, vol. 2656. Springer, 2003, pp. 416–432.

[7]M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, "Dynamic and Efficient Key Management for Access Hierarchies," ACM Transactions on Information and System Security (TISSEC), vol. 12, no. 3, 2009.

[8]J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records," in Proceedings of ACM Workshop on Cloud Computing Security (CCSW '09). ACM, 2009, pp. 103–114.

[9]F. Guo, Y. Mu, Z. Chen, and L. Xu, "Multi-Identity Single-Key Decryption without Random Oracles," in Proceedings of Information Security and Cryptology (Inscrypt '07), ser. LNCS, vol. 4990. Springer, 2007, pp. 384–398.

[10]V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted data," in Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06). ACM, 2006, pp. 89–98.

[11]S. G. Akl and P. D. Taylor, "Cryptographic Solution to a Problem of Access Control in a Hierarchy," ACM Transactions on Computer Systems (TOCS), vol. 1, no. 3, pp. 239–248, 1983.

[12]G. C. Chick and S. E. Tavares, "Flexible Access Control with Master Keys," in Proceedings of Advances in Cryptology – CRYPTO '89, ser. LNCS, vol. 435. Springer, 1989, pp. 316–322.

[13]W.-G. Tzeng, "A Time-Bound Cryptographic Key Assignment Scheme for Access Control in a Hierarchy," IEEE Transactions on Knowledge and Data Engineering (TKDE), vol. 14, no. 1, pp. 182– 188, 2002.

[14]G. Ateniese, A. D. Santis, A. L. Ferrara, and B. Masucci, "Provably-Secure Time-Bound Hierarchical Key Assignment Schemes," J. Cryptology, vol. 25, no. 2, pp. 243–270, 2012.

[15]R. S. Sandhu, "Cryptographic Implementation of a Tree Hierarchy for Access Control," Information Processing Letters, vol. 27, no. 2, pp. 95–98, 1988.

[16]Y. Sun and K. J. R. Liu, "Scalable Hierarchical Access Control in Secure Group Communications," in Proceedings of the 23th IEEE International Conference on Computer Communications (INFOCOM '04). IEEE, 2004.

[17]Q. Zhang and Y. Wang, "A Centralized Key Management Scheme for Hierarchical Access Control," in Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '04). IEEE, 2004, pp. 2067–2071.